UNIVERSITY OF SOUTHAMPTON

Faculty of Physical and Applied Science
School of Electronics and Computer Science
Web and Internet Science Group

Learning to program in a connected world: A study of 1st Year Undergraduates

By
Jian Shi

February 2016

A thesis submitted in partial fulfilment for the degree of Doctor of Philosophy

Supervisors: Dr. Su White, Professor. Hugh C. Davis

Abstract

Every year thousands of students arrive at university to be taught an introduction to programming. Any introductory course needs to be technically robust and utilise an appropriate range of methods to establish common foundation skills on which to build in subsequent years. Recent technological changes provide many more opportunities for student to gain access to resources which can support learning formally and informally. Understanding how students go through this process can be valuable for instructors, whilst crafting courses, which realistically develop independent learning skills using online materials, will equip students to develop their technical skills and expertise throughout their future careers.

Rapid and on-going technological changes mean that often students and working programmers everyday practices are rather different from those which were predominant when much of the existing literature on learning to program was written. Those texts date from an era when learning to program and computer science educational resources were restricted to paper, labs and the lecture room. However, as the Web is changing the world, so it also changes students' approaches to learning formally and informally.

This study focuses on investigating learning practice of undergraduates and asks the question how do learning practices evolve as programmers move from novice to experienced. The subjects of the study are undergraduates who were taught how to program in an introductory programming module in the University of Southampton

A mixed methodology which combined quantitative and qualitative approaches was adopted, to gather evidence throughout the process as the students learnt to program, focussing on identifying does when learning to program. The findings are based on surveys, interviews and observations that were administered over the duration of a programming course for undergraduates.

This study presents and analyses the findings of an investigation into the attitudes and behaviours of first year undergraduates learning to program in this changed context. It also proposes, a systematic way to gather evidence which can be used to review or change the overall learning design of such teaching. A reusable research framework is presented, which could be applied not only those learning to program but also in broader educational settings.

The study uses a range of measures to determine students' strategies to cope with course requirements, and their use of adjunct resources are evaluated and documented. The thesis provides evidence which may be especially valuable for academics preparing and revising teaching programmes offering insights into what different types of students actually do when learning to program especially how they integrate and evolve their use of the Web to enhance their learning.

Table of Content

Figure List

Table List

Declaration of Authorship

I, Jian Shi, declare that this thesis titled "Learning to program in a connected world: A study of 1st Year Undergraduates" and the work presented in it are my own and has been generated by me as the result of my own original research.

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;

2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;

3. Where I have consulted the published work of others, this is always clearly attributed;

4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;

5. I have acknowledged all main sources of help;

6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;

7. None of this work has been published before submission.

Signed: …….

Date: …….

1 Introduction

1.1 Overview

We have become an information society and the processing of information quickly and accurately is a vital part of the infrastructure that enables us to conduct social discourse on a global basis. No doubt modern society has progressed much further than Marshall McLuhan envisioned in 1954 when he coined the phrase, "global village”, to express his vision of an electronic nervous system consisting of media, technology, and communications. Today, however, the creation, distribution and use of information are a substantial part of our economic, political, and cultural lives. At the heart of this global information exchange is a massive network of computers that make this communication possible. As the globally networked society shares such vast amounts of information, it will increasingly require services, both automated and human, to manage and interpret tremendous amounts of information. As the amount of information increases, the demand for people who can structure and manage what information is processed will become even more valuable to help fulfil economic, political and cultural needs that will emerge to shape the workplace and society.

In this global digital information network, computer programming is the medium that makes these connections possible. Whether the intent is to design and develop computer programs, software or applications, the basic gateway for entering this global dialogue is an understanding and capability of conversing in the lingua franca of the digital universe, computer programming. Whilst the realisation of the intrinsic and extrinsic value of computer programming knowledge may only come after a certain degree of mastery, nonetheless, there are thousands of students every academic year, who arrive at universities and will be taught an introduction to programming in first contact modules. According to the statistics from the HESA (Higher Education Statistics Agency), in the UK, in 2010/2011, 25,465 new undergraduate students registered for the subject of computer science, including 21,140 full-time students. It is essential to be conscious of the fact that the backgrounds and prior experience of the new student cohorts are certainly varied.

There is extensive evidence that academics in universities continue to explore better teaching patterns to ensure an equal outcome for all students, seeking to balance a technically sound introduction which can also sustain motivation across the cohort. However, keeping up to date with pedagogical methods and changing life experience of students makes this task more complex. There is a significant volume of literature in this area, but recent technological changes which impact learners’ approaches to life and study have introduced additional factors, which may also need to be taken into consideration. Nonetheless, existing literature mainly discusses the teaching and learning on the basis of physical world activities and paper. However, as the Web is changing the world, it also changes students’ learning in this connected world. Therefore, it may be helpful for academics to understand what students actually do, especially how the Web changes their learning, by gathering detailed perspectives from our undergraduates.

1.2 Research Questions

Computer scientists have been interested in the recurrent problem of teaching introductory programming for years. This can be addressed through the curriculum via chosen languages and programming paradigm, and through educational approaches such as setting and scheduling purposeful activities and providing contextual information and timely teaching and support.

The work presented in this thesis is predominantly concerned with investigating factors associated with educational approaches. Due to the varied backgrounds and prior experience of new undergraduates, teachers are additionally faced with the complex task of ensuring that there are equal opportunities for all students to achieve successful/satisfactory learning outcomes. Factors which are commonly discussed are wide ranging but frequently include the importance of helping students maintain and build motivation, self-belief and work ethic.

A large volume of literature exists that addresses these last three factors and suggests strategies to tackle such issues. However, much of that literature (the majority) dates from an era that learning was almost totally based on practical activities and paper-form references and learning resources. More recently, the writings of John Biggs have been highly influential in motivating academics to adopt new perspectives on the focus of effective approaches teaching.

The work in this thesis has been driven from the assumption drawn from Biggs (2003) that by gaining insights into how the learning process operates, we can then begin to identify useful changes to the educational approach. On-going technological innovations, especially those driven by use of the Internet and the Web, are strongly impacting on students' learning approaches and are likely to continue to evolve.

Biggs argues that it is advantageous to move away from focussing on what the student is and what the teacher does, towards understanding (and later, engineering) what the student does (Biggs, 2003, p20-25). It is likely that academics engaged in instructing these courses will have learned some period of time before their students and thus may have only a limited experience of being a novice learner in a connected world.

The objective of this thesis therefore is to assist academics in understanding what students do as they learn to program, and given the need to address differing backgrounds and prior experience, whether there are differences between different sub groups within programming cohorts. This work is investigating the nature of students' attitudes and beliefs, as well as their behaviours and motivations. It is intended that the findings of this research add to the existing body of knowledge and may therefore be useful to academics who are reviewing or revising their educational approaches.

Given this context and perspective, the research study was designed to address the following research questions:

A. **Considering first year undergraduates: before being formally introduced to learning to program at university, do any differences exist, within specific sub–groups of students, between the attitudes to and beliefs about the task of learning to program?**

B. **Whilst learning to program in this connected world, to what extent do different students:**
    a. **Gain or lose confidence and motivation?**
    b. **Modify their approaches to learning as they progress?**

The advent of widespread publication of formal and informal learning materials online provides students with many more ways to customise or personalise their approaches to learning. This part of the study considers that use of online resources from the two perspectives of formal and informal learning.

C. How do students integrate their formal and informal learning activities? To what extent do students:

  a. Rely on formally provided materials and exercises?

  b. Develop individual approaches to learning which integrate real world and online activities and materials?

1.3 Contribution

The research presents the findings of a three-year long study of first year programmers learning to program at a Russell group university in the UK. It provides detailed data and analysis evidencing student attitudes and behaviours during this process.

In investigating the research questions listed above, this study will make a valuable contribution to the body of knowledge on the subject of "how 1st year undergraduates learn to program in this connected world." As it will be demonstrated, there is a dearth of information on how students actually learn how to program.

This research collected pertinent data on the effectiveness of the typical programming course curriculum and how students adapt this in order to make programming easier to comprehend and master. Accordingly, the findings demonstrate that teachers may benefit from making use of a broader range of learning opportunities beyond the classroom. These will better serve different types of learners as well as providing opportunities for students who are excelling on the course to further advance their skills. Therefore, this represents an area for further study on this topic in the future, while offering teachers recommendations on how to improve the learning experience of students.

The findings (Chapter 10) provide fresh insights for researchers and academics on how students can learn programming in the current online environment. By developing their understanding, teachers can enhance their teaching strategies. In particular, by utilising the benefits of the internet, instructors are able to offer two separate online platforms which cater to novices and pre-experienced students respectively. The former can concentrate on establishing a basic theoretical foundation, which suits beginners, whilst the latter can set more difficult tasks for pre-experienced students.

Furthermore, an efficient research framework (Figure 3.1) is suggested. It could be easily applied and has been created as a point of reference for teachers who wish to understand what their students do, not only for introductory programming modules, but also for other similar research areas with minor changes.

1.4 Thesis Structure

The structure of this thesis is evaluate a learning process that is currently in use as an introductory programming module in computer programming offered to first year undergraduate students. The demographics of the students are offered in an attempt to determine how their experiences may have been in shaped by some of these variables. Having established the population participating in the study, the rest of the dissertation is a sequential description of the rationale and for the evaluation process. The educational foundations of the study were based on relevant theories of motivation and learning theories which are discussed in detail. The methods section describes the data collection process that included direct observation, student surveys, group participation and personal interviews that were administered at different times during the course. The data analysis methods are described in detail – descriptive and inferential statistics, content analysis of surveys and interviews. The study concludes with a discussion of what was learned from the study, an evaluation of the research questions and conclusions that can be formed based on the total results. Lastly, there are recommendations made as to future research that appears to be warranted based on this study's findings.

In detail, Chapter 2 introduces the foundation of this research. In this chapter, what major educators have been done in terms of higher education (Section 2.1) is introduced first. On this basis, some research into specific factors (Section 2.2) is explained, followed by a brief introduction about formal and informal learning (Section 2.3). Subsequently (Section 2.4), the teaching at the University of Southampton is described combined with the educational theories analysed in Section 2.1 – 2.3. At the end, a table, mapping research questions and related sections, is presented (Section 2.5).

Having analysed the background literature and research content, a range of research methods are identified which can be used to investigate the research questions. Chapter 3 present in detail the general design of the experimental studies for this research (Section 3.1). Subsequently, the issues of how to select respondents (Section 3.2) and analyse data (Section 3.3) are explored.

From Chapter 4 to Chapter 9, each stage/survey of the research framework is introduced, including detailed design, preparation, data collected and data analysis. Chapter 4 describes a pilot study. The data collected could help to shape the following formal series of surveys. Chapter 5 presents the initial survey. The data collected could provide evidence for the answer to the first research questions. Chapter 5 & 6 presents weekly survey and end of term survey respectively. The collected data offer some evidence for the rest of the research questions. Meanwhile, the responses could help to design the following NGT survey and individual interviews, which are described in Chapter 8 & 9.

In Chapter 10, all the data collected from the research framework is analysed and summarized to answer all the research questions.

Chapter 11 gives the conclusion and contribution. Some possible future work is also recommended afterward.

2 Background

This chapter describes existing literature, which relates to the research questions listed in Chapter 1:

A. Considering first year undergraduates: before being formally introduced to learning to program at university, do any differences exist, within specific sub–groups of students, between the attitudes to and beliefs about the task of learning to program?
B. Whilst learning to program in this connected world, to what extent do different students:
    a. Gain or lose confidence and motivation?
    b. Modify their approaches to learning as they progress?
C. How do students integrate their formal and informal learning activities? To what extent do students:
    a. Rely on formally provided materials and exercises?
    b. Develop individual approaches to learning which integrate real world and online activities and materials?

Before addressing these research questions there is a large volume of literature which offers relevant existing insights.

This chapter explains the background of existing analysed research outcomes and relevant theories from proceedings and publications in terms of teaching and learning to program for first-year undergraduates. Some major researchers in Higher Education have presented and summarized good theories and framework to understand higher education in general, which are the basis of answering to research questions. Hence, Section 2.1 introduces some mainstream major educational influences in higher education around the world, including the theories of Kolb's experimental learning cycle (Section 2.1.1), Laurillard's conversational framework (Section 2.1.2), Marton & Säljö's surface and deep learning approach (Section 2.1.3), Lave & Wenger's legitimate peripheral participation (Section 2.1.4) and Biggs's levels about thinking teaching (Section 2.1.5).

Section 2.2 presents specific factors of this research, including issues of motivation (Section 2.2.1), the diversity of students (Section 2.2.2) and formal & informal learning (Section 2.2.3).

Section 2.3 moves onto the research in computer science education, including the existing work of how to motivate all our students (Section 2.3.1 – 2.3.2) and how computer-aided tools could enhance out teaching (Section 2.3.3).

Subsequently, detailed example about what academics do in the introductory programming module at the University of Southampton is examined in Section 2.4, mapping with the theories introduced above.

At the end of this chapter, Section 2.5 gives a table which maps all the theories introduced in this chapter to the research questions.

2.1 Major Educational Influences

Practice in Higher Education has been strongly influenced by the work of a small number of major educational researchers. The next section presents an overview of the contribution.

2.1.1 Kolb - Experiential Learning Cycle

Writing in the 1980s, Kolb proposes a model of learning in which a repeated cycle of linked educational processes enable the student to learn.

Learning could be considered as the process of building knowledge and skills through reading, listening (communication), research and practice. It can be effectively accomplished only by self-experiential practice. Based on such experiential learning, Kolb summarized a model which is known as Kolb's learning cycle model (Kolb 1984), shown in Figure 2.1.

Each stage is associated with learning activities through the four stages in the model, which can be listed as:

- **Experience**. Learners should completely immerse themselves in relevant experiential activities.
- **Reflection.** Actual experiential activities and experiences should be observed and seriously thought through from multiple aspects.
- **Abstract.** Logical conceptions and theories could be abstracted through observation and thinking.
- **Experiment.** Learners are supposed to make decisions and solve problems by adopting these theories. Furthermore, newly formed conceptions and theories will be verified in practical works.

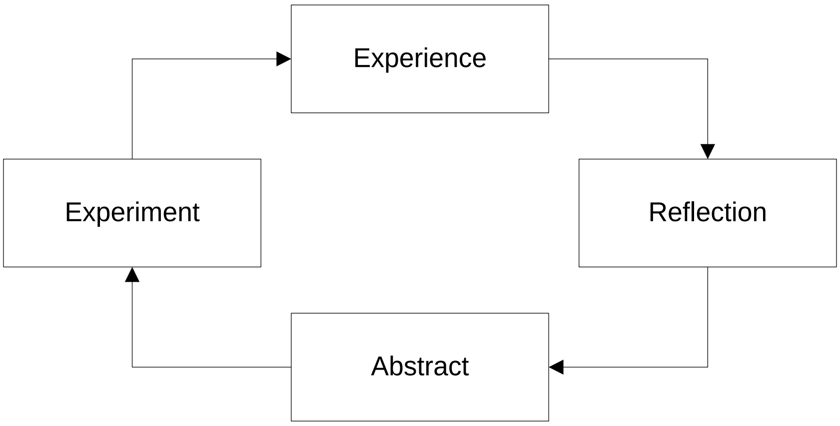

Figure 2.1 Kolb's Learning Cycle Model

As illustrated in the Figure 2.1 and the explanation above, Kolb defined learning as a process of "creating" knowledge from the transition of experiences, rather than the acquirement and transition of the content. There is no prescribed starting point in the cycle, and the cycle can be repeated until learning has been achieved. Furthermore, according to individual preferences as to where and how knowledge is accreted, in the model, Kolb proposed a theory of learning styles which incorporates four associated types of learner illustrated in Figure 2.2 and expanded Table 2.1:

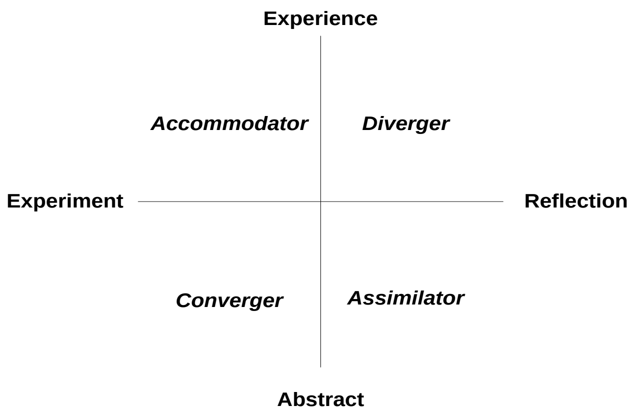

Figure 2.2 Kolb's Learning Styles(Kolb 1984)

STYLE	DERIVATION	SUITABILITY FOR LEARNING
Diverger	• Practical Experience • observation and Analysis	• Rich imagination needed for knowledge • Sensitive learning required knowledge (especially those theories with different views)
Assimilator	• Observation and Thinking • Abstract and Conceptualization	• Synthesizing multiple opinions • Forming the learning framework
Converger	• Abstract and Conceptualization • Experiment	Practice lead by theories.
Accommodator	• Experiment	Knowledge on the basis of an experiment.

Table 2.1 Components of Kolb's Learning Styles (Kolb 1985)

2.1.1.1 Mapping the learning cycle to education in practice

It is possible to map classroom experiences and learning activities which are commonly adopted in programming education against the four stages in the cycle which Kolb identifies. A suggested mapping is shown in Table 2.2.

LEARNING CYCLE	LEARNING/TEACHING STRATEGIES	FOR LEARNING TO PROGRAM
Experience	Simulation, case study, direct experience, demonstration	Code review, live programming demonstration, coding activities
Reflection	Group discussion, group work	Group project with group meeting
Abstract	Sharing, transmission	The use of student wiki
Experiment	Laboratories, interns, practical use	Practical labs, coursework

Table 2.2 Mapping typical educational approaches in teaching programming to the four stages of Kolb's Learning Cycle

Having considered these learning/teaching activities of the introductory programming module in the University of Southampton, Kolb's learning cycle model (Figure 2.1) could be re-drawn as the following figure:

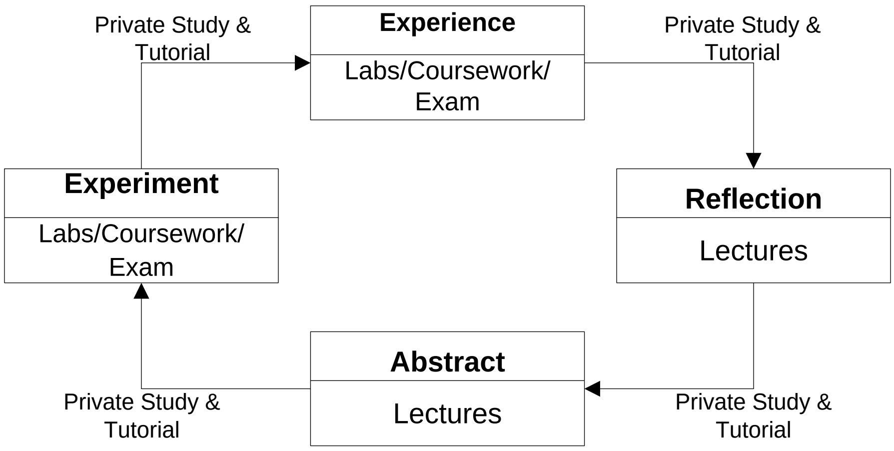

Figure 2.3 Kolb's Learning Cycle Model Identified in COMP1004

Actual examples of the mapping of Kolb's learning cycle to the curriculum under observation will be discussed in further detail in Section 2.4 below.

2.1.2 Laurillard - Conversational Framework

With the growing use of technology in the classroom, researchers began to propose theoretical models which provided alternative interpretations to the way in which learning takes place. In the UK, Laurillard's 'conversational model' has become particularly widely referenced. This model of the interaction among students, teachers and courseware was the outcome of observation and analysis of student use of educational media (Laurillard 2002) and acknowledges earlier influences of models of learning from Pask (1975). In the conversational framework student learning can be regarded as an iterative process which is active and relies on 'conversational' interactions as shown in Figure 2.3.

Activities that are needed to accomplish the process of studying are identified in the model. Based on the Laurillard's viewpoint the process of studying can be considered as a type of conversation; it may be a dialogue between student and teacher; or between student and student. Key processes within the framework can be discursive, adaptive, interactive and reflective. From this model it is clear that feedback has an important role to play in the process of learning. By implication, teachers when crafting opportunities for learning need to incorporate activities which will generate feedback, from teachers, peers or reflectively by the learner themselves.

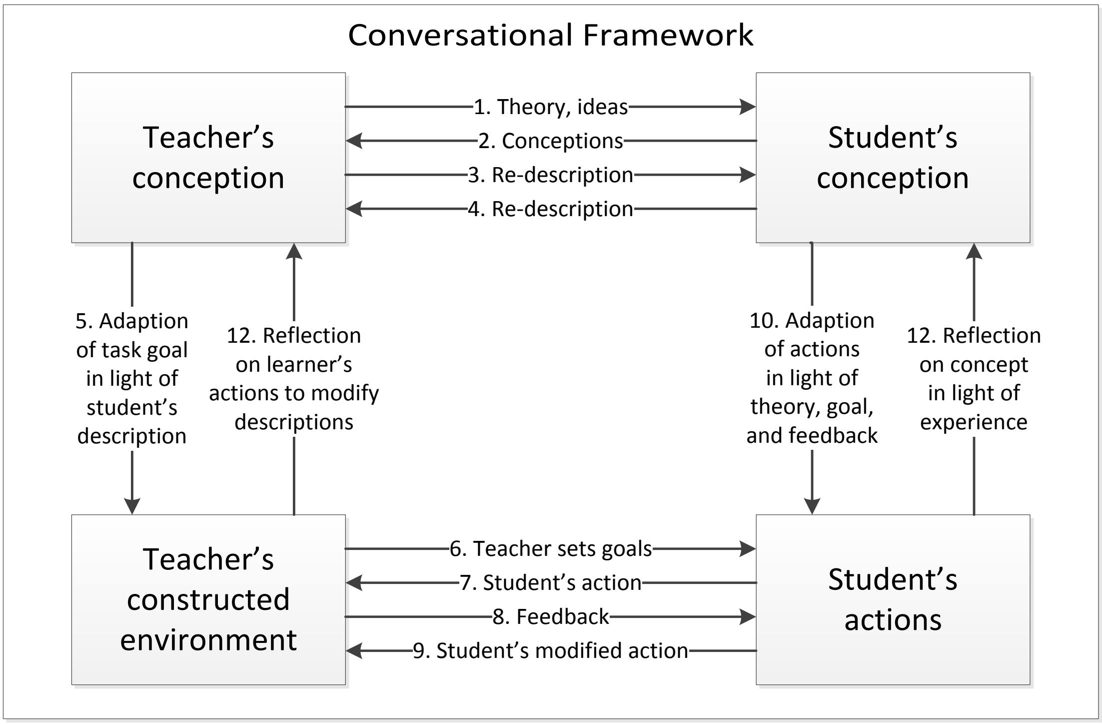

Figure 2.4 The Conversational Framework identifying the activities necessary to complete the learning process (Laurillard 2002, p87)

Laurillard focuses on the features of distinct media of teaching, which are divided by her into five types narrative, interactive, adaptive, communicative and productive. Meanwhile, she also lists associated learning activities by as well as necessary technologies for the application of these media types. As a result, the table is obtained as follows:

Learning Experience	Media Form	Appropriate media technologies
Attending, Apprehending	Narrative	Print, TV, video, DVD
Investigating, exploring	Interactive	Library, CD, DVD, Web resources
Experimenting, Practising	Adaptive	Seminar, online conference
Discussing, debating	Communicative	Laboratory, field trip, simulation
Articulating, expressing	Productive	Essay, product, animation, model

Table 2.3 Five principal media forms with the learning experiences and the methods (Laurillard 2002, p90)

- **Narrative Media:** this includes media such as digital text, video and audio. It can help the learners' comprehensions by adding coherence to the content's structure. Nevertheless, they belong to linear media and thus they can only provide the opinions, perspectives or teachers' guidance than incorporating the students' response to or reformulation of the content. As a result, what narrative media supports is only the first and non-dialogic section of the discursive studying stage.
- **Interactive Media:** systems such as hypertext, and the World Wide Web belong to non-linear media, what they support is the exploration and discovery. With the adoption of interactive media, learners are able to make connections by themselves and enquire according to their own willingness. Meanwhile, some restricted internal feedback such as the feedback coming from the activity itself is also allowed by interactive media. In addition, when the targets are set and guidance is provided by combining interactive media with narrative media, both the discursive and interactive stages of studying can be supported by interactive media.
- **Adaptive Media.** With the application of adaptive media like simulations and virtual environments enable the creation of more elaborate learning objects. Learners are more able to control their interaction with the experience of studying. At the same time, they are able to carry out tests by changing the parameters, simulate systems or environments and observe the conditions when they attempt to apply what they have learned lately to practice. Besides, they are also able to obtain more specific internal feedback and record the process of interaction so that they can think about it carefully. Thus, it can be said that not only the interactive but also the adaptive stages of studying can be supported by adaptive media, which may support the last stage of reflection at the same time.
- **Communicative Media.** The discursive aspect of studying is supported by communicative media, which include online social as well as cooperative environments. Some people discuss and state that what the communicative media that can be adopted by not only teachers but also other learners support is the second and dialogic section of the discursive stage of studying; nevertheless, an extra type of studying content is also provided by these media in the form of information and ideas; moreover, external feedback is also allowed by these media during the stages of interaction and adaption. Therefore, the communicative media also support the reflective section during the last two phases. What's more, the output of productive studying can also be provided by communicative media in such forms as wikis or blogs; whereas it is not easy for these media to support the studying stages of interaction and adaption by themselves.
- **Productive Media.** In terms of the productive media which include webpage, blog posts, digital objects, etc., an output from studying is able to be obtained by productive media. In this case, learners will clearly express and share their learning content, think carefully about their experience of studying, modify their initial attitude based on interaction and consider the importance of their experience. Besides, what productive media, with which communicative media always overlap, support is the last and reflective stage of studying.

2.1.2.1 Why it is relevant to learning to program

Laurillard's framework is aimed at defining various formal learning that is encountered and not only conventional but also digital media technologies are included in her list. However, with regard to the practitioners who study programming, a question is proposed by the framework: what are the online technologies that are most appropriate for supporting the experience range necessary for the important studying of programming to occur in the world which is interconnected?

If methods which are typically deployed in teaching programming (drawn from experience at the University of Southampton) are examined, it is possible to modify and customize approaches as shown in Table 2.4

Learning Experience	Media Form	Appropriate media technologies	Modified media technologies
Attending, Apprehending	Narrative	Print, TV, video, DVD	Lectures, printed notes, online notes
Investigating, exploring	Interactive	Library, CD, DVD, Web resources	Online tutorials, library, Web resources
Experimenting, Practising	Adaptive	Seminar, online conference	Practical labs , programming environments
Discussing, debating	Communicative	Laboratory, field trip, simulation	Online/offline discussion groups
Articulating, expressing	Productive	Essay, product, animation, model	Programming artefacts in labs, coursework, final examination

Table 2.4 Laurilard's table modified to reflect the programming learning experiences

2.1.3 Marton & Säljö – Surface and Deep Learning

Marton and Säljö have suggested a mainstreaming classification system known worldwide about "student approaches to learning". They divide learning approaches into deep and surface learning (Marton & Säljö 1976), which can be summarized in Table 2.5.

	SURFACE LEARNING	DEEP LEARNING
Characteristic	• Memorise learning materials • No related knowledge • Only repeat the exact words when assessed	• Understand learning materials • Linked to relevant knowledge • Theories can be used smoothly
Focus	• "What" • External	• "How" and "Why" • Internal
Objective	• Simply memorise objectives • For assessments	• Create a complete knowledge framework • For assessments and future practical usage

Table 2.5 Surface and Deep Learning

According to Laurillard, *"learning approaches are not the same as learning styles. Students can use different learning approaches for different tasks. These are not inherent personality traits and they are produced by the interaction of the student with the learning tasks", i.e.* whilst students learn, they may use different learning approaches to solve certain challenges depending on their objectives (Laurillard 1997).

2.1.3.1 *Why approaches to learning are relevant to the study*

Teaching programming focuses on many small practical tasks enabling the learner to systematically build their skill set alongside their knowledge and understanding of the theoretical underpinnings of computer science and software engineering. Teachers designing tasks and learning activities for introductory programming modules want to ensure that students establish deep approaches – so that they are able to establish sound foundations for subsequent study.

It seems that teachers would prefer students to take deep learning approaches all the time. However, in fact, we cannot totally ignore surface learning. In terms of learning to program, it is still useful and enough to use surface learning to remember programming keywords and syntax, whilst deep learning has to be adopted when learning object-oriented theories. Thus, the ideal learning approach for students should lie between these two extremes.

However, a student's approach and commitment to learning is an important determinant of the ultimate learning outcome. For many students, the objective of university classroom performance is to learn to memorise and reproduce knowledge in ways that are acceptable to the teacher Over time many begin to discover that learning is more than that and a university education requires more of them than rote memorization and repetition. Some seek more meaning and personal benefit for their own knowledge and understanding.

2.1.4 Lave & Wenger – Legitimate Peripheral Participation

Lave & Wenger (1991) proposed the theory of situated learning in their book "Situated Learning: Legitimate peripheral participation" in 1991. Building on work by Vygotski (1978) who emphasised the importance of the role of social interaction in the development of cognition through the concept of the Zone of Proximal Development, he summarizes that situated learning is learning in the specific situation or context where the knowledge is about to be applied.

According to Lave & Wenger, apart from being a mental process of individual sense-making, learning is also a socialised, experimental-participation process, with resource difference as a medium. Meanwhile, learning should not be simply regarded as passing the abstract, situation-free knowledge from one to another. It is a socialized process, where the knowledge is built by all those who participate in it. This kind of learning is always located in a particular situation and permeates specific social and natural environments.

Lave and Wenger propose three key concepts in their publication "Situated Learning: legitimate peripheral participation":

- Community of practice. It means a 'circle' that formed by the people doing the actual work while the new comers trying to enter the circle and gain social and cultural experience.
- Legitimate peripheral participation. It contains three aspects:
    a) 'Legitimate' indicates that all parts in the community of practice are willing to take the under-qualified new ones as the members of the community;
    b) 'Peripheral' suggests that the beginners can only follow the key members and do some trivial work until their skills improve and then get into the core of the circle;
    c) 'Participation' means learning while doing the actual work because knowledge lies in the practice of the community, and not in the books.
- Apprenticeship. It is the way of study where the mentor teaches the apprentice.

2.1.4.1 Why it is relevant to learning to program and this study

In Southampton the long established approach to first year programming seeks to create and develop a community of practice amongst the students (Davis et al. 2001). This approach has evolved as new academics join the teaching team, however the core principle have been retained. Looking at the process from Lave and Wenger's perspective it is possible to see how initially post graduate students participate in the community to assist in building the bonds required to improve students' understanding of programming concepts. Working in pairs in the lab and informally providing peer support, student learn to communicating with each other. They take this process over into the online world using commonly established programming discussion groups as well as within their own social media support such as Facebook. It is also discussed and analysed in further detail in Section 2.4.

	Description	Associated to Learning to Program
Community of practice	It means a 'circle' that formed by the people doing the actual work while the new comers trying to enter the circle and gain social and cultural experience.	Pair programming and assisted working on common set tasks in the lab provide opportunities for community building. The new first year undergraduate cohort typically ranges across novices through to pre-experienced students. Demonstrators and pre-experienced students alike can assist newcomer to enter the circle.
Legitimate peripheral participation	'Legitimate' indicates that all parts in the community of practice are willing to take the under-qualified new ones as the members of the community.	The demonstrators work actively to ensure that this objective is achieved. Successfully completing the labs will enable novice programmers to gain sufficient experience to move from 'under qualified' to being members of the community.
	'Peripheral' suggests that the beginners can only follow the key members and do some trivial work until their skills improve and then get into the core of the circle.	Teaching and class assignments are carefully structured beginning with trivial tasks developing skills and enabling student to move into the core of the circle. Students, especially novices will start the introductory programming module to build programming skills, which will remain useful to the end of their degree study and beyond.
	'Participation' means learning while doing the actual work because knowledge lies in the practice of the community, and not in the books.	The focus on the educational activities is on participation rather than on knowledge, Although there are lectures, students are expected to spend the majority of their time actually programming. They learn in stages (Peripheral, participation) moving from mastering the environment through familiarity the language to demonstrating competencies combining the use of programming constructs with algorithmic problem solving.
Apprenticeship	It is the way of study where the mentor teaches the apprentice.	Demonstrators and lecturers are programmers who previously experienced the process of gaining mastery.

Table 2.6 How introductory programming at Southampton enables legitimate peripheral participation

2.1.5 Biggs – Levels about Thinking Teaching

Biggs has been particularly influential in recent debates about the best way to model student learning. It has been observed that observed that "the way we go about accomplishing learning will of course depend on what we conceive learning to be" (Schmeck 1988).

In his earlier work Biggs develops a model of the student learning process, which consists of three different stages: presage, process and product (Biggs 1995).

- The presage stage refers to student’s approaches to prior experience and performance.
- The process stage refers to the approaches student’s use to learn the current subject
- The product stage refers to the outcome of the students in the current subject.

The labels and stages were later changed (Biggs 2003) to *surface*, *deep* and *achieving,* making them more consistent with other writers on the subject.

Biggs concluded that a student’s choice of approaches to learning or techniques were a function of prior motives for learning. Commentators in the field seek to provide explanations of the behaviours associated with each stage, for example:

“The motive for *surface* learning would be a 'pass only' strategy. This strategy is achieved with minimum effort and the learning method could be rote. The motivation for *deep* learning is 'study for study’s sake'. The learner would read widely and aim for a breadth and depth of understanding. Effectively this is a deep working strategy. Learners who wish to *achieve* are reliant on ego-enhancement and competition. Their choice of strategy is flexible and can be a combination of deep and surface learning” (Nield 2007).

Others have concluded that there is an intermediate type of learning which is termed *strategic*. In the surface approach the intention is just to cope with the task, routine memorisation and pass the course. The deep approach looks for patterns and principles and tries to learn the logic of the particular argument or concept, as well as monitoring the development of one’s own understanding (Entwistle 2000). Whereas, the strategic approach is intended to achieve the highest grades, using well-organised study methods and making good use of time. Entwistle (2000) further posits that the strategic approach focuses on subject content while being very attentive to the course’s assessment requirements. Regardless of the learning strategy the student adopts and employs, the academic result is fairly well established. In observing student behaviours and in crafting environments to learn, it becomes clear that academics take into account such factors which in turn influences their choice of tasks such as lab work as well as formative and summative assessments.

Continuing his investigation into student learning, Biggs (2003) subsequently establishes a model for thinking about these processes from a teacher’s point of view, which is summarized in Table 2.7.

	LEVEL 1	LEVEL 2	LEVEL 3
Focus	What the student is	What the teacher does	What the student does
Principles	Only fundamental activities are taken by both teachers and students.	Active teaching techniques adopted.	• Learning is promoted. • Ensure students learn. • Prevent from avoid learning.
Students' Actions	Just basic actions: attending lectures, taking notes, completing coursework and examinations.	• Basic actions (same as Level 1) • Providing feedback about teaching they received.	• Same as Level 2 • Probably getting engaged in all the learning strategies
Teaching Activities	• One-way process in terms of communication. • Little or nothing to do to guide students.	Motivation may be provided for students for engagement purposes by introducing some new active teaching techniques.	• Same as Level 2 • Start to understand students' understanding. • Design new effective teaching activities. • Suitable assessments

Table 2.7 Reworked version of Biggs's Three levels of thinking about teaching (Biggs 2003)

While arguing that level 3 is the optimal way to design teaching, Biggs demonstrates that normally, novice teachers tend to start from Level 1, and will transition smoothly to Level 3 as they gain greater professional experiences (Biggs 2003, p24).

Further discussion of this model in the context of teaching at Southampton is included in Section 2.4.

Biggs also proposes a theory called constructive alignment (Biggs 2003, p25-26), which builds a bridge between constructivist learning and achieving consistency between teaching objectives, learning outcomes and assessment processes. Constructive alignment is a principle to develop teaching, learning and assessment-related activities. Biggs argues that this is not typically achieved in the traditional format of higher education which focuses on exams and lectures.

2.1.5.1 Why it is relevant to learning to program

Biggs's theories are proposed generally for higher education. The concepts can be applied to any subject, including computer programming. His theory of three levels of understanding reminds teachers of introductory programming to review their teaching.

If we examine teaching as it is constructed in both the University of Southampton in the UK, and in the wider world of Higher Education Computer Science teaching there are various teaching strategies to improve academic's' understanding about "what the students do", At Southampton, tasks such as weekly practical labs, Ground Controller and Space Cadet provide examples of this approach (Davis et al. 2001).

As a fundamentally practical and even vocational subject the teaching of programming is clearly a task which fulfils the requirements of constructive alignment – the principles of which are long stabled as Biggs observed quoting much earlier work by Tyler (1949) “Learning takes place through the active behaviour of the student: it is what he does that he learns, not what the teacher does”.

Ben-Ari (1998) presents a comprehensive analysis of the impact of constructism in computer science education In terms of constructive alignments, through the approaches to assessment – weekly practical labs are a widespread and common practice (which also satisfy some conditions of the legitimate peripheral participation) are a mean of assessing and providing feedback on tasks which are “designed to encourage deep engagement ” (Biggs 2003, p32) this is also true for the exam which is conducted under open web in real programming environments.

2.1.6 Summary

Kolb’s experimental learning cycle could help academics to understand how our students learn. Laurillard’s conversational model also suggests a framework identifying all the necessary learning activities to complete learning processes. Marton and Säljö argue that academics should try to help students to take deep learning. The theory of community of practice proposed by Lave and Wenger could help to understand students’ formal and informal learning activities. Biggs’s theory of understanding levels suggests academics that it is important to understand “what our students do” when learning.

2.2 Research into Specific Factors

Section 2.1 has introduced the main influenced educators. Their theories are mainly focusing on the general higher education, which could be applied to learning to program. Meanwhile, some detailed specific factors are also worth being investigated in order to answer the research questions.

2.2.1 Motivation

Although Kolb’s learning cycle seems to be easy, as a matter of fact, not all the students could follow it. The most significant reason is the lack of motivation. “Motivation is the key” (Carter et al. 2010; Carter et al. 2011), and, without motivation, students will not learn to program.

In his study of motivation and programming (Jenkins 2001b), Jenkins suggests that Level 1 is insufficient for students to achieve expectations, in terms of learning to program. Level 2 thinking can lead to some success. Nonetheless, there is little evidence to show that it is enough to cope with our changing students (Jenkins 1998). One possible reason for this is that the effectiveness of the approaches adopted at this level is still indistinctive (Jenkins 2001a).

Jenkins (2001b) goes on to argue that it is optimal for teachers to focus strictly on Level 3 thinking (what the student does); i.e., students should be motivated to be engaged in certain learning activities which are contrived by teachers. To achieve this goal, academics need to have a deep understanding of the students’ perspective of our teaching, as well as their learning.

However, many of the concepts about learning assume that learning is the traditional transmission model of teaching, whereby information is transferred from the teachers to the learner and the learner simply absorbs the information and applies it as appropriate. It should also be noted that teachers have a role to perform that goes beyond purveying information. It has also been argued that “students want teachers to make the subject ‘alive’, make it relevant to today’s circumstances and involve the students in the everyday interaction in the classroom, that is, to cultivate active learning” (Yap et al. 1991). This describes a teacher who is flexible in the approach to learning and uses a variety of methods to present the subject matter and to engage students in ways that make it more interesting or relevant to the learner.

The causes for the lack of motivation involve various aspects. Two important factors are locus of control, and learned helplessness.

2.2.1.1 Locus of Control

Perceptions of locus of control may increase or undermine motivation. Locus of control (Lefcourt 1982) is defined as the views of human beings towards the forces that influence peoples' lives and destinies. The loci of control can be divided into two types: internal and external. This means that individuals can demonstrate a response either to an external locus of control, or an internal locus of control.

Internals (those who demonstrate internality) believe that activities (include learning) and the results are determined by internal factors, implying that their abilities and intentions are able to control the learning process. They believe that rewards should depend on individual performance. On the other hand, externals think they are forced by destiny, luck, chance and other people. Thus, complicated and unpredictable external forces may control their performances. They believe rewards are not dependent on their performances.

Most individuals are between these two extreme poles; however, teachers need to be sensitive to the way in which they enable learners to interpret the degree of control which exists as they undertake learning activities. Different loci of control held by learners will affect their learning. The learners' perceptions will affect achievement, motivation, the input effort, attitude and performance of the task. Different learners will have different sensitivity towards rewards and the meaning of punishment and scores. It will also affect their sense of responsibility, and their attitudes towards and trust of teachers.

2.2.1.2 Learned Helplessness

Individuals who have experiences of learned helplessness may lose motivation. The concept of learned helplessness was raised by American psychologist Seligman in research on animals in 1967. It is defined as failing to respond to repeated failure or punishment (Petersen et al. 1993).

Desperation and depression caused by learned helplessness is the source of many psychological and performance problems. It is the long term bad learning performance caused the inability to learn well. They seldom experience the happiness of success no matter what they try. Frequent failure prompts them to attribute the reason for failure incorrectly. They consider themselves as ungifted, weak and not suitable for the education system, which leads to them abandoning their effort. Meanwhile there exist some students who used to achieve at a level that satisfies their own targets but is not comparable to, and frequently ignored by, others, which results in a loss of self-respect. All these form the cluster of learned helplessness. Helplessness and the unreceptiveness is a negative psychological phenomenon 'learned' and caused by repeated failure and torture instead of innate instinct.

The main source of learned helplessness is what individuals believe to have caused the situation. There are more possibilities for individuals to suffer from depression and low self-esteem when they believe that internal, uncontrollable factors caused study and psychological problems. They will take it for granted that it is impossible to enhance academic performance, which will cause lower motivation to study, and a failure to try.

Even though this phenomenon seems pessimistic, fortunately, academics are trying to solve these problems with diversified teaching approaches, discussed in greater detail in the next section.

2.2.2 Issues Associated with Students

Since this is a research project on behaviours from students, i.e. understanding of "what the students do". Academics admit that things are changing rapidly in terms of students, their learning strategies and teaching methods within the UK higher education, especially after the rise in importance of the Internet. The following sections give an overview of our current students.

2.2.2.1 Expansion

In the past two decades, class sizes in universities like Southampton have been around 70 to 100. With these relatively large numbers, not only are facilities in increasing demand (including bigger lecture rooms, more laboratories with more workstations, more tutors, etc.), but also students' backgrounds are getting more complex, and their learning requirements are difficult to balance. Academics have worried that this phenomenon will force them back to teaching surface conceptions from in-depth concepts, i.e. from problem-based learning back to traditional lectures (Jenkins 2001b).

2.2.2.2 Heterogeneity and Retention

It is undoubted that the heterogeneity of the new students is one of the more intractable challenges which confront all the academic staff of introductory programming modules. This means there may be students who are total novices, whilst some others may have programmed for a long period of time. Thus, students' expectations are varied. The overachievers can easily get bored from the basic theory lectures and stop engaging, whilst the beginners may still be confused about programming tasks encountered late in an introductory module.

According to the research findings from the ITiCSE working (Carter et al. 2010; Carter et al. 2011), such a phenomenon can force instructors to implement a suitable pedagogical method, which could meet all the students' requirements in the most efficient way. However, it is quite difficult to handle this diversity. Davis et al also point out that it is tempting for universities provide similar introductory programming materials for every participant (Davis et al. 2001). Just focusing on the novices, and neglecting the experienced candidates, is identified as one of the most "popular" teaching approaches in the past decades. Instructors sometimes assume that minimal supervision should be enough for the students with the greatest prior experience. This may be problematic. Such students frequently become bored and stop attending to the lectures. Instead of achieving the instructional objectives, they sometimes lose their enthusiasm of learning, and even fail. Internet-based courses have also been conducted to improve retention (Haungs et al. 2012).

2.2.2.3 Others

Current students may spend much less time studying than those in the last century. The reasons are varied. Numerous entertainments distract students' attention from learning. With a computer, they would do everything but learning to program. Apart from this, students may be occupied with part-time jobs due to financial problems.

In addition, differences between institutions may affect students' learning strategies. According to cognitive learning theory, the immediate environment affect's one's perceptions and also that self-regulated learning develops in contexts, not just in classrooms (Pintrich 1999). Although Ames maintains that classrooms affect individual subjective experiences and motivational outcomes (Ames 1992), it is reasonable to assume that institutional differences would have an effect on learning outcomes as well.

2.2.3 Formal and Informal Learning

Approaches to learning can be divided into two basic overarching forms: formal learning and informal learning (Scribner & Cole 1973). "Formal learning" is orchestrated, mainly refers to structured instruction accompanied with set tasks chosen to develop the learner's knowledge and understanding along an intended pathway, most often in an educational establishment such as school, college or university class, but it can also be individually with a tutor or specialist instructor. In contrast "informal learning" refers to self-initiated, self-regulated and self-responsible learning out of the time and place of formal learning, through social interaction to convey the non-teaching knowledge. Informal learning may take place anywhere, at any time, and it is integrated into people's lives, which is really important for students' learning.

In terms of learning to program, formal learning activities within this study involve lectures, practical labs, coursework and examination. Informal learning activities may include after-class group and individual study or research, particularly online learning activities using the Web. Examples include watching online tutorials, videos which demonstrate approaches to problem solving and programming, participation in discussion group (e.g facebook or other social media), and reading online commentaries or discussions by skilled and experienced programmers in specialist discussion fora such as ZDNet.

2.3 Computer Science Education

Learning to program is an activity which may be very different from the learning tasks with which most new undergraduate students are familiar. This is perhaps the reason why academics in computer science have become so concerned with this initial educational challenge. Usually universities recruit students who are judged to be academically capable of successfully completing their studies. The task therefore is to ensure that students are taught and learn in such a way that they achieve this potential, acknowledging that some students may choose to change the direction of their academic career or, for external reasons, may not be able to sustain the motivation or the workload which is necessary for ultimate success.

After the general principles we need to talk about research specifically into teaching and learning in computer science.

Alongside the educational literature explored above, there is also a large body of literature well known in the computer science education community which reflects the path of change and innovation in curriculum, teaching processes, and student learning and assessment. This section provides an overview of some of the work which is most relevant to the study.

The broader importance of motivation was discussed in Section 2.2.1. However, specific studies have focussed on the role of motivation amongst student learning to program.

2.3.1 Motivating Strugglers

Typically, students may be roughly classified as either strugglers or high achievers, and both of the two sub groups may lose motivation. Academics have tried to explore different strategies to ensure all students across the cohort are motivated.

There is a large volume of literature which addresses the approach of instructors helping strugglers to program (Barker 2009; Boyer et al. 2007; Boyer et al. 2009; Cutts et al. 2010; Gehringer & Miller 2009; Hanks et al. 2009; Kearse & Hardnett 2008; Lahtinen et al. 2005; Riordan & Traxler 2003; Rogerson & Scott 2010; Stamouli et al. 2005; Stubbings et al. 2009).

Some researchers (Ragonis & Ben-Ari 2005) identify the most difficult topics, especially for novices, and provide reasonable suggestions to help them, whilst automation is also recommended by other academics (Ala-mutka et al. 2004). Meanwhile, investigations are also undertaken to indicate students' competencies (Chen et al. 2006), as well as their programming experiences (Eidelman & Hazzan 2007), before learning officially starts. In addition, some institutions even study how to introduce inclusivity and a sense of belonging through non-technical activities, and have adopted the ideas of the extreme apprenticeship method by providing continuous feedback opportunities (Vihavainen et al. 2011).

Recently, academics have started to focus on what novices actually "think" whilst learning to program (Shinners-Kennedy et al. 2011; Caruso et al. 2011; Dyke et al. 2011; Marceau et al. 2011) aiming to generate potential innovations.

2.3.2 Motivating High Achievers

When considering working with pre-experienced and high achieving students, dropout rate is a major issue of concern to academics. According to some academics' experience (Section 2.2.2.2), not only the students who had never programmed before might fail the final examination, but also some pre-experienced students would fall into the same situation. Students may learn less than is expected by failing to engage with the learning materials (Jenkins 2001c; Jenkins 2005). Tasks which extend the students existing skills and present additional intellectual challenges are popular. Researchers suggest that "motivation is the key" (Carter et al. 2010). Activities which are stimulating and creative, often open ended, are often suggested. Robots (Lawhead et al. 2003), musical composition (Hamer 2004) and toys (Jenkins 1998) are all established and recommended to motivate high achievers.

TOPS (Teaching Over-Performing Students) project was established in order to motivate the pre-experienced students to learn programming, which is led by the University of Southampton and University of Kent with the cooperation of some other universities in the UK (White et al. 2007). There are two kinds of competitions in this project: Pair Programming, and Challenges from other Institutions. The former means that the students from each institution will be divided into a group of two, sharing one laptop, to solve several programming problems given by the academic staff. The latter one is to divide the participants to a group of eight, solving questions designed by students from other institutions.

2.3.3 Computer-Aided Tools for Learning to Program

Apart from the professional pure programming platforms just involving coding, compiling and showing output, academics never stop attempting to create more sound and interesting computer-aided tools or environments for students' learning to program. With these tools, "reflection" and "experiment" stages from Kolb's Learning Cycles would be strengthened. Most of the new tools and concepts have received a great deal of benefit in terms of helping students to successfully complete their learning processes.

Findings have been published as well. Alice, Greenfoot and Scratch are considered to be the mainstreaming Initial Learning Environments (ILE) for learning to program in higher education. Both students and teachers are satisfied with the benefits these ILEs brought (Fincher et al. 2010). Apart from these tools used world-wide, institutions may have created their own environments for students, such as Parallel Puzzle (Adams et al. 2010), a problem-oriented animated learning platform (Stone & Clark 2011) and a language-based GUI generator (Dewan & Hill 2012).

Along with the growing of the ILEs, web-based environments are growing increasingly popular as well, such as an online environment for pedagogical code review (Hundhausen et al. 2010), a web-based tool providing drill and practice support for Java programming (Denny et al. 2011), a web-based generation and delivery system for active code reading (Hoffman et al. 2011), the Java Wiki Integrated Development Environment (JavaWIDE) used for promoting active learning for programming modules (Jenkins et al. 2012) and an online forum for novice animation programmers (Scaffidi et al. 2012). In the University of Southampton, AnnAnn.Net, an online animated code annotator, is designed which assists the teaching of programming (Hooper et al. 2007; Carr et al. 2005).

2.4 What Academics Do at the University of Southampton

Looking at teaching methods deployed in the University of Southampton for the introductory programming module in AY2012-13, COMP1004 Programming Principles, it seems there is a strong reference to an experiential model of learning, which could be identified in ways summarized in the following table:

DESCRIPTION	TIME	FUNCTION
COMP1004 Programming Principles	200h	Total study
24 lectures in 12 weeks	24h	• Teachers teach the knowledge, understanding and intellectual skills. • Students develop the techniques needed to program.
1 lab/week and 3 solo exercise/lab (10 labs with 30 tasks in all) + Tutorial (Space Cadet and Ground Controller)[1]	36h	• Teachers set these hands-on labs for students to practice and consolidate what was covered in lectures. • Students reinforce their learning of skills, and supplement them with practical skills.
1 individual coursework	57h	• Consists of formative and summative assessments. • Students demonstrate their mastery of all the skills they learned, including transferable skills.
Private Study	80h	• Programming is a skill which takes practice, so students should focus on practicing, including self-study and group discussion (probably in student wiki)
1 Examination	3h	• Testing understanding of topics that it is difficult to fully assess in an assignment. • Students demonstrate that the work submitted in their assignments was their own, by being asked to reproduce some of their programming skills under exam conditions.

Table 2.8 Stages of the Learning Cycle Identified in COMP1004

In detail, Programming Principles is a double module, which is a key course for first-year undergraduate students learning to program. It presents the ideas of programming, starting from simple basic syntax to more complex theories.

2.4.1 Formal Activities

2.4.1.1 Lectures

Typically, these 24 hours of lectures are given in stages of up to three per week. In the lectures, students will listen to lecturers presenting knowledge and showing how to solve programming problems, as well as providing answers and discussing questions when needed.

These activities could be recognized as reflection and abstract according to Kolb's learning cycle (Figure 2.1), because students are observing actual experiential activities and experiences, and abstracting logical conceptions and theories during the lectures, including times when they are asked to think about problems or concepts, or to discuss specific questions in pairs or small groups. According to Laurillard's conversational framework (Figure 2.3), lecture is one important activity in Teacher's Conception and Teacher's constructed environments. It is the main "pathway" where

[1] The teaching takes a differentiated approach allowing students to self-classify according to their perceived strength either as Space Cadet (advanced prior knowledge) or Ground Controller – (limited prior knowledge).

academics could teach theories, set goals and give feedback to the students. Meanwhile, academics could encourage students to take deep learning approaches whilst learning to program (Section 2.1.3).

2.4.1.2 Practical Work

- Weekly Practical Labs

After lectures, hands-on programming labs are scheduled and conducted. Students are expected to complete a certain amount of specific programming tasks individually in the lab (although it may be that some students prefer to start them beforehand). Teaching assistants and online resources (such as Google and student wiki) are available to assist with their work. These labs will be marked every three weeks, and feedback will be provided as well. Whilst programming, the students also experience turning theories into practice in these labs.

The teaching also corporates a differentiated approach offering separated additional classes based on students'' self-evaluation of their programming expertise. Those who consider themselves to be in need of additional support, or who are progressing slowly are invited to join the ground controllers group. Those who are confident of their progress and consider they may have already mastered the necessary skill are invited to participate in the space cadets sessions.

- Tutorial - Ground Controller

These tutorials designed for anyone who needs help, especially the students without any programming experience (the former kind of students who fail the exams). It could be considered to be a support team providing help to students who get stuck while learning programming. Students have made favourable comments about this session because it is a very common experience that they might get stuck at some of the obstacles; even students with pre-existing skills would be hitting exactly the same problems as well. Also, all the participants could try to help each other before they ask the postgraduate demonstrators to offer help, which would enhance their understanding for the particular programming topics.

Ground controllers in its modern incarnation have around 90 students attend at the start of term but it drops off to about 40 by the end of the term. It is generally started by having a 10 minute recap of last week's lab with a Q&A about things people didn't understand. The focus of the class is to get students who are struggling to write code on their own (not in pairs). They each sit on a machine and are given an exercise to do using *http://codestrom.ecs.soton.ac.uk*, which is an anonymous Java programming tool with a deliberately limited feature set. The work they do is submitted to the server every time they compile. Tutorial leaders go through the examples of what students are writing anonymously on the projector to point out mistakes people are commonly making. Students can see examples of good stuff and bad stuff and get feedback in real time without me knowing whose code they are discussing.

Students have previously said that they like this format because it means they don't have to put their hand up to say something they don't understand because it will usually come up when we look at the code. From the instructors point of view it is particularly powerful because it forces the students to actually do programming learning by doing in an authentic context, which is the real way to get experience of problems like "what does this compiler error mean" and "I don’t know how I would start".

- Tutorial - Space Cadet

Having analysed the performance and behaviours of pre-experienced students, it was observed that such students might lose of their motivation, particularly early on in the module, since they often believed they would learn nothing new from the lectures. The lectures started to feel boring, and hence students with prior experience often stopped attending them. Indeed, it is unusual to teach a class which full attendance from those with prior experience.

Thus, for the purpose of pushing and challenging such students while developing and sustaining their motivation, the Space Cadets programme was created. It runs weekly activities in parallel with the core module lectures which are intended to extend and engage the students making them think deeply about what they are programming, while creating an interesting application.

A typical class will progress as follows: Self-professed good programmers go along and are given a challenge for the week. The challenge will usually be something outside the learning objectives or syllabus topics of the module but is designed to be a thought provoking challenges. Students are invited to suggest to the class ways in which they might approach this challenge. They then showcase the work which they did to complete the previous challenge talking about the code on the projector. Generally the tutorial leader will pick examples based on their diversity so students can discuss the pros and cons of the different approaches. They aim of the session is to challenge students who have experience who may be finding the material in the main part of the course too easy or boring. It is also an opportunity to grow their critical skills and presentation skills.

- Coursework and Examinations

Similarly, the individual coursework and examination require students to do the same thing, and certain feedback will also be provided to enhance students’ learning. The examination is an open-book online test. These practical works could be identified as experience and experiment according to Kolb’s learning cycle, because students are supposed to make decisions and solve problems by adopting the theories they “abstracted” from lectures, and verifying these formed conceptions and theories, whilst they are completely immersing themselves in such experiential activities.

- Summary

These learning activities provide good opportunities for academics to understand “what students do” whilst learning to program, which is important according to Biggs’s levels about thinking teaching theory (Section 2.1.5). According to Lave & Wenger (Section 2.1.4), the set tasks build a “community of practice” for our students which could enhance the “legitimate peripheral participation” (Table 2.6). Special teaching sessions, Ground Controller and Space Cadet, were designed to meet different requirements of the diverse students (Section 2.2.2), in order to motivate all the students (Section 2.2.1).

2.4.2 Informal Activities

In addition, students should expect to spend reasonable hours on private study, which mainly involves lots of informal learning activities. Such private study could be conducted in various ways, such as searching for help online (probably with Google, Facebook or Forums) as needed, reviewing some relevant technique newsletters, chatting with friends in coffee time, group study (probably having discussions about examinations) with friends, etc. These activities help students go around Kolb's learning cycle, because such informal study will help students to understand their formal learning activities better and continue their learning processes more smoothly. Meanwhile, informal learning activities could help to build up communities of practice (Section 2.1.4). Normally the module leader does not set or suggest what external resources students could refer to after class.

2.5 Summary

Learning to program is a challenging task, both for academics and students. The backgrounds and prior experience of the students will certainly vary, but the teachers must ensure, wherever possible, that there is an equal outcome for all. The introductory course needs to be technically sound and at a level that sustains motivation and engagement across the group, by having a clear understanding about students' learning strategies, including their learning styles and approaches. Course content needs to incorporate new content as it becomes available from academics who continue to provide up-to-date methods, keeping in mind that the changing life experience of students makes this task even more complex. Keeping abreast of the new research and literature as it becomes available, can provide a better learning experience for first-year undergraduate students and enable them to develop more effective learning strategies as they learn to program in this connected world.

To conclude, many people have been teaching programming for quite a long while. Some have published their findings and have identified some key factors, such as motivation, models of learning, learned helplessness, etc. that contain useful information and insights into how best to communicate complex information and concepts. Successful outcomes from the teaching-learning process are most likely to occur when the needs of the learner are considered and the learning experience is designed in such a way that it takes into account an individual's readiness, appropriate delivery systems that activate the learner's motivations, and still enabling an educational process that is designed to achieve requisite learning outcomes. In the next chapter, existing research methodologies will be introduced and evaluated, following up with the experimental design that was used in this study.

The teaching-learning process is a dynamic interaction between all of the variables that interact in the context of introductory programming modules. It is important, therefore, to take into account those factors that can be controlled to some extent (e.g. content, experiential learning designs, student inputs) to determine how these affect student approaches to learning and influence their motivations to learn and expected outcomes. This research study is predicated on the belief that the application of these principles will provide knowledge as to how students that were involved in this experiment modified their learning behaviour, if at all, and the adequacy of the course content and materials for meeting their desired educational outcomes, which were the ultimate aims of this project.

The mapping of this chapter to the research questions could be summarized in the following table:

Research Questions	Related Sections
A. Considering first year undergraduates: before being formally introduced to learning to program at university, do any differences exist, within specific sub–groups of students, between the attitudes to and beliefs about the task of learning to program?	Section 2.2.2 Issues associated with students
B. Whilst learning to program in this connected world, to what extent do different students: a. Gain or lose confidence and motivation? b. Modify their approaches to learning as they progress?	Section 2.1.1 Kolb – Experiential learning cycle
	Section 2.1.2 Laurillard - conversational framework
	Section 2.1.3 Marton & Säljö – Surface and deep learning
	Section 2.2.1 Motivation
C. How do students integrate their formal and informal learning activities? To what extent do students: a. Rely on formally provided materials and exercises? b. Develop individual approaches to learning which integrate real world and online activities and materials?	Section 2.1.4 Lave & Wenger – Legitimate Peripheral Participation
	Section 2.1.5 Biggs – Levels about Thinking Teaching
	Section 2.2.3 Formal and Informal Learning
	Section 2.3.3 Computer-Aided Tools for Learning to Program

Table 2.9 Mapping the research questions to the relevant sections

3 Methodology

This chapter presents a summary of the research methodology, linking the research questions to the different approaches which were adopted. As has been discussed in earlier chapters, particularly Section 2.5, Biggs suggests that academics should focus “what the student does” to understand students’ behaviours and approaches to learning as deeply as possible.

The over-arching goal of this research is to gain insights into how unique personal attributes of student learners determine the approaches to learning or processes that students adopt, and adapt over time, to satisfy their motivational needs. An important aspect of their self-directed learning process is the extent to which they rely on recommended learning materials and exercises versus substituting their own real world and online learning sources.

Thus, the research questions listed in Chapter 1, which are the basis for this research, are designed to investigate what the student does. As follows:

A. Considering first year undergraduates: before being formally introduced to learning to program at university, do any differences exist, within specific sub–groups of students, between the attitudes to and beliefs about the task of learning to program?

B. Whilst learning to program in this connected world, to what extent do different students:
   a. Gain or lose confidence and motivation?
   b. Modify their approaches to learning as they progress?

C. How do students integrate their formal and informal learning activities? To what extent do students:
   a. Rely on formally provided materials and exercises?
   b. Develop individual approaches to learning which integrate real world and online activities and materials?

Aiming to answer those research questions, this study should gather as much data as possible to identify the potential findings.

A range of established approaches in research are used in order to study student behaviours and teaching interventions, to enable academics to gather this type of data. Quantitative methods are widely used to gather base data, whilst qualitative approaches enable the researcher to categorise subjective factors such as thoughts, beliefs and attitudes.

Since this study is aiming to understand “what the student does” while learning to program, gathering quantitative data alone would not be sufficient. Quantitative data can provide suggestions about differing attitudes and behaviours, but qualitative data allows us to gain insights and to back up or further illustrate suggestions which have been made from a reading and analysis of the quantitative data.

Qualitative evidence will be collected as much as possible to triangulate findings from the quantitative evidence. This mixed methods approach has the advantage of anticipating and remedying the potential gaps and risks in our data collection.

This chapter now explains the range of research methods that were adopted for this study and the rationale for assembling each stage of the process.

Section 3.1 describes the respondent cohort for this research. Section 3.2 introduce the research framework and gives a brief overview on quantitative and qualitative methods. It then expands each stage of the framework, justifying the choice of methodology for each stage. Section 3.3 explains how it was proposed to analyse the collected data.

3.1 General Design

Overall, the experiments were scheduled from late September 2012 to January 2013, covering the introductory programming module. The main design principles were:

The procedure of the experiment could be summarized as an experimental framework as shown in Figure 3.1:

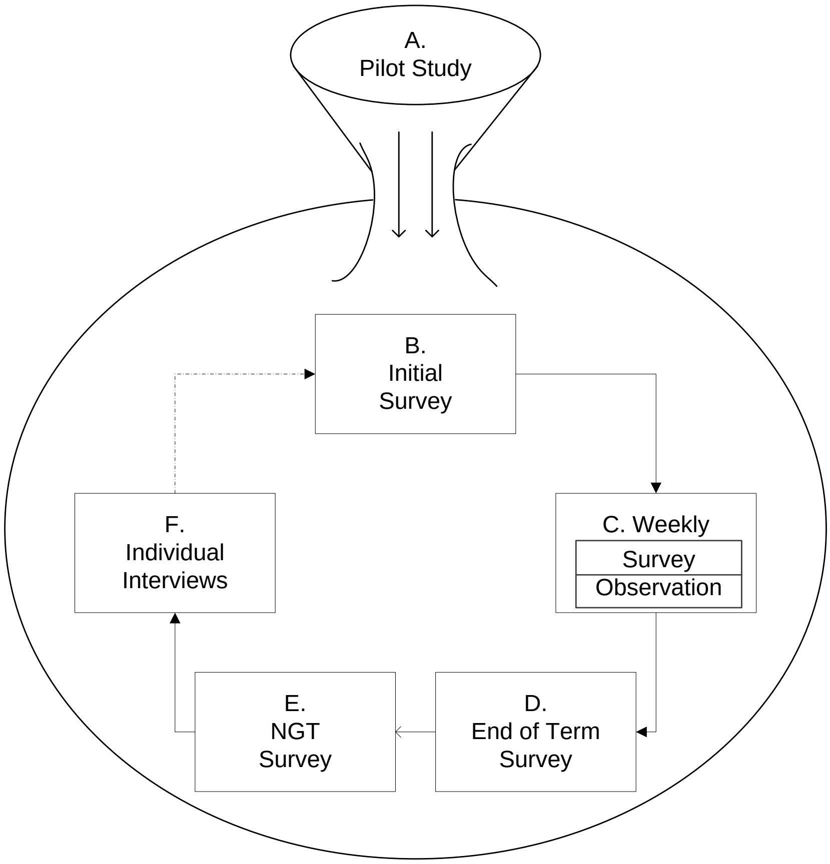

Figure 3.1 Experiment Framework

- The pilot study (A) is designed to validate the survey method and questions further after an initial peer review.
- The initial survey (B) gathers as many responses as possible from a wide cohort of students on introductory programming courses, because it would help the investigator to obtain comprehensive insights from a larger population of students.
- After data collection from these early surveys, the investigator would analyse the collected evidence to design in-depth questions for subsequent investigations. This would involve Weekly Surveys and Observations in parallel (C) and an End of Term Survey (D).

- During the later experimental period the objective would be to drill down into the findings and if possible gather triangulation data. The process would use a Nominal Group Technique (NGT) approach (E) and individual interviews (F). The investigator would recruit a small number of students to participate in individual or group meetings, aimed at gathering detailed insights and perspectives on the basis of findings from previous responses.
- It could be possible to repeat the entire framework process to gather further data and gain further insights, however in this case, time constraints associated with the research would not enable that to be achieved

The rest of this chapter explores the design decisions and proposed purpose of each step of the framework in greater depth. A detailed account of the implementation and results analysis can be found in the next chapter.

3.1.1 Pilot Study

A pilot study for the purposes of obtaining base-line information was designed (Appendix A). It investigated what factors were important to undergraduates when introduced to programming at universities. Although questions were peer evaluated to validate the instruments before the survey was conducted, the pilot study also served to monitor the effectiveness of survey questions and to shape the future formal surveys.

This study would be run at a time that students would be busy with exams. As a result, an online semi-structured questionnaire with mixed types of questions (open-ended and closed-ended questions) was a preferred choice because respondents would be able to choose their own free time to provide responses.

In terms of the questionnaire mode, a semi-structured questionnaire was preferred because it has a mixture of structured and open questionnaire features. Some questions would offer a range of fixed and standard responses while others would not.

Categorical questions using a Likert scale were used to gain insights into the distribution of factors such as motivation, prior experience or confidence. Open questions were designed to elicit insight into aspects such as personal attitudes and behaviours. In this latter case, an open ended question allowed the student voice to be heard. The objective was to combine the comparative strengths of structured and open questionnaires.

Thus, respondents would be allowed to offer various types of opinions. For example,

- Students would only be asked about why they missed lectures if they said they had been absent for some sessions.
- Students would not be asked their detailed thoughts about class size if they didn't believe class size affected their learning.

In terms of the question type, for this study, mixed question type would help respondents to describe their perspectives in a proper way and thus provide the researcher with the insights relevant to the research questions.

Open-ended questions related to questions in which the answers of respondents would be solely up to them and their responses could be in a word, phrase or a sentence, especially when looking for robust (with a certain depth and breadth) information, the open-ended question is more suitable. In detail,

- For questions that the potential responses may contain various of aspects, e.g. what was the most useful feedback to help learn and what was the least.
- For questions that the investigators had no idea what options to offer, e.g. what students would do immediately after they received their feedback.
- For questions encourage new ideas, e.g. ask for suggestions about additional ways in which students might be provided with help.

But the use of close-ended questions would be helpful when seeking categorical information that could be summarised and classified as to frequencies. For instance:

- Demographic questions, e.g. students' age, origin, gender, etc.
- For rating purposes, e.g. students' prior programming skill level before coming to the university; students' thoughts about how practical labs and coursework affect their learning.
- Aiming to do result analyse effectively, options would be provided for students to select, e.g. students' favourite online learning platform; their preferred way in terms of coping with practical exercise.

Details relating to the implementation of the pilot study can be found in Chapter 4.

3.1.2 Initial Survey

This is the start point of the formal series of experiments, if it is the first time to run this framework. This survey would be conducted before students formally start their learning of introductory programming module. The investigation results were then used to inform the design of the subsequent weekly surveys.

This survey focuses on new undergraduates' background related to learning to program, including looking for any potential relationship between their broad demographic (e.g. age, gender, the course they'll take, etc.) and initial self-rated motivation.

A paper-form structured questionnaire with closed-ended questions would be used to gather baseline data in the welcome meeting for all the students who would be taught introductory programming. It was designed to allow respondents to provide their responses in a short time.

The respondent cohort in this case extended to all full-time undergraduates who were going to learn to program in their first year, in order to ensure a good volume of responses. Subsequent research instruments would only elicit responses from students specifically studying computing specific subjects (computer science; software engineering; IT in Organisations).

Due to the large expected amount of respondents, closed-ended questions were preferred in this study. In detail,

- Demographic factors, e.g. gender, age, origin, enrolled courses, etc.
- Four-point Likert scale to identify students' attitudes, e.g. current beliefs and motivation.

The survey was designed to elicit responses that might be able to provide

- a general understanding of the cohorts starting points
- offer basic information with which to address research question A
- help design questions for the following surveys.

As shown in the diagram, it might be possible to repeat the research framework cycle. If the cycle of the proposed research framework were to be run more than once, responses elicited from the final individual interviews (F), might be taken into account in refining the questions in the next round of initial survey.

Detailed design for initial survey can be seen in Chapter 5.

3.1.3 Weekly Survey and Observation

3.1.3.1 Weekly Survey

The weekly survey was designed to build a richer picture that than created by the pilot and initial surveys and gather insights into the dynamics of learning to programme. A short weekly survey was run throughout the introductory programming module. The three question survey was given to students as they completed their programming lab.

The weekly online questionnaire traces new undergraduates' learning progress and attitudes for each week whilst learning to program. Similar to the initial survey, the weekly questionnaire should mainly consist of closed-ended questions. The reason is that, potential respondents were supposed to provide their thoughts at the end of the lab. Such questionnaire would allow them to respond in a short time and could help to enhance their willing to offer responses. The questions should attempt to gather participants' believes about the lab and themselves. The responses should be able to answer the first two research questions (A and B) in a surface level and provide "threads" for further detailed surveys.

3.1.3.2 Weekly Observation

During the programming lab, students were observed by the researcher and notes made relating to their behaviours and attitudes. The observation allows the investigator learn respondents' behaviours through visual contact, including how students code and get help if struggling.

The objective of this part of the design was to use observation method to collect various data about students' behaviours, learning approaches, along with the overall situation including the special atmosphere and phenomenon. Respondents' behavioural performance would be observed under natural conditions, so the obtained result could be more realistic.

Detailed design for weekly survey can be seen in Chapter 6.

3.1.4 End of Term Survey

At the end of the introductory programming module, a further survey was conducted to analyse more in-depth findings aiming to identify how the Web affects first year undergraduate students who were learning to program. It allowed the students to review their past programming learning process and provide insights in general to their approaches to learning, and more specifically as to whether the Web was helpful for their learning, and if so in what way.

An Online structured questionnaire with both open-ended and closed ended questions would be used to gather baseline data, because online structured questionnaire allowed respondents to give quick answers to the questions in their own free time. Mixed type of question could allow respondents to offer comprehensive opinions. Closed-ended questions allowed participants to report on their behaviours within a pre-chosen range of options, e.g. information related to participants' favourite online learning platform; why they use the Web to learn; etc. the closed format would facilitate broad groupings within the data providing a 'big picture' . Open-ended questions could encourage and reveal new ideas, e.g. participants may have suggestions on how learning could be enhanced by adopting the Web. The responses could assist shape the following in-depth studies and provide surface answers for research question C.

Detailed design for end of term survey could be seen in Chapter 7.

3.1.5 NGT Survey

The NGT surveys were proposed to enable the researcher to drill down into the research question. This decision was based on the assumption that, through analysis of the responses from previous surveys, the investigator will have obtained a certain number of data from a series of previous investigations and some preliminary insights into the research questions will have been gained.

In order to get more in-depth evidence and students' ideas and understand what the student does more accurately, it was necessary to summarize any uncertain points in previous findings and select a sample of students with whom to conduct deeper investigations. Since it would be at the end of the semester when students begin to prepare for the final exam, we did not expect a large number of students to choose to cooperate and participate in the survey.

Thus, it was preferred to conduct focus group discussion to gather relevant data. Nominal Group Technique and Delphi Technique are two possible approaches to manage/orchestrate such discussion. Table 3.1 compares and contrasts the different techniques in order to design experiments to gather data required.

CATEGORY	NGT	Delphi/ Modified Delphi Technique
Process Characteristics	Generate opinions individually trough reflection, Use cyclic process to express conception, conduct a series of discussions to clarify the evaluation of individual opinions, integrate the results through rigorous voting process	Construct a series of questionnaires reach a consensus through respondents' remote feedback, Integrate the results using the statistical techniques
Member number	host：1-2 NGT member：around 10，not exceeding 20	Decision maker：less than10 interviewee：about 15 (MDT basically has no limitation)
Statistical analysis techniques	Not necessary.	Compulsory
Originality quantity	Many (independent thinking）	Many (isolated thinking)
Originality quality	High	High
Required time	Moderate (depending on question quantity 1.5h-8h)	Long (about a week)
Conclusion accuracy	High accuracy	Moderate accuracy

Table 3.1 Nominal Group Technique Compared with Delphi/Modified Delphi Technique

To summarize, given the likelihood of a small number of participants and the high uncertainty of the answers in this investigation, the Nominal Group Technique was recommended to help with gaining understanding of what the students. The Delphi and Modified Techniques can be used to enable all the participants to achieve a consensus opinion, but needs either an extended period or a number of iterations to gain a substantial amount of Data. By contract, the processes of NGT might not only assist in identifying and evaluating the participants' assertions, but also enable the researcher to understand them more clearly.

Moreover, if there were still some unclear thoughts expressed in the responses during the NGT, the investigator would have the opportunity to ask for explanations during the subsequent individual interviews.

Detailed design for NGT survey can be seen in Chapter 8.

3.1.6 Individual Interviews

Finally, some unclear points to answer all the research questions might still exist, mainly in qualitative aspects. Thus, individual interviews with some students might reveal valuable extra data. Using interviews after the survey and group discussions were designed extend to the scope of the preceding research findings. It was designed to focus on understanding the in-depth perspectives/and understandings gained by students' whilst learning to program. Interviews would provide the opportunity for the researcher to address any unsorted points needed to gain insights into the data/evidence gathered using the proceeding research instruments.

The semi-structured interview would be much more suitable to probe unclear thoughts from previous interventions. The strength of an individual interview, rather than using a group interview, would be that individual interviews would allow interviewees to share actual perspectives which might have been obscured for example by a desire for privacy, or wishing to be seen as agreeing with other participants. Thus, semi-structured individual interviews, using mostly open-ended questions generated on the basis of responses from previous surveys, were preferred in this stage.

Detailed design for individual interviews could be seen in Chapter 9.

3.2 Respondent Selection

The respondents were selected from AY2012-13 first year undergraduates enrolled on an introductory programming module (COMP1004, details in Section 2.4) in the School of Electronics and Computer Science in the University of Southampton. There were 141 potential participants, combined with full-time students majoring in Computer Science (CS n=98), Software Engineering (SE n=17) and Information Technology in Organisations (ITO n=26).

3.3 Data Analysis

The data analysis in this study was synchronized with the entire investigation plan. That is to say, after the completion of each round of survey and data collection, the data analysis concerning this survey would be performed, and combined the results of the previous survey (if applicable). After the last round of interviews, the results obtained would be used to help answer the research questions. If a new trial would be needed the following year, the current results could help adjust any new questions from the initial survey, making the series of surveys in the following year more targeted and direct.

3.4 Summary

This experimental design for this study would call for data to be collected at pre-determined points throughout the course of the experiment, using a survey-feedback technique, beginning with a pilot study. The pilot study would consist of a small sample of students who would volunteer to take a version of the survey prior to it being administered to the experimental group. Some modifications would be made to the survey prior to administering the initial survey. The initial survey would be administered to the total population at the onset of the experiment. Quantitative analyses would be used to interpret the results. Once the course began, surveys would be taken at the end of each week and observations of the programming lab were conducted. A further survey would be conducted at the end of term administered to each student who completed the course.

This would mark the conclusion of the survey-feedback data collection process and all of the data were analysed using quantitative analysis statistical treatments. Upon completion of the teaching of the module and after all surveys had been administered, a Nominal Group Technique (NGT) would be used to supplement the survey data in an effort to provide additional insights about the learning experience from the viewpoint of a range of different participants.

The final data collection was planned in the form of personal interviews. The interview data would be analysed using a qualitative analysis technique called Thematic Analysis. The following chapters explain the data collection procedures and the analysis used to interpret the results.

The framework, which was designed specifically for this research study, would be in itself is generic, and could be applied to this study, or to other studies which would seek to identify insights into student learning processes. Due to the limited time, this research would only run one iteration of the framework. However, it would be possible to run through the framework repeatedly, enabling the approach to be refined, and a greater volume of data retrieved.

The mapping of this chapter to the research questions could be summarized in the following table:

Research Questions	**Related Experiments**
A. Considering first year undergraduates: before being formally introduced to learning to program at university, do any differences exist, within specific sub–groups of students, between the attitudes to and beliefs about the task of learning to program?	Initial Survey
B. Whilst learning to program in this connected world, to what extent do different students: a. Gain or lose confidence and motivation? b. Modify their approaches to learning as they progress?	Pilot Study, Weekly Survey and Observation, NGT Survey, End of Term Survey, Individual Interviews
C. How do students integrate their formal and informal learning activities? To what extent do students: a. Rely on formally provided materials and exercises? b. Develop individual approaches to learning which integrate real world and online activities and materials?	Pilot Study, End of Term Survey, NGT survey, Individual Interviews

Table 3.2 Mapping the research questions to the relevant experiments

4 Pilot Study

According to the previous analysis, a pilot survey is suggested to be conducted before the formal experiment begins. This chapter describes the reasons for the pilot study, how it is designed and implemented to obtain results from a preliminary survey. Use of a pilot study can improve the quality of the overall project. It can give advance warning about where the main research project might fail, whether the research protocols work, and whether proposed data collection methods or instruments can be expected to yield expected results. In this instance, the questionnaire was administered to the pilot subjects for their feedback and input into the utility of the items in the questionnaire. Feedback from participants was used to identify any ambiguities or clarifications that needed to be made regarding the questions. The pilot survey served as a protection against the possibility of conducting the entire study only to find out later that there were serious design flaws that jeopardise the results of a study that has already been completed but could have been avoided. Once completed, it is essentially impossible to re-engage survey participants in any meaningful follow up. Not only does that require a monumental effort to correct past errors, it also undermines the credibility of the entire project.

4.1 Reasons

Based on information in Section 3.7.2, a pilot survey in a smaller scale is needed before a final formal survey is directly conducted to a larger amount of students. The reasons are summarized as following:

- This pilot survey provides an opportunity to try questions and obtain students' perception and experience in the learning to program.
- The feedback from this activity can be used to shape the format of the student surveys and interviews in the proposed final stage of the study.
- It provides an opportunity to check whether the questions are being asked in an appropriate way, which could help to avoid possible problems with misunderstood or imprecisely worded questions in subsequent studies.
- The pilot stage also provides an opportunity for the researcher to develop and practice survey skills, including seeking ethical approval of the study, the design of questionnaire, timing issues, selection of respondents, result analysis, etc.

4.2 Design

This pilot survey was conducted using AY2011-12 first year undergraduate students who have finished the introductory programming module (COMP1004 Programming Principles) with the cooperation with the school of Electronics and Computer Science in the University of Southampton.

Since this study does not include any formal experiments, it is not necessary to confine the topic to what the students do while learning to program with the Web. The core research question for this survey could be expanded to investigate:

How do 1st year undergraduate students learn to program?

Questionnaire with both open-ended and closed-ended questions is the preferred tool due to the huge amount of the possible responses. However, this question cannot be directly addressed with the students in the questionnaire. On the one hand, the respondents may get confused about what to answer. Probably some of them would write a huge paragraph about a question, while, on the other hand, the data they provide could be difficult to analyse. Thus, questions should be broken down into several sub-questions. For the purpose of getting answers to such questions, the following question is offered:

What factors are important to undergraduate students when they first learn to program at universities?

This question focuses on the “factors” which could affect students’ learning, which seems to be intended to help academics understand what students do when they learn to program and how and what students feel and believe as they learn to program. Biggs suggests that it is most useful to think about what students do and when they learn. This survey looks at behaviours and attitudes in an attempt to gather evidence for better understanding how students learn.

Nevertheless, the data gathered from this question might be extremely arduous to analyse, because without specific directions, different respondents would answer in various areas. In such case, no details but only rough data would be collected. Therefore, the question requires further refinement.

Also, aiming to ensure that this pilot survey could cover all the topics of interest, this question is initially defined in three parts based on the standard teaching processes in the UK universities – lectures, assessments and feedback. Plus, students’ personal information should be considered due to student diversity (Section 4.2.2.1). In addition, the last part of questionnaire, named “others”, is necessary, since there might be some questions that are difficult to be categorised into any of the previous four parts.

4.2.1 What and Why to Ask

Having reviewed previous work relevant to this area (Jenkins 2001b; Jenkins 2001a) and books about survey technologies (Fowler 1995; Buckingham & Saunders 2004), the structure of this pilot survey is described as follows:

4.2.1.1 Personal Information

This part is designed for the purpose of addressing the issue of diversity and the final classification statistic (Table 4.1).

AREA	TOPIC	REASON
Basic Information	• Gender, age, origin • English ability, final grade	In the future statistics, the information are important since they are the basis of classification. As for the pre-skills, probably there is some potential relationship between some subjects and programming (e.g. math VS.programming).
Background	• Experienced? (Section 2.4.3) • Pre-skills	

Table 4.1 Questionnaire Structure for Personal Information

4.2.1.2 All about Our Teaching

In this section, we are trying to find out how well-structured teaching/lectures contribute to students’ mastery of programming at first year undergraduate level, which is still a really “big” question to answer. Students may respond simply “good lectures make me grasp the theories, which is essential for the practical work" or things like that. These opinions are too obvious and they are quite useless for research.

In addition, it is also interesting to know whether students have any special expectations for their textbook, because there are students who prefer textbooks with sufficient practical work, rather than pure theory. This question could help us to understand whether a suitable textbook contributes to students' learning to program.

Thus, this issue could be divided into elements shown in Table 4.2.

AREA	TOPIC	REASON
Lectures	• Attendance • Live programming demonstration	Helping to check the engagement and the importance of experiential learning (Section 2.1.1). Live programming demonstration might enhance student class preparation such as using YouTube (Carlisle & Educator 2010) and revisions.
Concepts	Understanding of basic concepts	This could help us to see whether a good start is important for learning to program? Probably students will suffer from learned helplessness in here (Section 2.2.1.2).
	Any challenges?	Investigate the most challenging theories for students and the methods they used to cope with them. The issue of deep and surface learning in here (Section 2.1.3).
Classroom	• Learning materials (Inc. textbooks) • Class-size	Investigate whether external factors have strong influence whilst students are learning to program.

Table 4.2 Questionnaire Structure for All about Our Teaching

4.2.1.3 Practical Work

We also care about what students think about the practical work they were asked to do (Table 4.3).

AREA	TOPIC	REASON
Evaluation	• Difficult? • Useful? • Amount?	This will help academics to have a general view about the quality of the assessments.
Effect	• Positive? • Negative?	Investigates whether assessments will promote or demotivate students to learn to program.
Style	• Timing • Mode (solo or group)	Checks students' initiative (Section 2.1.1) and how the mode of practical work will affect students' learning, especially for novices.

Table 4.3 Questionnaire Structure for Practical Work

Through these questions, we could have a better understanding about the required practical work, determine if there is some relationship between compulsory practical work and learning to program in universities and make some changes for the future semesters, if necessary. In terms of the mode, although much has been examined that uses programming materials as an effective method (Braught et al. 2010; Radermacher 2011; Radermacher & Rummelt 2012), there is no obvious evidence showing that students do not prefer individual and group projects.

4.2.1.4 Feedback

This section is intended to gather students' thinking about the feedback they have received, generated or used (Table 4.4).

AREA	TOPIC	REASON
Evaluation	• Rate • Response	Investigate the effectiveness of our feedback and the students' behaviour towards it.
Opinion	• Advantage • Disadvantage	Gather qualitative data what students think about the feedback.

Table 4.4 Questionnaire Structure for Feedback

4.2.1.5 Others

All the other questions not belonging to any of the four previous categories will be asked in this section (Table 4.5).

AREA	TOPIC	REASON
Getting Help	• How • Why	Investigate students' ability for solving problems on their own and if they have different backgrounds, use different ways to help them (Section 2.1).
Self-Study	• Frequency • Timing	Investigate the relationship between how much time they spend for learning to program (Section 2.2.2) and their final grade.
Summary	• Gateway language • Interest	Explore whether the programming language they used is believed to be good and ask their vision for the future.

Table 4.5 Questionnaire Structure for Others

Questions about "getting help" are not specific to practical work. It is expected that we could get to know that, how students get help and solve their problems when they get stuck in anytime while learning to program, including in the labs, preparing examinations, etc. According to previous surveys, without proper help, students may lose their self-confidence and get entangled in "learned helplessness" (Section 2.2.1.2). By analysing the data gathered from this question after the survey, maybe we could propose some innovations for providing students the needed help.

4.2.2 How to Ask

Generally, there are three response types gathered from this pilot survey which are described as follows:

4.2.2.1 Single/Multi Response
Students will be asked to select single or multi option(s) from lists. This kind of question is normally asked when the potential answers are specific, such as gender and origin. Sometimes, the answers might generally be found in particular aspects, such as the reasons why students missed the first few lectures. However, the option “other” will always be added in this type of question following by an open-ended question like “please specify”, just in case that some students have unique responses.

4.2.2.2 Quantitative Rating
A Four-point Likert Scale was adopted for this kind of question. The reason why to choose a four-point scale is that, the respondent is forced to select a distinct negative or positive response. Evidence suggests that a Likert scale of an odd number of choices more frequently elicits a mid-point response.

4.2.2.3 Qualitative Text
Open-ended questions will be asked to gather qualitative data (text). All of the questions with uncertain responses are suitable for this type.

This text can be more difficult and time-consuming to analyse, but can provide richer insights into the reasoning that underlies the quantitative responses. It also provides the opportunity to use illustrative quotations from the respondents.

The final questionnaire used in this phase of the study can be found in Appendix A.

4.3 Implementation

4.3.1 Respondents
The respondents were recruited from first year undergraduate Computer Science and Software Engineering students at the University of Southampton. The survey was advertised via emails, Facebook group and word of mouth to attract responses from a wide range of disciplines and cohorts. There was no limit to the number of participants that could take part and all participants took part of their own free will.

4.3.2 Procedure
This questionnaire was hosted at the University’s iSurvey website[1]. As a result, the consent information and the participation information sheet (PIS) were given on the first page of the questionnaire before any consent was given by the participant. The submitted consent form describes what would be displayed to the participant on the iSurvey webpage before the survey begins.

The questionnaire on the iSurvey website was able to accept participants from Wednesday 15th May 2012. The questionnaire remained open because students may have some time to respond in the coming summer vacation.

4.3.3 Results
The ethics approval for this study took longer than had been expected, which unfortunately resulted in poor timing for the recruitment of respondents. Due to the bad timing, until the end of June,

[1] https://www.isurvey.soton.ac.uk

there are eight complete responses (all the sections were completed) collected from about 125 potential respondents. Fortunately, this is a pilot survey and the responses have covered all the mainstreaming students' opinions. The results from section to section are listed as following:

4.3.3.1 Personal Information

There were six male and two female respondents complete the survey, which contained three UK and five international students. All of the eight respondents were in the ages between 18 and 22. They all started the course on time and half of them received over 70% as their final grade, although their pre-skills were different.

4.3.3.2 All about Our Teaching

Six out of the eight respondents attended all the lectures and all the respondents believed that this attendance was higher than other modules. Half of the respondents stated that programming is a little challenging for them (challenged areas are different) but they still successfully followed the lectures and planned to cope with that.

Most of them liked live programming demonstration but preferred it in ways that were different. In terms of learning materials, they held a positive attitude towards what have been provided to them, including the textbook.

4.3.3.3 Practical Work

Basically, students were satisfied with the practical work they were supposed to finish.

4.3.3.4 Feedback

Similar to the practical work, students held a positive attitude towards the feedback they received.

4.3.3.5 Others

Some of the students required extra help while learning to program and they preferred getting help online. The amount of working hours were varied weekly, but most of the students would not start to practice until they had to. In addition, they considered Java to be a good gateway programming language and they would not say "no" to job containing programming in the future.

In this section, the data gathered from the pilot survey will be discussed with previous literature review (Section 2), followed by some critical evaluation about the pilot study.

4.4 Discussion

There was a low response rate for the pilot study, only 6.4% (8 out of 125). The main reason for that is the difficult timing of the survey, which was close to the final examination and most students did not choose to respond due to their limited time. Also, it may be that, students who did not do well in the module were possibly de-motivated for learning to program, and would not be likely to take the survey as well. Due to the limited responses and the small total number of respondents, it is not appropriate to use a formal analysis (e.g. basic statistics such as mean and standard deviation, t-test) for the data gathered from the pilot survey. Nonetheless, we can still gain some perspectives on students' perceptions of the relationship between their learning approaches and their final achievements, through classifying, analysing and summarizing their responses. Some questions from the questionnaire will be considered as individual classification criteria and used to investigate and compare students' attitudes and behaviours while learning to program.

4.4.1 Personal Information

4.4.1.1 Male VS. Female

There were two female students out of the eight respondents. Actually, in the entire respondent cohort, there were about 10% female students. Both respondents got over 60% in their final assessments. They have similar background and preferences of learning to program as the other male respondents. Further insight could be gained through their interviews in the future.

4.4.1.2 Age

The UK respondents were under 20 years old, whilst international students were over 20. There was no evidence from their responses that age differences were significant.

4.4.1.3 Pre-Experience

There was no significance associated with the performance of previously experienced students. However, novice respondents still found it difficult to understand the first few lectures, although some of them finally understood the basic concepts by themselves. More data should be gathered via in-depth research in the future.

4.4.1.4 Origin and English Ability

There are large numbers of EU and international students that study in the UK universities. It is understood that their English ability may influence their learning, especially learning to program. From this pilot study, there were five international students and most of them graded their English ability as medium. They had great attendance, spent reasonable time on learning and demonstrated a wide range of different learning methods and approaches (e.g. using Internet, asking friends) to cope with challenges they met. Although most of them only received 50%-59% for final assessment, they remained motivated and happy to consider programming as their future career.

It is believed that interviews would be useful in final study to examine these effects in greater detail.

4.4.1.5 Final Grade

No obvious differences of learning processes existed among students with different grades. Students who got over 70% may have similar learning exposure as the ones who got 50%-59%.

This may be due to the paucity of the data. It could be that students who responded at this awkward time were motivated, and had a positive learning experience. The future study needs to be carefully timed in order to gather greater volumes of data to provide useful information.

It may also be helpful to conduct follow up interviews with these students.

4.4.1.6 Pre-Skills

Although Southampton requires that all students in Computer Science and Software Engineering have good qualifications in mathematics, the self-ratings of prior mathematical experience varied.

People believed that good math ability is helpful for learning to program. However, based on the responses, there was no evidence to show that a students’ self-assessed prior good mathematical ability was a necessary pre-condition for a good programming experience.

4.4.2 All about Our Teaching

4.4.2.1 Attendance

It is positive to see that there was great attendance at introductory programming lectures. Probably current lectures in the University of Southampton have successfully engaged students. It may however be that those who did not attend, also chose not to participate.

4.4.2.2 First Few Lectures

Most students claim that they had a good understanding of the first few lectures. Only one female international student with very good English ability said that she could not understand anything. They were totally novice respondents, but fortunately, they finally got 60%-69% after doing really hard work. Thus, improvements were made from the first few lectures.

It is worth mentioning that one international student responded as follows:

I found it really hard in the beginning but after some time when I get used to it. It becomes much easier to learn new stuff. I found looking at tutorial on YouTube very useful since I'm International student going to lecture really wasn't very helpful (just in the beginning).

Due to the low response rate, it cannot be assumed that this comment is a unique case. It could be common among our students. Certain innovations could be created to deal with this.

4.4.2.3 Basic Concepts

Most students believed that they had a sound understanding of basic concepts and a good level of motivation for their future plans. Only one international novice student with very good English ability claims that he had a very poor understanding of the basic concepts. Although the respondent passed the final assessments (50%-59%), he still felt de-motivated since he just attempted to finish the practical work without any extra help from his friends and lecturers (reasons not provided). Fortunately, he had a reasonable amount of working hours per week. It could be assumed that the understanding of basic concepts will strongly influence the student's attitude toward future learning.

A larger sample will hopefully provide a richer insight into this aspect of the study. However, again, it is perhaps an aspect of learning which could be usefully explored in a more in-depth manner via interviews or focus group discussions.

4.4.2.4 Challenging Theories

It should be interesting to research as to when students begin to feel that learning to program is a challenging task, because it would be useful for academics to improve their teaching strategies in certain areas. In the pilot survey, students were asked to identify the three most challenging theories and provide their solutions. Excluding two random responses due to the lack of options for students who had no challenge, half of the remaining students claimed "Testing & Debugging" to be their nightmare. One surprising reason is that they thought it was really difficult to provide the testing script. Object-orient theories, including inheritance and polymorphism, are normally considered to be difficult to teach. The results show that students rank them as the second and third most challenging topics.

The common solutions for students to cope with challenges were: reading more, reviewing some examples and getting help online, which helped most of them to sort the challenges out. Only one student claimed that after checking Google, he still was hardly successful. What's worse is that those respondents just continued their study like nothing ever happened. They finally got 50%-59% and could probably learn better, if they were encouraged to get additional help.

According to expressed mental attitudes after they coped with those challenges, they typically felt very pleased and highly motivated, or at least happy with their progress. Thus, we can conclude that challenges are NOT disasters, depending on how we treat them. If teachers could teach a way to reach solutions, students will be highly motivated and learn much better.

4.4.2.5 Live Programming Demonstration

Most respondents who took live programming believed that demonstrations were an important method to learn to program, including in-class demonstration and online videos. In the future, academics could start paying attention to this teaching strategy (Section 2.1).

4.4.2.6 Learning Materials

Almost all the students hold a positive attitude in terms of the quality and quantity of printed/online notes, computing facilities, library materials and specialist resources. Although there were two respondents who expected better quality of the printed/online notes, there was no evidence showing, that it had negatively affected their learning.

In terms of the textbooks, less than half of the respondents read the textbook recommended. Most of them have no preferences. One potential reason for that was they care more about the notes lecturers provided. More in-depth research needs to be conducted about this.

4.4.2.7 Class-Size

Surprisingly, the respondents who claimed the class-size to be big all got over 70% in final scores. They said that "smaller class sizes would allow instructors to spend more time with students", "large class size made it harder to get help." So the main problem caused by the class-size was the difficulty of obtaining additional help when needed. Other students made no comment on this issue.

4.4.3 Practical Work

According to the results, respondents believed that the practical work (labs and coursework) was useful for their learning, and they had a reasonable amount of coursework. About half of the respondents considered the compulsory practical work to be difficult. Also, some of them blamed that there were too many labs. However, no relationship was found in which students who held a negative attitude towards compulsory work would become de-motivated and fail for their lack of learning.

In terms of the effect of the practical work, most of the respondents considered it to be positive (in different levels though, detailed analysis needs to be done when more responses are gathered in future work). The one who had problems with basic concepts (Section 4.4.2.3) was the only student holding a negative attitude towards this. This phenomenon might be usual for those students who got stuck from the beginning. The externally oriented control (Section 2.2.1.1) and the feeling of learned helplessness (Section 2.2.1.2) might be the responsible for their learning processes (Section 2.1.1), which means potential research topics for academics to improve this situation.

In addition, it is hard to tell which mode of practical work was preferred by students since among the eight respondents, three preferred working individually, three preferred pair programming and two preferred group project. However, it was obvious that they normally started their coursework early and the three novices do NOT prefer group projects. Only one experienced student, who got over 70% as the final grade, who started not long before deadline, was a competent programmer who could finish it on time.

4.4.4 Feedback

4.4.4.1 Evaluation

Generally, the results illustrate that the most popular feedback was from the discussion with academics and lecturers. Nonetheless, it is worth mentioning that, although students loved that kind of feedback, they still preferred NOT to turn to lecturers when extra help was needed. One student worried that the lectures would blame him for asking stupid questions, so he just preferred getting help online.

Other forms of feedback (e.g. from programming environment, personal reflection, discussion with friends/demonstrators, comments on practical work) were also welcome. Rather than "quantitative" marks, students did care more about the "qualitative" feedback for themselves. In contrast, comments and marks from other students' practical work did not appear to affect their learning, because most of them just consider such feedback not relevant/valuable. However, in the meantime, some students still valued the feedback from their classmates, since it might remind them to avoid some mistakes in the future.

In addition, most respondents agreed that the feedback was fair and useful, whilst controversy existed in terms of the timeliness. Almost half of the respondents believed that they did not receive their feedback on time, which might have delayed their learning progress.

This might be construed as learned helplessness.

4.4.4.2 Opinions

Students loved to receive comments about their shortages and recommendations for improvement. They felt happy and motivated when they were told how to program professionally, including programming habits and optimisations. They would act on the feedback in their following learning.

However, in the meantime, some students got really upset when "conflicting feedback" was provided due to "different markers have different opinions on what the mark scheme was." Those students also felt that the feedback was not fair enough, although they did not get de-motivated for this. Academics should pay particular attention about this since this is not an exceptional phenomenon.

4.4.5 Others

4.4.5.1 Getting Help

Much has been discussed in this section about how students get help. Respondents definitely searched the Internet or turned to their friends, whilst asking lecturers was almost ignored. The core reason was that they worried the lecturers would consider them to be stupid (Section 4.4.4.1). Thanks to the Internet, students with certain concerns nowadays have obtained a brilliant platform for help. They just enjoyed searching suggestions, such as sample code, without worrying how silly their questions might be, which also could help them to beat learned helplessness and rebuild their self-confidence and self-reliance for learning to program.

4.4.5.2 Self-Study

There was no evidence from this pilot survey that showed any relationship between the length of learning and their achievements. Students determined the quantity of hours suitable for themselves. It was very difficult to obtain an objective measure of the amount of time spent in self-study. However, the survey asked respondents for a retrospective measure which was likely to be self-assessed. It may be appropriate to consider whether future studies should be designed to obtain a more accurate measure of this factor.

Normally, we hope students could practice a new theory, once they have learned from lectures, because they just had a surface “sign” about the theory and it will be efficient to create a deep understanding of certain theories at that time. However, it is a pity that most of the respondents would not do more until the compulsory labs.

4.4.5.3 Summary

Having been asked to review their programming learning as whole, the respondents were apparently successfully engaged because none of them ever thought about leaving the course. As for their future careers, none of the respondents would reject programming relevant jobs. In terms of the gateway programming language, Java was well received by the respondents. In order to compare Java with other programming languages, such as Pascal, Python and C, interviews and surveys could be conducted in the future with other UK institutions.

Although the number of students who participated in the pilot study was small, they were somewhat representative of the total population in terms of the demographic representation – age, gender, regions represented, and prior programming experience. The feedback they provided was valuable. Like other forms of qualitative research, it provided an opportunity to explore some areas of inquiry in more depth. It provided valuable input into refinements to the questions that were subsequently included in the group surveys. The pilot study proved to be a very effective component of the overall design.

4.5 Summary

The pilot study was intended to be a part of the overall design from the beginning. It was believed then and has been confirmed now that input and feedback from students would be helpful in determining whether or not there were any unknown issues with any of the survey questions that were planned for future surveys. Although the sample was relatively small (eight respondents), their suggestions were valuable. As a result, some minor modifications to the questions were made and the results of that were reflected in the quality of the returns. There appeared to be no issues that anyone had with the questions or the responses and the completed survey questionnaires were useable. The pilot study was a fail-safe mechanism that performed exactly as planned.

According to the responses, some students withdraw in the middle of this survey because this questionnaire was too long for them to complete. It is important to optimize the size of each of the subsequent formal survey to maximize the respond rate. Meanwhile, it is essential to ensure that no potential factors related to learning to program are missing. Thus, the required information could be distributed into different stages of the following formal experiments. In details:

1. Personal Information. Information of age, level of prior programming experiences/skills and origin/English skill could be asked in the following initial survey, NGT survey and individual interviews, aiming to find out the potential relationship between these factors and learning. Final grade is a private issue, so in the following formal surveys, this question should be ignored. Alternatively, it should be useful to ask students' expected grade in each week of their study in the weekly survey.

2. All about Our Teaching. Information of attendance, first few lectures, basic concepts and challenging theories, as well as opinions about live programming demonstration, learning materials and class-size could be observed in weekly survey and proofed in end of term survey, NGT survey and individual interviews, aiming to obtain the behaviours in the first place. Students may forget their detailed feeling at the end of the module.

3. Practical Work. All the relevant issues could be observed in weekly survey and then proofed in final NGT survey and individual interviews.

4. Feedback. All the relevant issues could be observed in weekly survey and proofed in NGT survey and individual interviews.

5. Others. Issues asked in this session (how do students get help and do self-study) could be observed in weekly survey and proofed in end of terms survey, NGT survey and individual interviews. In details, it is possible to observe how student get help when needed and how they do self-study in class. Detailed perspectives could be investigated in the NGT survey and individual interviews.

5 Initial Survey

The previous chapter presented the results of a pilot study that was done to test the proposed research process as to its practical aspects regarding a) the effectiveness of design procedures, b) the questionnaire itself and c) the learning process from the student's perspective. Insights gained from the pilot programme have prompted changes in the actual study which are reflected in the following chapter.

The following chapter is about the design of a survey of the AY 2012-13 cohort group that was conducted in the form of questionnaire prior to when the teaching began (Appendix B). The intent was to

1) gain a general understanding of how much prior experience the new students had
2) to assess their initial attitudes towards learning to program, including
   factors associated with their motivations
   and their beliefs about how difficult they expected programming would be.

The outcome of this survey could provide evidence for Research Question A (Section 3.1.2 & 3.4):

A. Considering first year undergraduates: before being formally introduced to learning to program at university, do any differences exist, within specific sub–groups of students, between the attitudes to and beliefs about the task of learning to program?

5.1 Design

Based on the general design (Section 3.1), a paper-form questionnaire with closed questions was chosen as the preferred tool because it appeared to be the most efficient way to get the most information the easiest and quickest manner.

The main purpose for this survey was to gather some base data to provide evidence for the first research question:

A. Considering first year undergraduates: before being formally introduced to learning to program at university, do any differences exist, within specific sub–groups of students, between the attitudes to and beliefs about the task of learning to program?

The survey consisted of two broad parts, firstly questions relating to demographic factors were used to identify students' backgrounds upon entry into the programme. These background variables would subsequently be used to analyse whether they had any effect on the approaches to learning that were used to master course content. After the demographic questions attitudes towards learning to program were investigated with questions using a four-point Likert scale to measure their attitudinal range.

- **Demographics**

Initially, demographic factors – age, gender and origin were used to help to determine whether these backgrounds would affect students' learning behaviours. The assumptions were:

  - **Age**: students in different age groups may have different attitudes

- **Gender**: there are not very many female students learning programming related courses; why is this and is there any possibility that they have different attitudes and beliefs with male students?
- **Origin**: since this research focuses on what the students do in UK universities, students from different origins – UK, EU and international may have different learning experience and attitudes.

Apart from these, it is also necessary to ask what their courses are and how much programming experience they have. These might also affect students' learning behaviours.

- **Course**: different courses have different entry requirements (e.g. math skills, etc.). This might be one potential factor that affects students' attitudes.
- **Prior programming experience**: this might be a key issue that affect students, but detailed perspectives are important to understand our students better.

- **Attitude Measurements**

Four-point Likert scale, is preferred for measuring students' attitudes. The reason why five-point scale (Section 3.2.2 & Table 3.1) was not applied was in order to steer the respondents towards a clear preference for either positive or negative ratings for this study rather than neutral opinions. However, respondents remained free to ignore and not provide answers to those questions indicating that they had neither a positive nor a negative rating for those items.

To shorten this survey, the statements that were used for gathering levels of agreements could be generated as shown in the following table:

STATEMENTS	PURPOSE
I feel motivated to learn to program.	Check the level of initial motivation.
I think learning to program will be easy for me.	Check the amount of positive-attitude holders
I think I will have to work hard to learn to program.	Check beliefs.
I think learning to program will be difficult for me.	Check the amount of negative-attitude holders
I'm sure I will have mastered programming by the exam.	Check final expectations.

Table 5.1 Initial Survey: Likert Scale Evaluation Statements Designed to Measure Attitudes Showing the Purpose of Each Question

The final questionnaire used in this phase of the study can be found in Appendix B.

5.2 Preparation and Implementation

The paper-form questionnaires were given out in the welcome meeting to all the students (n=231) who were to be taught about programming in AY 2012-13 on Wednesday 17th September 2012 in the University of Southampton, whilst they were registering for the course. The consent information and the participation information sheet (PIS) are given on the first page of the questionnaire before any consent has been given by the participant.

5.3 Responses

The results of the questionnaire were collected from the paper form and later input into excel format. A number of steps were taken to ensure the validity of the data. The total number of records in the database is 169. The respond rate is 73.16%.

- Of the 169 students, 142 were male and 27 female, a ratio of 84%:16%.
- The distribution of their ages is illustrated in the following graph:

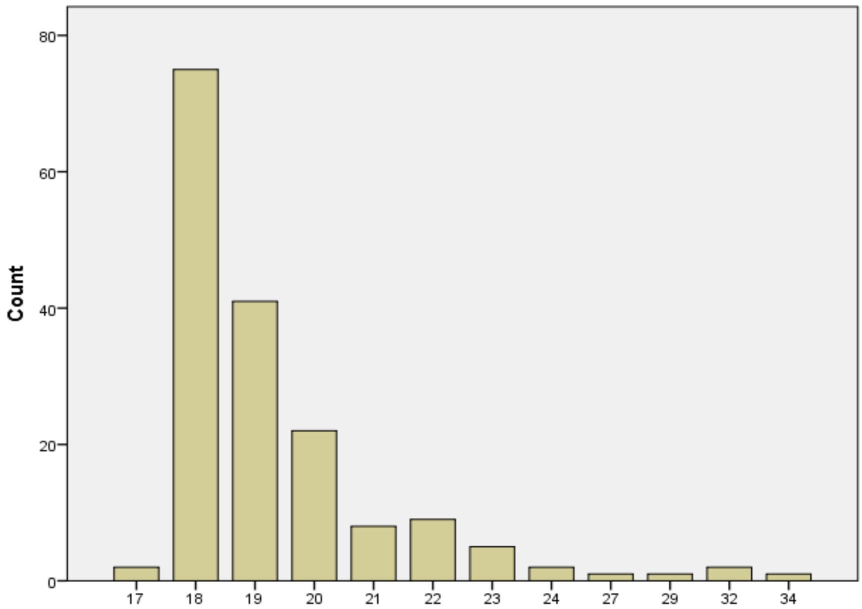

Figure 5.1 Participant Age Range

Ages range from 17 to 34 with the largest group being 18. This is a positively skewed distribution with a median age of 19.0 years.

- Their regions of origin are shown in the table, which states that just over half of the students are from the UK.

	Frequency	Percent
UK	94	55.6
EU	23	13.6
International	52	30.8
Total	169	100.0

Table 5.2 Participant Region of Origin

- The table below shows their distribution across the different courses they will be taking.

	Frequency	Percent
Computer Science	53	31.4
Software Engineering	7	4.1
Electrical Engineering	25	14.8
Electromechanical Engineering	12	7.1
Electronic Engineering	58	34.3
ITO	14	8.3
Total	169	100.0

Table 5.3 Respondent Course Enrolment

The majority of the students, 65%, were taking just two subjects: Computer Science or Electronic Engineering with the others spread across the other four subject areas.

- The students were asked about their prior experience of programming; the results are shown in the table.

	Frequency	Percent
None	40	23.7
A little	107	63.3
A lot	17	10.1
Extensive	5	3.0
Total	169	100.0

Table 5.4 Prior Programming Experience

A quarter of the students had no experience of computer programming at all with a large majority having some experience.

- The students were asked about their motivation to learn to program. Programming is a compulsory subject for all these students so that motivated to learn it may vary.

	Frequency	Percent
Strongly disagree	8	4.8
Disagree	7	4.2
Agree	52	31.1
Strongly Agree	100	59.9
Total	167	100.0

Table 5.5 Motivation to Learn Programming

The great majority indicated that they were motivated to learn programming (91%) but there were a minority that indicated that they did not want to learn to program.

- Students were asked how easy they thought learning to program would be and these are the results:

	Frequency	Percent
Strongly disagree	6	3.6
Disagree	62	36.7
Agree	90	53.3
Strongly Agree	11	6.5
Total	169	100.0

Table 5.6 Ease of Learning Programming

Although the majority of the students (60%) indicated that they thought learning to program would be easy. there was a substantial number who thought it would not.

- The students were asked about how hard they thought they would have to work to learn programming and these are the results:

	Frequency	Percent
Strongly disagree	8	4.7
Disagree	20	11.8
Agree	91	53.8
Strongly Agree	50	29.6
Total	169	100.0

Table 5.7 Expectations of Working Hard

The great majority (83%) believed that they would have to work hard.

- The students were asked if they were worried that learning to program would be difficult the following results these are the results:

	Frequency	Percent
Strongly disagree	22	13.0
Disagree	70	41.4
Agree	60	35.5
Strongly Agree	17	10.1
Total	169	100.0

Table 5.8 Worries about Learning to Program

The attitudes were fairly evenly split with just under half indicating that they were worried that programming would be difficult for them.

- The final question on the questionnaire related to the students' confidence about being able to be successful in the examination at the end of the course. The results were:

	Frequency	Percent
Strongly disagree	7	4.1
Disagree	21	12.4
Agree	105	62.1
Strongly Agree	36	21.3
Total	169	100.0

Table 5.9 Program Mastery by Final Exam

The great majority, over 80%, indicated that they were confident about being successful on the course.

5.4 Data Analysis

Each of the set of responses above shows the distribution of the responses to individual questions for the group of students as a whole. In what follows further analysis will be undertaken.

In the subgroup analysis how closely the student group are homogeneous was examined. We will be able to answer a number of questions like, for example, are the students on the different courses of the same age? Do they have the same gender breakdown? And how the key questions on attitude, motivation and other factors interact?

Afterwards, an examination of the key questions on attitude, motivation and other factors in relation to the grouping factors like gender, course, origin and prior programming experience is described. The statistical tools used was SPSS.

5.4.1 Subgroup Analysis

The basic analysis of the data given in Section 5.3 is for the group of students taken as a whole. It is expected that different subgroups of these students might have different characteristics.

One element of that analysis will look at responses by gender. The total number of responses from females (about 16%, see Section 5.3) is naturally small as is the female population in the total student cohort (37 female out of 231, approximately 16%). All the following gender-related analysis in this section is based on a same data set. Although, therefore, it is difficult to establish statistical significance, results may point to areas which would benefit from future examination in larger scale studies.

Further analysis aimed to examine how different subgroups of students evaluated their response to the statements related to their motivation and attitudes

1) I feel motivated to learn to program.
2) I think learning to program will be easy for me.
3) I think I will have to work hard to learn to program.
4) I am worried that learning to program will be difficult
5) I'm sure I will have mastered programming by the exam.

For each analysis the responses will be amalgamated into two categories so that a chi-square test can be done.

A Note about Grouping

In the two way analyses that follow in each of the three sections, we used chi-square tests to determine if the two variables are statistically independent.

In order to have a sufficient volume of data to perform this test in a meaningful manner the student responses given in the survey have been recoded into a smaller number of categories. 'Strongly Agree' and 'Agree' have become the single category 'Agree' with a similar amalgamation for 'Strongly Disagree' and 'Disagree'.

5.4.1.1 Age VS. Gender

This is a graph showing the age distributions for male and female students.

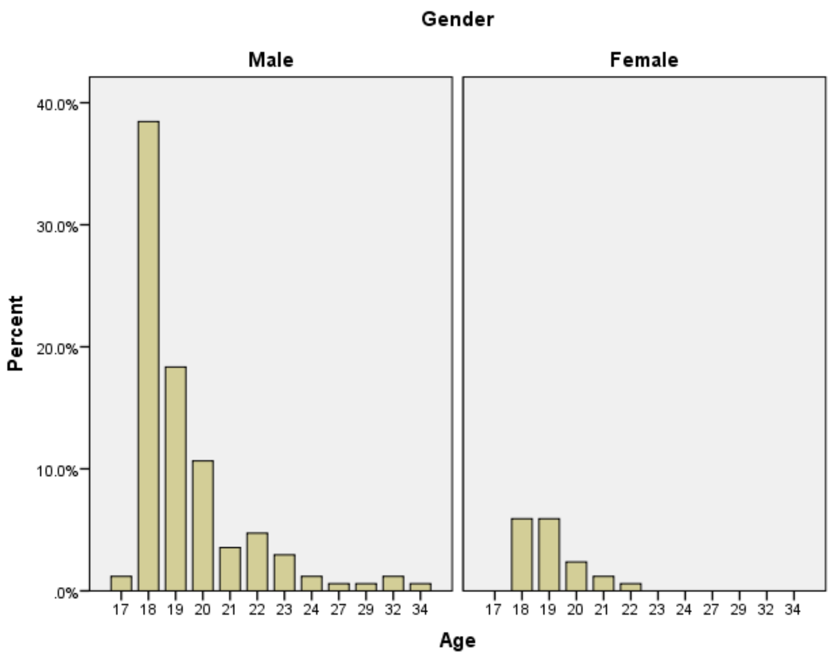

Figure 5.2 Age Distributions by Gender

The median ages for male and female students are both 19 years. Since the distribution of ages is skewed a non-parametric test (Mann-Whitney) is used to determine if there is a statistically significant difference in ages between the genders. The P value is 0.838 so we conclude there is no significant difference in ages between male and female students.

5.4.1.2 Age VS Origin

Here is a graph showing the age distributions for students from different regions:

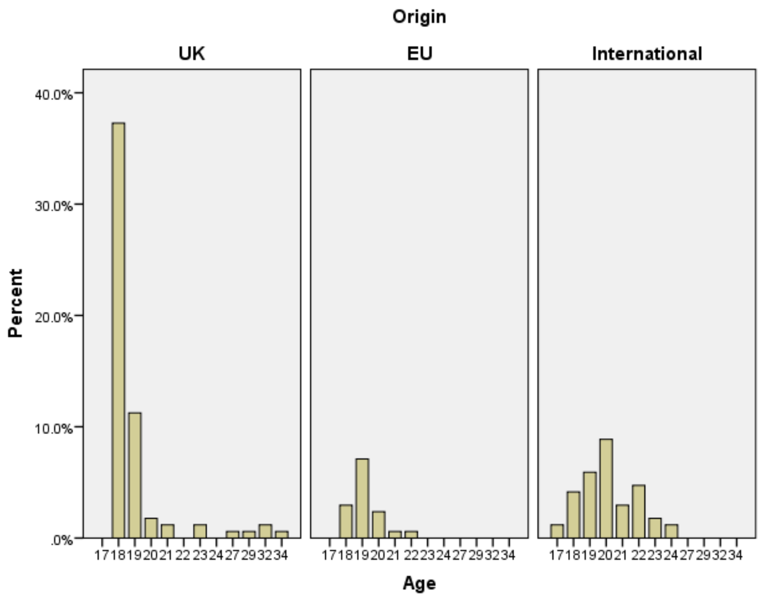

Figure 5.3 Age Distribution by Region

The median ages for UK, EU and International students are 18, 19 and 20 respectively. A test (Kruskal-Wallis) to determine if the differences are statistically significant shows that they are, with a P value of 0.000.

5.4.1.3 Age VS Courses

Below is a graph showing the age distributions for students on the different courses:

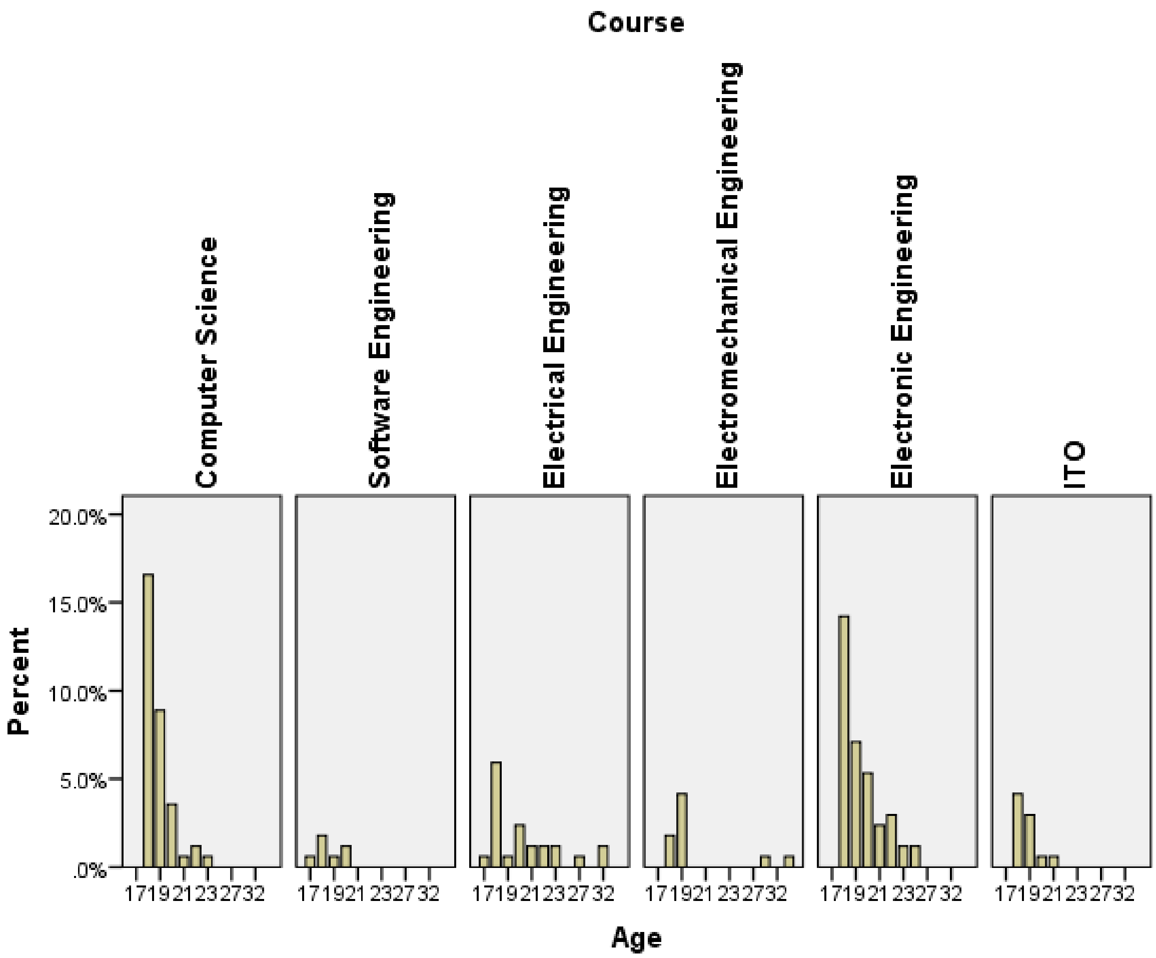

Figure 5.4 Age Distribution by Degree Enrolment

A test (Kruskal-Wallis) to determine if the differences are statistically significant shows that they are not; the P value is 0.190.

5.4.1.4 Age VS Prior Programming Experience

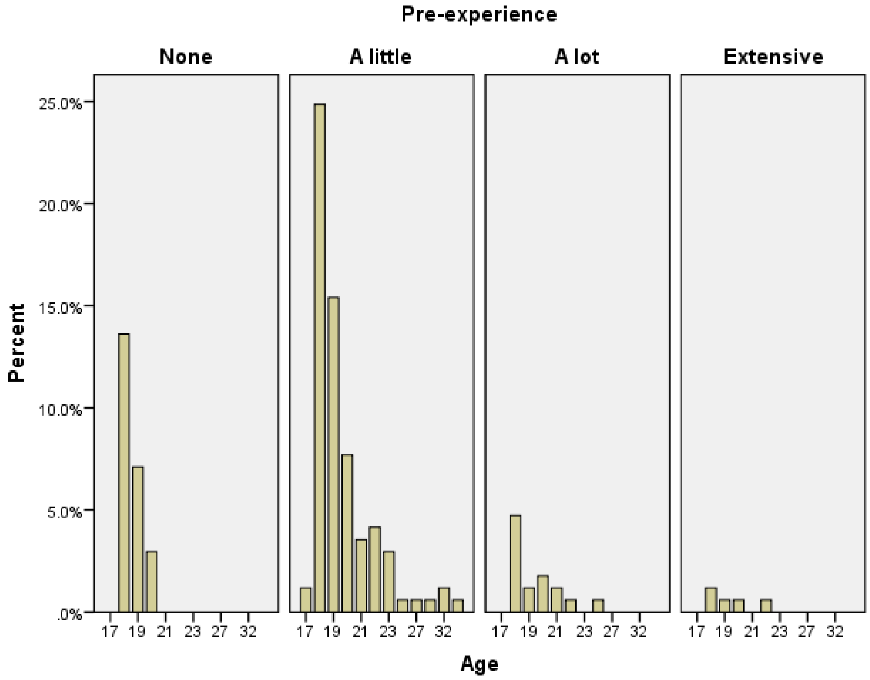

Figure 5.5 Age Distribution by Pre-Program Experience

A test (Kruskal-Wallis) to determine if the differences are statistically significant shows that they are not; the P value is 0.086.

5.4.1.5 Gender VS Origin

This variable and subsequent analysis of variables consists of categorical data that will be reported as tables.

			Origin			Total
			UK	EU	International	
Gender	Male	Count	81	19	42	142
		% within Gender	57.0%	13.4%	29.6%	100.0%
	Female	Count	13	4	10	27
		% within Gender	48.1%	14.8%	37.0%	100.0%
Total		Count	94	23	52	169
		% within Gender	55.6%	13.6%	30.8%	100.0%

Table 5.10 Gender Distribution by Region

The above table showed that there were 142 male respondents, including 81 (57.0%) from the UK, 19 (13.4%) from the EU and 42 (29.6%) international students. Meanwhile, among the 27 female respondents, there were 13 (48.1%) students from the UK, 4 (14.8%) and 10 (37.0%) students from EU and international countries respectively. A Chi-square test was then used to test the following hypothesis:

H_0. Gender has no effect on their origin (UK, EU, international).

H_1. Gender has some effects on their origin (UK, EU, international).

The calculated chi-square value is 0.767 with an associated P value of 0.68. It could be concluded that origin and gender are independent factors. Thus, hypothesis H_0 was accepted.

5.4.1.6 Gender VS Course

As mentioned earlier the data from courses have been amalgamated iinto computing vs engineering in order to have a reasonable population size to enable testing.

			Course		Total
			CS/SE/ITO	Engineering	
Gender	Male	Count	61	81	142
		% within Gender	43.0%	57.0%	100.0%
	Female	Count	13	14	27
		% within Gender	48.1%	51.9%	100.0%
Total		Count	74	95	169
		% within Gender	43.8%	56.2%	100.0%

Table 5.11 Gender Distribution by Degree Enrolment

The above table showed that there were 142 male respondents, including 61 (43.0%) students majored in Computer Science related courses (Computer Science, Software Engineering and Information Technology and Organization) and 81 (57.0%) majored in Electronic Engineering related courses (Electrical Engineering, Electromechanical Engineering and Electronic Engineering). Meanwhile, among the 27 female respondents, there were 13 (48.1%) students majored in Computer Science related coursed and 14 (51.9%) in Electronic Engineering related courses. A Chi-square test was then used to test the following hypothesis:

H_0. Gender has no effect on their enrolled course.

H_1. Gender has some effect on their enrolled courses.

The calculated chi-square value is 0.618 with an associated P value of 0.675. It could be concluded that course and gender are independent factors. Thus, hypothesis H_0 was accepted – gender has no effect on the choice of course.

5.4.1.7 Gender VS Prior Programming Experience

			Pre-experience		Total
			None/A little	A lot/Extensive	
Gender	Male	Count	127	15	142
		% within Gender	89.4%	10.6%	100.0%
	Female	Count	20	7	27
		% within Gender	74.1%	25.9%	100.0%
Total		Count	147	22	169
		% within Gender	87.0%	13.0%	100.0%

Table 5.12 Gender Distribution by Prior Programming Experience

The above table showed that there were 142 male respondents, including 127 (89.4%) students had little or no prior programming experience and 15 (10.6%) had a lot or extensive programming experience. Meanwhile, among the 27 female respondents, there were 20 (74.1%) novices and 7 (25.9%) pre-experienced ones. A Chi-square test was then used to test the following hypothesis:

H_0. Gender has no effect on their prior programming experience.

H_1. Gender has some effect on their prior programming experience.

The calculated chi-square value is 4.73 with an associated P value of 0.03. It could be concluded that prior experience and gender are not independent. Thus, hypothesis H_1 was accepted. In detail, a greater proportion of female students have prior programming experience compared to the male students.

5.4.1.8 Origin VS Course

			Course		Total
			CS/SE/ITO	Engineering	
Origin	UK	Count	47	47	94
		% within Origin	50.0%	50.0%	100.0%
	EU	Count	16	7	23
		% within Origin	69.6%	30.4%	100.0%
	International	Count	11	41	52
		% within Origin	21.2%	78.8%	100.0%
Total		Count	74	95	169
		% within Origin	43.8%	56.2%	100.0%

Table 5.13 Degree Enrolments by Region

The above table showed that there were 94 UK respondents, including 47 (50.0%) students enrolled in Computer Science related courses (Computer Science, Software Engineering and Information Technology and Organization) and 47 (50.5%) enrolled in Electronic Engineering related courses (Electrical Engineering, Electromechanical Engineering and Electronic Engineering). Meanwhile, among the 23 EU respondents, there were 16 (69.6%) Computer Science related students and 7 (30.4%) Electronic Engineering related ones, whilst 11 out of 52 international students (21.2%) were

Computer Science related and the rest 41 (78.8%) international students were Electronic Engineering related. A Chi-square test was then used to test the following hypothesis:

H_0. Origin has no effect on their chosen course.

H_1. Origin has some effects on their chosen course.

The calculated chi-square value is 18.5, with an associated P value of 0.00. It could be concluded that Origin and Course are not independent. Thus, hypothesis H_1 was accepted. In detail, although the proportion of UK students taking the either electronics or computing courses is equal, the proportions for EU and International students are very different; EU students favour computing courses and International students favour engineering courses.

5.4.1.9 Origin VS Prior Programming Experience

			Pre-experience		
			None/A little	A lot/Extensive	Total
Origin	UK	Count	83	11	94
		% within Origin	88.3%	11.7%	100.0%
	EU	Count	20	3	23
		% within Origin	87.0%	13.0%	100.0%
	International	Count	44	8	52
		% within Origin	84.6%	15.4%	100.0%
Total		Count	147	22	169
		% within Origin	87.0%	13.0%	100.0%

Table 5.14 Prior Programming Experience by Region

The above table showed that there were 94 UK respondents, including 83 (88.3%) students with little or no programming skill and 11 (11.7%) pre-experienced programmers. Meanwhile, among the 23 EU respondents, there were 20 (87.0%) novices and 3 (13.0%) pre-experienced ones, whilst 44 out of 52 international students (84.6%) were novices and the remaining eight international students (15.4%) possessed a lot of prior experience in programming. A Chi-square test was then used to test the following hypothesis:

H_0. Origin has no effect on prior programming experience.

H_1. Origin has some effects on prior programming experience.

The calculated chi-square value is 0.401 with an associated P value of 0.818. It could be concluded that Origin and Prior Programming Experience are independent factors. Thus, hypothesis H_0 was accepted - origin has no effect on the prior programming experience of students in the sample.

5.4.1.10 Course VS Prior Programming Experience

			Pre-experience		
			None/A little	A lot/Extensive	Total
Course	CS/SE/ITO	Count	62	12	74
		% within Course	83.8%	16.2%	100.0%
	Engineering	Count	85	10	95
		% within Course	89.5%	10.5%	100.0%
Total		Count	147	22	169
		% within Course	87.0%	13.0%	100.0%

Table 5.15 Prior Programming Experience by Degree Enrolment

The above table shows that there were 74 Computer Science related respondents, including 62 (83.8%) students with little or no programming experience and 12 (16.2%) pre-experienced programmers. Meanwhile, among the 95 Electronic Engineering related respondents, there were 85 (89.5%) novices and 10 (13.0%) with a prior programming experience. A Chi-square test was then used to test the following hypothesis:

H_0. Course has no relationship with their prior programming experience.

H_1. Course choice has some relationship with prior programming experience.

The calculated chi-squared value is 1.19 with an associated P value of 0.275. It could be concluded that Course and Prior Programming Experience are independent. Thus, hypothesis H_0 was accepted - course has no relationship with their prior programming experience.

5.4.2 Attitude and Motivation Analysis

This section will examine the relationship between the students' responses to the questions related to their motivation and attitudes.

The questions were whether they agreed or disagreed with these statements:

1) I feel motivated to learn to program.
2) I think learning to program will be easy for me.
3) I think I will have to work hard to learn to program.
4) I am worried that learning to program will be difficult
5) I'm sure I will have mastered programming by the exam.

For each analysis the responses will be amalgamated into two categories so that a chi-square test can be done.

5.4.2.1 "I feel motivated to learn to program." VS "I think learning to program will be easy for me."

			I think learning to program will be easy for me.		
			Disagree	Agree	Total
I feel motivated to learn to program	Disagree	Count	13	2	15
		% within I feel motivated to learn to program	86.7%	13.3%	100.0%
	Agree	Count	55	97	152
		% within I feel motivated to learn to program	36.2%	63.8%	100.0%
Total		Count	68	99	167
		% within I feel motivated to learn to program	40.7%	59.3%	100.0%

Table 5.16 Motivation to Learn V.S. Ease of Learning

The above table showed that there were 15 respondents who disagree or strongly disagree that "I feel motivated to learn to program", including 13 (83.8%) students who disagree or strongly disagree "I think learning to program will be easy for me" and 2 (13.3%) agree or strongly agree "I think learning to program will be easy for me". Meanwhile, among the 152 respondents who agree or strongly agree that "I feel motivated to learn to program", there were 55 (36.2%) students who disagree or strongly disagree "I think learning to program will be easy for me" and 97 (63.8%) agree or strongly agree "I think learning to program will be easy for me". A Chi-square test was then used to test the following hypothesis:

H_0. Respondents' agreement level of "I feel motivated to learn to program" has no effect on their agreement level of "I think learning to program will be easy for me".

H_1. Respondents' agreement level of "I feel motivated to learn to program" has some effects on their agreement level of "I think learning to program will be easy for me".

The calculated chi-square value is 14.0 with an associated P value of 0.00. It could be concluded that the two statements **"**I feel motivated to learn to program" and "I think learning to program will be easy for me" are not independent. Thus, hypothesis H_1 was accepted. In detail, students who are motivated to learn to program are more likely to think that learning to program will be easy than those who are not motivated.

5.4.2.2 "I feel motivated to learn to program." VS "I think I will have to work hard to learn to program."

			I think I will have to work hard to learn to program.		
			Disagree	Agree	Total
I feel motivated to learn to program	Disagree	Count	8	7	15
		% within I feel motivated to learn to program	53.3%	46.7%	100.0%
	Agree	Count	19	133	152
		% within I feel motivated to learn to program	12.5%	87.5%	100.0%
Total		Count	27	140	167
		% within I feel motivated to learn to program	16.2%	83.8%	100.0%

Table 5.17 Motivation to Learn V.S. Hard Work Requirement

The above table showed that there were 15 respondents who disagree or strongly disagree that "I feel motivated to learn to program", including 8 (53.3%) students who disagree or strongly disagree "I think I will have to work hard to learn to program" and 7 (46.7%) agree or strongly agree "I think I will have to work hard to learn to program". Meanwhile, among the 152 respondents who agree or strongly agree that "I feel motivated to learn to program", there were 19 (12.5%) students who disagree or strongly disagree "I think I will have to work hard to learn to program" and 133 (87.5%) agree or strongly agree "I think I will have to work hard to learn to program". A Chi-square test was then used to test the following hypothesis:

H_0. Respondents' agreement level of "I feel motivated to learn to program" has no effect on their agreement level of "I think I will have to work hard to learn to program".

H_1. Respondents' agreement level of "I feel motivated to learn to program" has some effects on their agreement level of "I think I will have to work hard to learn to program".

The calculated chi-square value is 16.8 with an associated P value of 0.00. It could be concluded that the two statements "I feel motivated to learn to program" and "I think I will have to work hard to learn to program" are not independent. Thus, hypothesis H_1 was accepted. In detail, students who are motivated to learn to program are more likely to think that learning to program will be easy than those who are not motivated.

5.4.2.3 "I feel motivated to learn to program." VS "I'm worried that learning to program will be difficult."

			I'm worried that programming will be difficult for me		
			Disagree	Agree	Total
I feel motivated to learn to program	Disagree	Count	9	6	15
		% within I feel motivated to learn to program	60.0%	40.0%	100.0%
	Agree	Count	81	71	152
		% within I feel motivated to learn to program	53.3%	46.7%	100.0%
Total		Count	90	77	167
		% within I feel motivated to learn to program	53.9%	46.1%	100.0%

Table 5.18 Motivation to Learn V.S. Difficulty Programming

The above table showed that there were 15 respondents who disagree or strongly disagree that "I feel motivated to learn to program", including 9 (60.0%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 6 (40.0%) agree or strongly agree "I'm worried that learning to program will be difficult". Meanwhile, among the 152 respondents who agree or strongly agree that "I feel motivated to learn to program", there were 81 (53.3%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 71 (46.7%) agree or strongly agree "I'm worried that learning to program will be difficult". A Chi-square test was then used to test the following hypothesis:

H_0. Respondents' agreement level of "I feel motivated to learn to program" has no effect on their agreement level of "I'm worried that learning to program will be difficult".

H_1. Respondents' agreement level of "I feel motivated to learn to program" has some effects on their agreement level of "I'm worried that learning to program will be difficult".

The calculated chi-square value is 0.247 with an associated P value of 0.619. It could be concluded that the two statements "I feel motivated to learn to program" and "I'm worried that learning to program will be difficult" are independent. Thus, hypothesis H_0 was accepted. In detail, being motivated to learn to program is not associated with thinking that learning to program will be easy.

5.4.2.4 "I feel motivated to learn to program." VS "I'm sure I will have mastered programming by the exam."

			I am sure I will have mastered programming by the exam.		
			Disagree	Agree	Total
I feel motivated to learn to program	Disagree	Count	11	4	15
		% within I feel motivated to learn to program	73.3%	26.7%	100.0%
	Agree	Count	17	135	152
		% within I feel motivated to learn to program	11.2%	88.8%	100.0%
Total		Count	28	139	167
		% within I feel motivated to learn to program	16.8%	83.2%	100.0%

Table 5.19 Motivation to Learn V.S. Mastery by Final Exam

The above table showed that there were 15 respondents who disagree or strongly disagree that "I feel motivated to learn to program", including 11 (73.3%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 4 (26.7%) agree or strongly agree "I'm sure I will have mastered programming by the exam". Meanwhile, among the 152 respondents who agree or strongly agree that "I feel motivated to learn to program", there were 17 (11.2%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 135 (88.8%) agree or strongly agree "I'm sure I will have mastered programming by the exam". A Chi-square test was then used to test the following hypothesis:

H_0. Respondents' agreement level of "I feel motivated to learn to program" has no effect on their agreement level of "I'm sure I will have mastered programming by the exam".

H_1. Respondents' agreement level of "I feel motivated to learn to program" has some effects on their agreement level of "I'm sure I will have mastered programming by the exam".

The calculated chi-square value is 37.8 with an associated P value of 0.00. It could be concluded that the two statements "I feel motivated to learn to program" and "I'm sure I will have mastered programming by the exam" are not independent. Thus, hypothesis H_1 was accepted. In detail, students who are motivated to learn to program are more likely to think that they will have mastered programming than those who are not motivated.

5.4.2.5 *"I think learning to program will be easy for me." VS "I think I will have to work hard to learn to program."*

			I think I will have to work hard to learn to program.		
			Disagree	Agree	Total
I think learning to program will be easy for me.	Disagree	Count	6	62	68
		% within I think learning to program will be easy for me.	8.8%	91.2%	100.0%
	Agree	Count	22	79	101
		% within I think learning to program will be easy for me.	21.8%	78.2%	100.0%
Total		Count	28	141	169
		% within I think learning to program will be easy for me.	16.6%	83.4%	100.0%

Table 5.20 Ease of Programming V.S. Hard Work Requirement

The above table showed that there were 68 respondents who disagree or strongly disagree that "I think learning to program will be easy for me", including 6 (8.8%) students who disagree or strongly disagree "I think I will have to work hard to learn to program" and 62 (91.2%) agree or strongly agree "I think I will have to work hard to learn to program". Meanwhile, among the 101 respondents who agree or strongly agree that "I think learning to program will be easy for me", there were 22 (21.8%) students who disagree or strongly disagree "I think I will have to work hard to learn to program" and 79 (78.2%) agree or strongly agree "I think I will have to work hard to learn to program". A Chi-square test was then used to test the following hypothesis:

H_0. Respondents' agreement level of "I think learning to program will be easy for me" has no effect on their agreement level of "I think I will have to work hard to learn to program".

H_1. Respondents' agreement level of "I think learning to program will be easy for me" has some effects on their agreement level of "I think I will have to work hard to learn to program".

The calculated chi-square value is 4.94 with an associated P value of 0.026. It could be concluded that the two statements **"**I think learning to program will be easy for me" and "I think I will have to work hard to learn to program" are not independent. Thus, hypothesis H_1 was accepted. In detail, students who think learning to program will be easy for them are less likely to think that they will have to work hard than those who don't think it will be easy.

5.4.2.6 "I think learning to program will be easy for me." VS "I'm worried that learning to program will be difficult."

			I'm worried that programming will be difficult for me		
			Disagree	Agree	Total
I think learning to program will be easy for me.	Disagree	Count	21	47	68
		% within I think learning to program will be easy for me.	30.9%	69.1%	100.0%
	Agree	Count	71	30	101
		% within I think learning to program will be easy for me.	70.3%	29.7%	100.0%
Total		Count	92	77	169
		% within I think learning to program will be easy for me.	54.4%	45.6%	100.0%

Table 5.21 Ease of Programming V.S. Programming Difficulty

The above table showed that there were 68 respondents who disagree or strongly disagree that "I think learning to program will be easy for me", including 21 (30.9%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 47 (69.1%) agree or strongly agree "I'm worried that learning to program will be difficult". Meanwhile, among the 101 respondents who agree or strongly agree that "I think learning to program will be easy for me", there were 71 (70.3%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 30 (29.7%) agree or strongly agree "I'm worried that learning to program will be difficult". A Chi-square test was then used to test the following hypothesis:

H_0. Respondents' agreement level of "I think learning to program will be easy for me" has no effect on their agreement level of "I'm worried that learning to program will be difficult".

H_1. Respondents' agreement level of "I think learning to program will be easy for me" has some effects on their agreement level of "I'm worried that learning to program will be difficult".

The calculated chi-square value is 25.4 with an associated P value of 0.00. It could be concluded that the two statements "I think learning to program will be easy for me" and "I'm worried that learning to program will be difficult" are not independent. Thus, hypothesis H_1 was accepted. In detail, students who think learning to program will be easy for them are less worried that programming will be difficult than those who don't think it will be easy.

5.4.2.7 *"I think learning to program will be easy for me." VS "I'm sure I will have mastered programming by the exam."*

			I am sure I will have mastered programming by the exam.		Total
			Disagree	Agree	
I think learning to program will be easy for me.	Disagree	Count	19	49	68
		% within I think learning to program will be easy for me.	27.9%	72.1%	100.0%
	Agree	Count	9	92	101
		% within I think learning to program will be easy for me.	8.9%	91.1%	100.0%
Total		Count	28	141	169
		% within I think learning to program will be easy for me.	16.6%	83.4%	100.0%

Table 5.22 Ease of Programming V.S. Mastery by Final Exam

The above table showed that there were 68 respondents who disagree or strongly disagree that "I think learning to program will be easy for me", including 19 (27.9%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 49 (72.1%) agree or strongly agree "I'm sure I will have mastered programming by the exam". Meanwhile, among the 101 respondents who agree or strongly agree that "I think learning to program will be easy for me", there were 9 (8.9%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 92 (91.1%) agree or strongly agree "I'm sure I will have mastered programming by the exam". A Chi-square test was then used to test the following hypothesis:

H_0. Respondents' agreement level of "I think learning to program will be easy for me" has no effect on their agreement level of "I'm sure I will have mastered programming by the exam".

H_1. Respondents' agreement level of "I think learning to program will be easy for me" has some effects on their agreement level of "I'm sure I will have mastered programming by the exam".

The calculated chi-square value is 10.6 with an associated P value of 0.001. It could be concluded that the two statements **"**I think learning to program will be easy for me" and "I'm sure I will have mastered programming by the exam" are not independent. Thus, hypothesis H_1 was accepted. In detail, students who think learning to program will be easy for them are more confident that they will have mastered programming by the exam than those who don't think it will be easy.

5.4.2.8 "I think I will have to work hard to learn to program." VS "I'm worried that learning to program will be difficult."

			I'm worried that programming will be difficult for me		
			Disagree	Agree	Total
I think I will have to work hard to learn to program.	Disagree	Count	23	5	28
		% within I think I will have to work hard to learn to program.	82.1%	17.9%	100.0%
	Agree	Count	69	72	141
		% within I think I will have to work hard to learn to program.	48.9%	51.1%	100.0%
Total		Count	92	77	169
		% within I think I will have to work hard to learn to program.	54.4%	45.6%	100.0%

Table 5.23 Hard Work V.S. Programming Difficulty

The above table showed that there were 28 respondents who disagree or strongly disagree that "I think I will have to work hard to learn to program", including 23 (82.1%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 5 (17.9%) agree or strongly agree "I'm worried that learning to program will be difficult". Meanwhile, among the 141 respondents who agree or strongly agree that "I think I will have to work hard to learn to program", there were 69 (48.9%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 72 (51.1%) agree or strongly agree "I'm worried that learning to program will be difficult". A Chi-square test was then used to test the following hypothesis:

H_0. Respondents' agreement level of "I think I will have to work hard to learn to program" has no effect on their agreement level of "I'm worried that learning to program will be difficult".

H_1. Respondents' agreement level of "I think I will have to work hard to learn to program" has some effects on their agreement level of "I'm worried that learning to program will be difficult".

The calculated chi-square value is 10.4 with an associated P value of 0.001. It could be concluded that the two statements **"**I think I will have to work hard to learn to program" and "I'm worried that learning to program will be difficult" are not independent. Thus, hypothesis H_1 was accepted. In detail, students who think they will have to work hard to program are more worried that programming will be difficult for them than those who don't think they will have to work hard.

5.4.2.9 *"I think I will have to work hard to learn to program." VS "I'm sure I will have mastered programming by the exam."*

			I am sure I will have mastered programming by the exam.		
			Disagree	Agree	Total
I think I will have to work hard to learn to program.	Disagree	Count	8	20	28
		% within I think I will have to work hard to learn to program.	28.6%	71.4%	100.0%
	Agree	Count	20	121	141
		% within I think I will have to work hard to learn to program.	14.2%	85.8%	100.0%
Total		Count	28	141	169
		% within I think I will have to work hard to learn to program.	16.6%	83.4%	100.0%

Table 5.24 Hard Work V.S. Mastery by Final Exam

The above table showed that there were 28 respondents who disagree or strongly disagree that "I think I will have to work hard to learn to program", including 8 (28.6%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 20 (71.4%) agree or strongly agree "I'm sure I will have mastered programming by the exam". Meanwhile, among the 141 respondents who agree or strongly agree that "I think I will have to work hard to learn to program", there were 20 (14.2%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 121 (85.8%) agree or strongly agree "I'm sure I will have mastered programming by the exam". A Chi-square test was then used to test the following hypothesis:

H_0. Respondents' agreement level of "I think I will have to work hard to learn to program" has no effect on their agreement level of "I'm sure I will have mastered programming by the exam".

H_1. Respondents' agreement level of "I think I will have to work hard to learn to program" has some effects on their agreement level of "I'm sure I will have mastered programming by the exam".

The calculated chi-square value is 3.50, with an associated P value of 0.61. It could be concluded that the two statements **"**I think I will have to work hard to learn to program" and "I'm sure I will have mastered programming by the exam" are independent. Thus, hypothesis H_0 was accepted. In detail, being motivated to learn to program is not associated with thinking that learning to program will be easy.

5.4.2.10 "I'm worried that learning to program will be difficult." VS "I'm sure I will have mastered programming by the exam."

			I am sure I will have mastered programming by the exam.		
			Disagree	Agree	Total
I'm worried that programming will be difficult for me	Disagree	Count	15	77	92
		% within I'm worried that programming will be difficult for me	16.3%	83.7%	100.0%
	Agree	Count	13	64	77
		% within I'm worried that programming will be difficult for me	16.9%	83.1%	100.0%
Total		Count	28	141	169
		% within I'm worried that programming will be difficult for me	16.6%	83.4%	100.0%

Table 5.25 Programming Difficulty V.S. Mastery by Final Exam

The above table showed that there were 92 respondents who disagree or strongly disagree that "I'm worried that learning to program will be difficult", including 15 (16.3%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 77 (83.7%) agree or strongly agree "I'm sure I will have mastered programming by the exam". Meanwhile, among the 77 respondents who agree or strongly agree that "I'm worried that learning to program will be difficult", there were 13 (16.9%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 64 (83.1%) agree or strongly agree "I'm sure I will have mastered programming by the exam". A Chi-square test was then used to test the following hypothesis:

H_0. Respondents' agreement level of "I'm worried that learning to program will be difficult" has no effect on their agreement level of "I'm sure I will have mastered programming by the exam".

H_1. Respondents' agreement level of "I'm worried that learning to program will be difficult" has some effects on their agreement level of "I'm sure I will have mastered programming by the exam".

The calculated chi-square value is 0.01, with an associated P value of 0.92. It could be concluded that the two statements "I'm worried that learning to program will be difficult" and "I'm sure I will have mastered programming by the exam" are independent. Thus, hypothesis H_0 was accepted. In detail, being worried that programming will be difficult by students is not associated with their confidence that they will have mastered programming by the exam.

5.5 Attitude and Motivation Analysis by Subgroup

The central focus of this part of the study is related to how different subgroups of students differ in the way they respond to the motivation and attitude questions related to these statements:

1) I feel motivated to learn to program.
2) I think learning to program will be easy for me.
3) I think I will have to work hard to learn to program.
4) I'm worried that learning to program will be difficult.

5) I’m sure I will have mastered programming by the exam.

5.5.1 Age and Responses

5.5.1.1 I feel motivated to learn to program.

			I feel motivated to learn to program		
			Disagree	Agree	Total
Age	Below 21	Count	9	129	138
		% within Age	6.5%	93.5%	100.0%
	21 and above	Count	6	23	29
		% within Age	20.7%	79.3%	100.0%
Total		Count	15	152	167
		% within Age	9.0%	91.0%	100.0%

Table 5.26 Age Distribution V.S. Motivation to Learn

The above table showed that there were 138 respondents who were under 21 years old, including 9 (6.5%) students who disagree or strongly disagree “I feel motivated to learn to program” and 129 (93.5%) agree or strongly agree “I feel motivated to learn to program”. Meanwhile, among the 29 respondents who were over 21 years old, there were 6 (20.7%) students who disagree or strongly disagree “I feel motivated to learn to program” and 23 (79.3%) agree or strongly agree “I feel motivated to learn to program”. A Chi-square test was then used to test the following hypothesis:

H_0. Age has no effect on the agreement level of “I feel motivated to learn to program”.

H_1. Age has some effects on the agreement level of “I feel motivated to learn to program”.

The calculated chi-square value is 5.88 with an associated P value of 0.015. It could be concluded that age and the agreement level of the statement “I feel motivated to learn to program” are not independent. Thus, hypothesis H_1 was accepted. In detail, a greater proportion of the under 21 years old students feel motivated to program than those aged 21 and over.

5.5.1.2 I think learning to program will be easy for me.

			I think learning to program will be easy for me.		
			Disagree	Agree	Total
Age	Below 21	Count	53	87	140
		% within Age	37.9%	62.1%	100.0%
	21 and above	Count	15	14	29
		% within Age	51.7%	48.3%	100.0%
Total		Count	68	101	169
		% within Age	40.2%	59.8%	100.0%

Table 5.27 Age Distribution V.S. Ease of Learning

The above table showed that there were 140 respondents who were under 21 years old, including 53 (37.9%) students who disagree or strongly disagree “I think learning to program will be easy for me" and 87 (62.1%) agree or strongly agree "I think learning to program will be easy for me". Meanwhile, among the 29 respondents who were over 21 years old, there were 15 (51.7%) students who disagree or strongly disagree "I think learning to program will be easy for me" and 14 (48.3%) agree or strongly agree "I think learning to program will be easy for me". A Chi-square test was then used to test the following hypothesis:

H_0. Age has no effect on the agreement level of "I think learning to program will be easy for me".

H_1. Age has some effects on the agreement level of "I think learning to program will be easy for me".

The calculated chi-square value is 0.195 with an associated P value of 0.659. It could be concluded that age and the agreement level of the statement "I think learning to program will be easy for me" are independent. Thus, hypothesis H_0 was accepted - age has no effect on the agreement level of "I think learning to program will be easy for me".

5.5.1.3 I think I will have to work hard to learn to program.

			I think I will have to work hard to learn to program.		
			Disagree	Agree	Total
Age	Below 21	Count	24	116	140
		% within Age	17.1%	82.9%	100.0%
	21 and above	Count	4	25	29
		% within Age	13.8%	86.2%	100.0%
Total		Count	28	141	169
		% within Age	16.6%	83.4%	100.0%

Table 5.28 Age Distribution V.S. Hard Work Required

The above table showed that there were 140 respondents who were under 21 years old, including 24 (17.1%) students who disagree or strongly disagree "I think I will have to work hard to learn to program" and 116 (82.9%) agree or strongly agree "I think I will have to work hard to learn to program". Meanwhile, among the 29 respondents who were over 21 years old, there were 4 (13.8%) students who disagree or strongly disagree "I think I will have to work hard to learn to program" and 25 (86.2%) agree or strongly agree "I think I will have to work hard to learn to program". A Chi-square test was then used to test the following hypothesis:

H_0. Age has no effect on the agreement level of "I think I will have to work hard to learn to program".

H_1. Age has some effects on the agreement level of "I think I will have to work hard to learn to program".

The calculated chi-square value is 0.195 with an associated P value of 0.659. It could be concluded that age and the agreement level of the statement "I think I will have to work hard to learn to program" are independent. Thus, hypothesis H_0 was accepted - age has no effect on the agreement level of "I think I will have to work hard to learn to program".

5.5.1.4 *I am worried that learning to program will be difficult.*

			I'm worried that programming will be difficult for me		
			Disagree	Agree	Total
Age	Below 21	Count	77	63	140
		% within Age	55.0%	45.0%	100.0%
	21 and above	Count	15	14	29
		% within Age	51.7%	48.3%	100.0%
Total		Count	92	77	169
		% within Age	54.4%	45.6%	100.0%

Table 5.29 Age Distribution V.S. Programming Difficulty

The above table showed that there were 140 respondents who were under 21 years old, including 77 (55.0%) students who disagree or strongly disagree "I am worried that learning to program will be difficult" and 63 (45.0%) agree or strongly agree "I am worried that learning to program will be difficult". Meanwhile, among the 29 respondents who were over 21 years old, there were 15 (51.7%) students who disagree or strongly disagree "I am worried that learning to program will be difficult" and 14 (48.3%) agree or strongly agree "I am worried that learning to program will be difficult". A Chi-square test was then used to test the following hypothesis:

H_0. Age has no effect on the agreement level of "I am worried that learning to program will be difficult".

H_1. Age has some effects on the agreement level of "I am worried that learning to program will be difficult".

The calculated chi-square value is 0.104 with an associated P value of 0.747. It could be concluded that age and the agreement level of the statement "I am worried that learning to program will be difficult" are independent. Thus, hypothesis H_0 was accepted - age has no effect on the agreement level of "I am worried that learning to program will be difficult".

5.5.1.5 *I'm sure I will have mastered programming by the exam.*

			I am sure I will have mastered programming by the exam.		
			Disagree	Agree	Total
Age	Below 21	Count	23	117	140
		% within Age	16.4%	83.6%	100.0%
	21 and above	Count	5	24	29
		% within Age	17.2%	82.8%	100.0%
Total		Count	28	141	169
		% within Age	16.6%	83.4%	100.0%

Table 5.30 Age Distribution V.S. Mastery by Final Exam

The above table showed that there were 140 respondents who were under 21 years old, including 23 (16.4%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 117 (83.6%) agree or strongly agree "I'm sure I will have mastered programming by the exam". Meanwhile, among the 29 respondents who were over 21 years old, there were 5 (17.2%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 24 (82.8%) agree or strongly agree "I'm sure I will have mastered programming by the exam". A Chi-square test was then used to test the following hypothesis:

H_0. Age has no effect on the agreement level of "I'm sure I will have mastered programming by the exam".

H_1. Age has some effects on the agreement level of "I'm sure I will have mastered programming by the exam".

The calculated chi-square value is 0.011 with an associated P value of 0.915. It could be concluded that age and the agreement level of the statement "I'm sure I will have mastered programming by the exam" are independent. Thus, hypothesis H_0 was accepted - age has no effect on the agreement level of "I'm sure I will have mastered programming by the exam".

5.5.2 Gender and Responses

5.5.2.1 I feel motivated to learn to program.

			I feel motivated to learn to program		
			Disagree	Agree	Total
Gender	Male	Count	13	127	140
		% within Gender	9.3%	90.7%	100.0%
	Female	Count	2	25	27
		% within Gender	7.4%	92.6%	100.0%
Total		Count	15	152	167
		% within Gender	9.0%	91.0%	100.0%

Table 5.31 Gender V.S. Motivation to Learn Programming

The above table showed that there were 140 male respondents, including 13 (9.3%) students who disagree or strongly disagree "I feel motivated to learn to program" and 127 (90.7%) agree or strongly agree "I feel motivated to learn to program". Meanwhile, among the 27 female respondents, there were 2 (7.4%) students who disagree or strongly disagree "I feel motivated to learn to program" and 25 (92.6%) agree or strongly agree "I feel motivated to learn to program". A Chi-square test was then used to test the following hypothesis:

H_0. Gender has no effect on the agreement level of "I feel motivated to learn to program".

H_1. Gender has some effects on the agreement level of "I feel motivated to learn to program".

The calculated chi-square value is 0.098 with an associated P value of 0.755. It could be concluded that age and the agreement level of the statement "I feel motivated to learn to program" are independent. Thus, hypothesis H_0 was accepted - gender has no effect on the agreement level of "I feel motivated to learn to program".

5.5.2.2 I think learning to program will be easy for me.

			I think learning to program will be easy for me.		Total
			Disagree	Agree	
Gender	Male	Count	56	86	142
		% within Gender	39.4%	60.6%	100.0%
	Female	Count	12	15	27
		% within Gender	44.4%	55.6%	100.0%
Total		Count	68	101	169
		% within Gender	40.2%	59.8%	100.0%

Table 5.32 Gender V.S. Ease of Learning

The above table showed that there were 142 male respondents, including 56 (39.4%) students who disagree or strongly disagree "I think learning to program will be easy for me" and 86 (60.6%) agree or strongly agree "I think learning to program will be easy for me". Meanwhile, among the 27 female respondents, there were 12 (44.4%) students who disagree or strongly disagree "I think learning to program will be easy for me" and 15 (55.6%) agree or strongly agree "I think learning to program will be easy for me". A Chi-square test was then used to test the following hypothesis:

H_0. Gender has no effect on the agreement level of "I think learning to program will be easy for me".

H_1. Gender has some effects on the agreement level of "I think learning to program will be easy for me".

The calculated chi-square value is 0.237 with an associated P value of 0.627. It could be concluded that age and the agreement level of the statement "I think learning to program will be easy for me" are independent. Thus, hypothesis H_0 was accepted - gender has no effect on the agreement level of "I think learning to program will be easy for me".

5.5.2.3 I think I will have to work hard to learn to program.

			I think I will have to work hard to learn to program.		Total
			Disagree	Agree	
Gender	Male	Count	24	118	142
		% within Gender	16.9%	83.1%	100.0%
	Female	Count	4	23	27
		% within Gender	14.8%	85.2%	100.0%
Total		Count	28	141	169
		% within Gender	16.6%	83.4%	100.0%

Table 5.33 Gender V.S. Hard Work Required

The above table showed that there were 142 male respondents, including 24 (16.9%) students who disagree or strongly disagree "I think I will have to work hard to learn to program" and 118 (83.1%) agree or strongly agree "I think I will have to work hard to learn to program". Meanwhile, among the 27 female respondents, there were 4 (14.8%) students who disagree or strongly disagree "I think

I will have to work hard to learn to program" and 23 (85.2%) agree or strongly agree "I think I will have to work hard to learn to program". A Chi-square test was then used to test the following hypothesis:

H_0. Gender has no effect on the agreement level of "I think I will have to work hard to learn to program".

H_1. Gender has some effects on the agreement level of "I think I will have to work hard to learn to program".

The calculated chi-square value is 0.071 with an associated P value of 0.789. It could be concluded that age and the agreement level of the statement "I think I will have to work hard to learn to program" are independent. Thus, hypothesis H_0 was accepted - gender has no effect on the agreement level of "I think I will have to work hard to learn to program".

5.5.2.4 I'm worried that learning to program will be difficult.

			I'm worried that programming will be difficult for me		
			Disagree	Agree	Total
Gender	Male	Count	81	61	142
		% within Gender	57.0%	43.0%	100.0%
	Female	Count	11	16	27
		% within Gender	40.7%	59.3%	100.0%
Total		Count	92	77	169
		% within Gender	54.4%	45.6%	100.0%

Table 5.34 Gender V.S. Difficulty of Programming

The above table showed that there were 142 male respondents, including 81 (57.0%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 61 (43.0%) agree or strongly agree "I'm worried that learning to program will be difficult". Meanwhile, among the 27 female respondents, there were 11 (40.7%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 16 (59.3%) agree or strongly agree "I'm worried that learning to program will be difficult". A Chi-square test was then used to test the following hypothesis:

H_0. Gender has no effect on the agreement level of "I'm worried that learning to program will be difficult".

H_1. Gender has some effects on the agreement level of "I'm worried that learning to program will be difficult".

The calculated chi-square value is 2.43 with an associated P value of 0.119. It could be concluded that age and the agreement level of the statement "I'm worried that learning to program will be difficult" are independent. Thus, hypothesis H_0 was accepted - gender has no effect on the agreement level of "I'm worried that learning to program will be difficult".

5.5.2.5 I'm sure I will have mastered programming by the exam.

			I am sure I will have mastered programming by the exam.		
			Disagree	Agree	Total
Gender	Male	Count	24	118	142
		% within Gender	16.9%	83.1%	100.0%
	Female	Count	4	23	27
		% within Gender	14.8%	85.2%	100.0%
Total		Count	28	141	169
		% within Gender	16.6%	83.4%	100.0%

Table 5.35 Gender V.S. Mastery Before Final Exam

The above table showed that there were 142 male respondents, including 24 (16.9%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 118 (83.1%) agree or strongly agree "I'm sure I will have mastered programming by the exam". Meanwhile, among the 27 female respondents, there were 4 (14.8%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 23 (85.2%) agree or strongly agree "I'm sure I will have mastered programming by the exam". A Chi-square test was then used to test the following hypothesis:

H_0. Gender has no effect on the agreement level of "I'm sure I will have mastered programming by the exam".

H_1. Gender has some effects on the agreement level of "I'm sure I will have mastered programming by the exam".

The calculated chi-square value is 0.071 with an associated P value of 0.789. It could be concluded that age and the agreement level of the statement "I'm sure I will have mastered programming by the exam" are independent. Thus, hypothesis H_0 was accepted - gender has no effect on the agreement level of "I'm sure I will have mastered programming by the exam".

5.5.3 Origin and Responses

5.5.3.1 I feel motivated to learn to program.

			I feel motivated to learn to program		
			Disagree	Agree	Total
Origin	UK	Count	10	83	93
		% within Origin	10.8%	89.2%	100.0%
	EU	Count	0	22	22
		% within Origin	0.0%	100.0%	100.0%
	International	Count	5	47	52
		% within Origin	9.6%	90.4%	100.0%
Total		Count	15	152	167
		% within Origin	9.0%	91.0%	100.0%

Table 5.36 Region V.S. Motivation to Learn

The above table showed that there were 93 UK respondents, including 10 (10.8%) students who disagree or strongly disagree "I feel motivated to learn to program" and 83 (89.2%) agree or strongly agree this statement. Meanwhile, all the 22 (100%) EU respondents agree or strongly agree "I feel motivated to learn to program", whilst 5 out of 52 international students (9.6%) disagree or strongly disagree and the rest 47 (90.4%) international students agree or strongly agree this statement. A Chi-square test was then used to test the following hypothesis:

H_0. Origin has no effect on the agreement level of "I feel motivated to learn to program".

H_1. Origin has some effects on the agreement level of "I feel motivated to learn to program".

The calculated chi-square value is 2.553 with an associated P value of 0.279. It could be concluded that origin and the agreement level of the statement "I feel motivated to learn to program" are independent factors. Thus, hypothesis H_0 was accepted - origin has no effect on the agreement level of "I feel motivated to learn to program".

5.5.3.2 I think learning to program will be easy for me.

			I think learning to program will be easy for me.		
			Disagree	Agree	Total
Origin	UK	Count	39	55	94
		% within Origin	41.5%	58.5%	100.0%
	EU	Count	7	16	23
		% within Origin	30.4%	69.6%	100.0%
	International	Count	22	30	52
		% within Origin	42.3%	57.7%	100.0%
Total		Count	68	101	169
		% within Origin	40.2%	59.8%	100.0%

Table 5.37 Region V.S. Ease of Learning Programming

The above table showed that there were 94 UK respondents, including 39 (41.5%) students who disagree or strongly disagree "I think learning to program will be easy for me" and 55 (58.5%) agree or strongly agree this statement. Meanwhile, among the 23 EU respondents, there were 7 (30.4%) disagree or strongly disagree "I think learning to program will be easy for me" and 16 (69.6%) agree or strongly agree this statement, whilst 22 out of 52 international students (42.3%) disagree or strongly disagree and the rest 30 (57.7%) international students agree or strongly agree this statement. A Chi-square test was then used to test the following hypothesis:

H_0. Origin has no effect on the agreement level of "I think learning to program will be easy for me".

H_1. Origin has some effects on the agreement level of "I think learning to program will be easy for me".

The calculated chi-square value is 1.073 with an associated P value of 0.585. It could be concluded that origin and the agreement level of the statement "I think learning to program will be easy for me" are independent factors. Thus, hypothesis H_0 was accepted - origin has no effect on the agreement level of "I think learning to program will be easy for me".

5.5.3.3 I think I will have to work hard to learn to program.

			I think I will have to work hard to learn to program.		
			Disagree	Agree	Total
Origin	UK	Count	18	76	94
		% within Origin	19.1%	80.9%	100.0%
	EU	Count	4	19	23
		% within Origin	17.4%	82.6%	100.0%
	International	Count	6	46	52
		% within Origin	11.5%	88.5%	100.0%
Total		Count	28	141	169
		% within Origin	16.6%	83.4%	100.0%

Table 5.38 Region V.S. Expected Hard Work

The above table showed that there were 94 UK respondents, including 18 (19.1%) students who disagree or strongly disagree “I think I will have to work hard to learn to program” and 76 (80.9%) agree or strongly agree this statement. Meanwhile, among the 23 EU respondents, there were 4 (17.4%) disagree or strongly disagree “I think I will have to work hard to learn to program” and 19 (82.6%) agree or strongly agree this statement, whilst 6 out of 52 international students (11.5%) disagree or strongly disagree and the rest 46 (88.5%) international students agree or strongly agree this statement. A Chi-square test was then used to test the following hypothesis:

H_0. Origin has no effect on the agreement level of “I think I will have to work hard to learn to program”.

H_1. Origin has some effects on the agreement level of “I think I will have to work hard to learn to program”.

The calculated chi-square value is 1.42 with an associated P value of 0.493. It could be concluded that origin and the agreement level of the statement “I think I will have to work hard to learn to program” are independent factors. Thus, hypothesis H_0 was accepted - origin has no effect on the agreement level of “I think I will have to work hard to learn to program”.

5.5.3.4 I'm worried that learning to program will be difficult.

			I'm worried that programming will be difficult for me		
			Disagree	Agree	Total
Origin	UK	Count	56	38	94
		% within Origin	59.6%	40.4%	100.0%
	EU	Count	15	8	23
		% within Origin	65.2%	34.8%	100.0%
	International	Count	21	31	52
		% within Origin	40.4%	59.6%	100.0%
Total		Count	92	77	169
		% within Origin	54.4%	45.6%	100.0%

Table 5.39 Region V.S. Programming Difficulty

The above table showed that there were 94 UK respondents, including 56 (59.6%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 38 (40.4%) agree or strongly agree this statement. Meanwhile, among the 23 EU respondents, there were 15 (65.2%) disagree or strongly disagree "I'm worried that learning to program will be difficult" and 8 (34.8%) agree or strongly agree this statement, whilst 21 out of 52 international students (40.4%) disagree or strongly disagree and the rest 31 (59.6%) international students agree or strongly agree this statement. A Chi-square test was then used to test the following hypothesis:

H_0. Origin has no effect on the agreement level of "I'm worried that learning to program will be difficult".

H_1. Origin has some effects on the agreement level of "I'm worried that learning to program will be difficult".

The calculated chi-square value is 6.22 with an associated P value of 0.045. It could be concluded that origin and the agreement level of the statement "I'm worried that learning to program will be difficult" are not independent factors. Thus, hypothesis H_1 was accepted. In detail, the proportions who agree with the statement are quite different in the difference regions of origin ranging from 40% for students from the UK to nearly 60% for international students.

5.5.3.5 *I'm sure I will have mastered programming by the exam.*

			I am sure I will have mastered programming by the exam.		
			Disagree	Agree	Total
Origin	UK	Count	18	76	94
		% within Origin	19.1%	80.9%	100.0%
	EU	Count	2	21	23
		% within Origin	8.7%	91.3%	100.0%
	International	Count	8	44	52
		% within Origin	15.4%	84.6%	100.0%
Total		Count	28	141	169
		% within Origin	16.6%	83.4%	100.0%

Table 5.40 Region V.S. Mastery Before Final Exam

The above table showed that there were 94 UK respondents, including 18 (19.1%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 76 (80.9%) agree or strongly agree this statement. Meanwhile, among the 23 EU respondents, there were 2 (8.7%) disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 21 (91.3%) agree or strongly agree this statement, whilst 8 out of 52 international students (15.4%) disagree or strongly disagree and the rest 44 (84.6%) international students agree or strongly agree this statement. A Chi-square test was then used to test the following hypothesis:

H_0. Origin has no effect on the agreement level of "I'm sure I will have mastered programming by the exam".

H_1. Origin has some effects on the agreement level of "I'm sure I will have mastered programming by the exam".

The calculated chi-square value is 1.54 with an associated P value of 0.464. It could be concluded that origin and the agreement level of the statement "I'm sure I will have mastered programming by the exam" are independent factors. Thus, hypothesis H_0 was accepted - origin has no effect on the agreement level of "I'm sure I will have mastered programming by the exam".

5.5.4 Course and Responses

5.5.4.1 *I feel motivated to learn to program.*

			I feel motivated to learn to program		
			Disagree	Agree	Total
Course	CS/SE/ITO	Count	8	65	73
		% within Course	11.0%	89.0%	100.0%
	Engineering	Count	7	87	94
		% within Course	7.4%	92.6%	100.0%
Total		Count	15	152	167
		% within Course	9.0%	91.0%	100.0%

Table 5.41 Degree Type V.S. Motivation to Learn

The above table showed that there were 73 respondents who enrolled in Computer Science related courses (Computer Science, Software Engineering, Information Technology and Organization), including 8 (11.0%) students who disagree or strongly disagree “I feel motivated to learn to program” and 65 (89.0%) agree or strongly agree “I feel motivated to learn to program”. Meanwhile, among the 94 respondents who enrolled in Electronic Engineering related courses (Electrical Engineering, Electromechanical Engineering and Electronic Engineering), there were 7 (7.4%) students who disagree or strongly disagree “I feel motivated to learn to program” and 87 (92.6%) agree or strongly agree “I feel motivated to learn to program”. A Chi-square test was then used to test the following hypothesis:

H_0. Course has no effect on the agreement level of “I feel motivated to learn to program”.

H_1. Course has some effects on the agreement level of “I feel motivated to learn to program”.

The calculated chi-square value is 0.62 with an associated P value of 0.431. It could be concluded that the course and the agreement level of the statement “I feel motivated to learn to program” are independent. Thus, hypothesis H_0 was accepted - course has no effect on the agreement level of “I feel motivated to learn to program”.

5.5.4.2 I think learning to program will be easy for me.

			I think learning to program will be easy for me.		
			Disagree	Agree	Total
Course	CS/SE/ITO	Count	25	49	74
		% within Course	33.8%	66.2%	100.0%
	Engineering	Count	43	52	95
		% within Course	45.3%	54.7%	100.0%
Total		Count	68	101	169
		% within Course	40.2%	59.8%	100.0%

Table 5.42 Degree Type V.S. Ease of Learning to Program

The above table showed that there were 74 respondents who enrolled in Computer Science related courses (Computer Science, Software Engineering, Information Technology and Organization), including 25 (33.8%) students who disagree or strongly disagree “I think learning to program will be easy for me” and 49 (66.2%) agree or strongly agree “I think learning to program will be easy for me”. Meanwhile, among the 95 respondents who enrolled in Electronic Engineering related courses (Electrical Engineering, Electromechanical Engineering and Electronic Engineering), there were 43 (45.3%) students who disagree or strongly disagree “I think learning to program will be easy for me” and 52 (54.7%) agree or strongly agree “I think learning to program will be easy for me”. A Chi-square test was then used to test the following hypothesis:

H_0. Course has no effect on the agreement level of “I think learning to program will be easy for me”.

H_1. Course has some effects on the agreement level of “I think learning to program will be easy for me”.

The calculated chi-square value is 2.28 with an associated P value of 0.131. It could be concluded that the course and the agreement level of the statement "I think learning to program will be easy for me" are independent. Thus, hypothesis H_0 was accepted - course has no effect on the agreement level of "I think learning to program will be easy for me".

5.5.4.3 I think I will have to work hard to learn to program.

			I think I will have to work hard to learn to program.		
			Disagree	Agree	Total
Course	CS/SE/ITO	Count	15	59	74
		% within Course	20.3%	79.7%	100.0%
	Engineering	Count	13	82	95
		% within Course	13.7%	86.3%	100.0%
Total		Count	28	141	169
		% within Course	16.6%	83.4%	100.0%

Table 5.43 Degree Type V.S. Working Hard to Learn Programming

The above table showed that there were 74 respondents who enrolled in Computer Science related courses (Computer Science, Software Engineering, Information Technology and Organization), including 15 (20.3%) students who disagree or strongly disagree "I think I will have to work hard to learn to program" and 59 (79.7%) agree or strongly agree "I think I will have to work hard to learn to program". Meanwhile, among the 95 respondents who enrolled in Electronic Engineering related courses (Electrical Engineering, Electromechanical Engineering and Electronic Engineering), there were 13 (13.7%) students who disagree or strongly disagree "I think I will have to work hard to learn to program" and 82 (86.3%) agree or strongly agree "I think I will have to work hard to learn to program". A Chi-square test was then used to test the following hypothesis:

H_0. Course has no effect on the agreement level of "I think I will have to work hard to learn to program".

H_1. Course has some effects on the agreement level of "I think I will have to work hard to learn to program".

The calculated chi-square value is 1.31 with an associated P value of 0.253. It could be concluded that the course and the agreement level of the statement "I think I will have to work hard to learn to program" are independent. Thus, hypothesis H_0 was accepted - course has no effect on the agreement level of "I think I will have to work hard to learn to program".

5.5.4.4 I'm worried that learning to program will be difficult.

			I'm worried that programming will be difficult for me		
			Disagree	Agree	Total
Course	CS/SE/ITO	Count	46	28	74
		% within Course	62.2%	37.8%	100.0%
	Engineering	Count	46	49	95
		% within Course	48.4%	51.6%	100.0%
Total		Count	92	77	169
		% within Course	54.4%	45.6%	100.0%

Table 5.44 Degree Type V.S. Difficulty Programming

The above table showed that there were 74 respondents who enrolled in Computer Science related courses (Computer Science, Software Engineering, Information Technology and Organization), including 46 (62.2%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 28 (37.8%) agree or strongly agree "I'm worried that learning to program will be difficult". Meanwhile, among the 95 respondents who enrolled in Electronic Engineering related courses (Electrical Engineering, Electromechanical Engineering and Electronic Engineering), there were 46 (48.4%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 49 (51.6%) agree or strongly agree "I'm worried that learning to program will be difficult". A Chi-square test was then used to test the following hypothesis:

H_0. Course has no effect on the agreement level of "I'm worried that learning to program will be difficult".

H_1. Course has some effects on the agreement level of "I'm worried that learning to program will be difficult".

The calculated chi-square value is 3.17 with an associated P value of 0.075. It could be concluded that the course and the agreement level of the statement "I'm worried that learning to program will be difficult" are independent. Thus, hypothesis H_0 was accepted - course has no effect on the agreement level of "I'm worried that learning to program will be difficult".

5.5.4.5 I'm sure I will have mastered programming by the exam.

			I am sure I will have mastered programming by the exam.		
			Disagree	Agree	Total
Course	CS/SE/ITO	Count	16	58	74
		% within Course	21.6%	78.4%	100.0%
	Engineering	Count	12	83	95
		% within Course	12.6%	87.4%	100.0%
Total		Count	28	141	169
		% within Course	16.6%	83.4%	100.0%

Table 5.45 Degree Type V.S. Programming Mastery by Exam

The above table showed that there were 74 respondents who enrolled in Computer Science related courses (Computer Science, Software Engineering, Information Technology and Organization), including 16 (21.6%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 58 (78.4%) agree or strongly agree "I'm sure I will have mastered programming by the exam". Meanwhile, among the 95 respondents who enrolled in Electronic Engineering related courses (Electrical Engineering, Electromechanical Engineering and Electronic Engineering), there were 12 (12.6%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 83 (87.4%) agree or strongly agree "I'm sure I will have mastered programming by the exam". A Chi-square test was then used to test the following hypothesis:

H_0. Course has no effect on the agreement level of "I'm sure I will have mastered programming by the exam".

H_1. Course has some effects on the agreement level of "I'm sure I will have mastered programming by the exam".

The calculated chi-square value is 2.43 with an associated P value of 0.119. It could be concluded that the course and the agreement level of the statement "I'm sure I will have mastered programming by the exam" are independent. Thus, hypothesis H_0 was accepted - course has no effect on the agreement level of "I'm sure I will have mastered programming by the exam".

5.5.5 Prior Programming Experience and Responses

5.5.5.1 I feel motivated to learn to program.

			I feel motivated to learn to program		
			Disagree	Agree	Total
Pre-experience	None/A little	Count	13	133	146
		% within Pre-experience	8.9%	91.1%	100.0%
	A lot/Extensive	Count	2	19	21
		% within Pre-experience	9.5%	90.5%	100.0%
Total		Count	15	152	167
		% within Pre-experience	9.0%	91.0%	100.0%

Table 5.46 Prior Programming Experience V.S. Motivation to Learn

The above table showed that there were 146 respondents who had no or a little prior programming experience, including 13 (8.9%) students who disagree or strongly disagree "I feel motivated to learn to program" and 133 (91.1%) agree or strongly agree "I feel motivated to learn to program". Meanwhile, among the 21 respondents who had a good foundation of programming skills, there were 2 (9.5%) students who disagree or strongly disagree "I feel motivated to learn to program" and 19 (90.5%) agree or strongly agree "I feel motivated to learn to program". A Chi-square test was then used to test the following hypothesis:

H_0. Prior programming experience has no effect on the agreement level of "I feel motivated to learn to program".

H_1. Prior programming experience has some effects on the agreement level of "I feel motivated to learn to program".

The calculated chi-square value is 0.009 with an associated P value of 0.926. It could be concluded that prior programming experience and the agreement level of the statement "I feel motivated to learn to program" are independent. Thus, hypothesis H_0 was accepted - prior programming experience has no effect on the agreement level of "I feel motivated to learn to program".

5.5.5.2 I think learning to program will be easy for me.

			I think learning to program will be easy for me.		
			Disagree	Agree	Total
Pre-experience	None/A little	Count	64	83	147
		% within Pre-experience	43.5%	56.5%	100.0%
	A lot/Extensive	Count	4	18	22
		% within Pre-experience	18.2%	81.8%	100.0%
Total		Count	68	101	169
		% within Pre-experience	40.2%	59.8%	100.0%

Table 5.47 Prior Programming Experience V.S. Ease of Learning

The above table showed that there were 147 respondents who had no or a little prior programming experience, including 64 (43.5%) students who disagree or strongly disagree "I think learning to program will be easy for me" and 83 (56.5%) agree or strongly agree "I think learning to program will be easy for me". Meanwhile, among the 22 respondents who had a good foundation of programming skills, there were 4 (18.2%) students who disagree or strongly disagree "I think learning to program will be easy for me" and 18 (81.8%) agree or strongly agree "I think learning to program will be easy for me". A Chi-square test was then used to test the following hypothesis:

H_0. Prior programming experience has no effect on the agreement level of "I think learning to program will be easy for me".

H_1. Prior programming experience has some effects on the agreement level of "I think learning to program will be easy for me".

The calculated chi-square value is 5.112 with an associated P value of 0.024. It could be concluded that prior programming experience and the agreement level of the statement "I think learning to program will be easy for me" are not independent. Thus, hypothesis H_1 was accepted. In details, the proportions who agree with the statement are quite different according to the level of prior programming experience. A significantly higher proportion of those with prior experience feel that learning to program will be easy.

5.5.5.3 I think I will have to work hard to learn to program.

			I think I will have to work hard to learn to program.		
			Disagree	Agree	Total
Pre-experience	None/A little	Count	20	127	147
		% within Pre-experience	13.6%	86.4%	100.0%
	A lot/Extensive	Count	8	14	22
		% within Pre-experience	36.4%	63.6%	100.0%
Total		Count	28	141	169
		% within Pre-experience	16.6%	83.4%	100.0%

Table 5.48 Prior Programming Experience V.S. Hard Work to Program

The above table showed that there were 147 respondents who had no or a little prior programming experience, including 20 (13.6%) students who disagree or strongly disagree “I think I will have to work hard to learn to program” and 127 (86.4%) agree or strongly agree “I think I will have to work hard to learn to program”. Meanwhile, among the 22 respondents who had a good foundation of programming skills, there were 8 (36.4%) students who disagree or strongly disagree “I think I will have to work hard to learn to program” and 14 (63.6%) agree or strongly agree “I think I will have to work hard to learn to program”. A Chi-square test was then used to test the following hypothesis:

H_0. Prior programming experience has no effect on the agreement level of “I think I will have to work hard to learn to program”.

H_1. Prior programming experience has some effects on the agreement level of “I think I will have to work hard to learn to program”.

The calculated chi-square value is 7.17 with an associated P value of 0.007. It could be concluded that prior programming experience and the agreement level of the statement “I think I will have to work hard to learn to program” are not independent. Thus, hypothesis H_1 was accepted. In details, the proportions who agree with the statement are quite different according to the level of prior programming experience. A significantly higher proportion of those with no prior experience feel that they will have to work hard to learn programming.

5.5.5.4 I'm worried that learning to program will be difficult.

			I'm worried that programming will be difficult for me		
			Disagree	Agree	Total
Pre-experience	None/A little	Count	75	72	147
		% within Pre-experience	51.0%	49.0%	100.0%
	A lot/Extensive	Count	17	5	22
		% within Pre-experience	77.3%	22.7%	100.0%
Total		Count	92	77	169
		% within Pre-experience	54.4%	45.6%	100.0%

Table 5.49 Prior Programming Experience V.S. Difficulty to Learn Programming

The above table showed that there were 147 respondents who had no or a little prior programming experience, including 75 (51.0%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 72 (49.0%) agree or strongly agree "I'm worried that learning to program will be difficult". Meanwhile, among the 22 respondents who had a good foundation of programming skills, there were 17 (77.3%) students who disagree or strongly disagree "I'm worried that learning to program will be difficult" and 5 (22.7%) agree or strongly agree "I'm worried that learning to program will be difficult". A Chi-square test was then used to test the following hypothesis:

H_0. Prior programming experience has no effect on the agreement level of "I'm worried that learning to program will be difficult".

H_1. Prior programming experience has some effects on the agreement level of "I'm worried that learning to program will be difficult".

The calculated chi-square value is 5.32 with an associated P value of 0.021. It could be concluded that prior programming experience and the agreement level of the statement "I'm worried that learning to program will be difficult" are not independent. Thus, hypothesis H_1 was accepted. In details, the proportions who agree with the statement are quite different according to the level of prior programming experience. A significantly higher proportion of those with no prior experience feel worried that they will program difficult.

5.5.5.5 I'm sure I will have mastered programming by the exam.

			I am sure I will have mastered programming by the exam.		
			Disagree	Agree	Total
Pre-experience	None/A little	Count	24	123	147
		% within Pre-experience	16.3%	83.7%	100.0%
	A lot/Extensive	Count	4	18	22
		% within Pre-experience	18.2%	81.8%	100.0%
Total		Count	28	141	169
		% within Pre-experience	16.6%	83.4%	100.0%

Table 5.50 Prior Programming Experience V.S. Mastering Programming by Final Exam

The above table showed that there were 147 respondents who had no or a little prior programming experience, including 24 (16.3%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 123 (83.7%) agree or strongly agree "I'm sure I will have mastered programming by the exam". Meanwhile, among the 22 respondents who had a good foundation of programming skills, there were 4 (18.2%) students who disagree or strongly disagree "I'm sure I will have mastered programming by the exam" and 18 (81.8%) agree or strongly agree "I'm sure I will have mastered programming by the exam". A Chi-square test was then used to test the following hypothesis:

H_0. Prior programming experience has no effect on the agreement level of "I'm sure I will have mastered programming by the exam".

H_1. Prior programming experience has some effects on the agreement level of "I'm sure I will have mastered programming by the exam".

The calculated chi-square value is 0.048 with an associated P value of 0.827. It could be concluded that prior programming experience and the agreement level of the statement "I'm sure I will have mastered programming by the exam" are not independent. Thus, hypothesis H_0 was accepted - prior programming experience has no effect on the agreement level of "I'm sure I will have mastered programming by the exam".

5.6 Summary

Having analysed the evidence gathered, the following statements could be made about Question A which was to determine how the students' backgrounds, attitudes and beliefs might affect their approach and practice to learning. Using the results obtained from the initial student survey responses as reported, the following statements were tested and their results are reported.

These are the statements that were made in order to test what certain student attributes might have on the expected course learning experiences. The statements that were tested are:

5.6.1 Beliefs and Prior Experience

B1. Students that have prior programming experience will not inevitably result in confidence of their abilities to master the final exam and more motivated to learn to program.

According to Section 5.5.5.5, current evidence (Table 5.50) shows that prior programming experience has no effect to students' abilities to master the final exam. In details, there were 23.7% of the initial study who had no prior programming experience and 123 out of 147 (83.7%) believed that they would master programming by the exam versus 81.8% of students with "a lot or extensive" previous programming. Meanwhile, Table 5.46 (Section 5.5.5.1) shows that prior programming experience has no effect to students' initial motivation to learn to program.

B2. Students that have no or little prior programming experience will worry that they have to work hard and learning to program will be difficult.

According to Section 5.5.5.3, the data gathered shows that 127 out of 147 students (86.4%) who have no or little prior programming experience felt they have to work hard to learning to program, whilst 14 out of 22 pre-experienced students (63.6%) believe so (Table 5.48). 72 out of 147 novices (49.0%) with no or little prior programming experience worried that learning to program is a difficult task, whilst 5 out of 22 pre-experienced students (22.7%) have a similar concern (Table 5.49 in Section 5.5.5.4).

B3. Students who express a motivation to learn will be more confident of their ability learn to program.

According to Section 5.4.2.1 & 5.4.2.3, students who feel motivated are more likely to consider learning to program easy (Table 5.16 & Table 5.18). In details, of the respondents who indicated that they were motivated, 63.8% (97) believed that learning to program would be easy and 53.3% (81)

were not worried that programming would be difficult.

B4. Students who believe that programming is easy will express less motivation to work hard.

According to Section 5.4.2.5, students who consider learning to program easy are less likely to believe they need to work hard. In details, of the students who stated that learning to program would be easy for them (101 out of 169), 78.2% (79) believed that they would have to work hard to learn to program (Table 5.20).

B5. Students who believe that programming is easy will express more confidence in having mastered programming by the exam.

According to Section 5.4.2.7, students who consider learning to program easy are more likely to believe they could master the final exam. In details, of the students that stated they found programming to be easy, 92 out of 101 (91.1%) said they were sure they would master programming by the time of the exam (Table 5.22).

5.6.2 Demographic Factors

A1. Age has some effects on opinions about the difficulty in computer programming.

According to Section 5.5.1.4, there were 87 out of 140 (62.1%) under age 21 who believed that programming was easy versus 14 out of 29 (48.3%) over age 21 who believed that learning to program was easy (Table 5.29).

A2. Age has some effects on opinions about the ability have mastered programming by the exam.

According to Section 5.5.1.5, of the respondents who stated they would master programming by exam time, 117 of 140 (83.6%) respondents were under age 21 while 24 of respondents over age 21 (82.8%) stated they would master programming by the time of the exam (Table 5.30).

A3. Age has some effects on the willingness to work hard to learn to program.

According to Section 5.5.1.3, there were 25 out of 29 respondents (86.2%) over age 21 who indicated that they would have to work hard to learn programming versus 115 out of 140 respondents (82.9%) who believed that they would have to work hard to learn programming (Table 5.28).

A4. Age has some effects on opinions about the motivation in computer programming.

According to Section 5.5.1.1, students below 21 years old feel motivated to learn to program. In details, there were 129 out of 138 (93.5%) under age 21 who believed that they are motivated versus 23 out of 29 (79.3%) over age 21 who believed so (Table 5.26).

G1. Gender has no effect on opinions about the difficulty in computer programming.

According to Section 5.5.2.4, there were 86 out of 142 (60.6%) males who believed that programming was easy versus 15 out of 27 (55.6%) females who believed that learning to program was easy (Table 5.34).

G2. Gender has no effect on opinions about the ability have mastered programming by the exam.

According to Section 5.5.2.5, of the respondents who stated they would master programming by exam time, 23 of 27 (85.2%) females versus 118 out of 142 male respondents (83.1%) stated they would master programming by the time of the exam (Table 5.35).

G3. Gender has no effect on the willingness to work hard to learn to program.

According to Section 5.5.2.3, there were 23 out of 27 females (85.2%) who indicated that they would have to work hard to learn programming versus 118 out of 142 males (83.1%) who stated they would need to work hard (Table 5.33).

G4. Gender has no effect on opinions about the motivation in computer programming.

According to Section 5.5.2.1, there were 25 out of 27 females (92.6%) who felt motivated versus 127 out of 140 (90.7%) males (Table 5.31).

R1. Region of origin has some effects on opinions about the difficulty in computer programming.

According to Section 5.5.3.4, there were 16 out of 23 respondents (69.6%) from the EU students who believed that programming was easy, versus 55 out of 94 (58.5%) UK students and 30 out of 52 (57.7%) international students who believed that learning to program were easy. There were 46 out of 75 respondents (61.3%) from the non-UK students who believed that programming was easy (Table 5.39).

R2. Region of origin has no effect on opinions about the ability have mastered programming by the exam.

According to Section 5.5.3.5, of the respondents who stated they would master programming by exam time, 21 of 23 (91.3%) from the EU versus 44 out of 52 International students (84.6%) and 76 out of 94 UK students (80.9%) stated they would master programming by the time of the exam. There were 65 of 75 (86.6%) students from the non-UK who stated they would master programming by the time of the exam (Table 5.40).

R3. Region of origin has no effect on the willingness to work hard to learn to program.

According to Section 5.5.3.3, there were 46 out of 52 International students (88.5%) who indicated that they would have to work hard to learn programming versus 19 out of 23 EU students (82.6%) and 76 out of 94 (80.9%) UK students who stated they would need to work hard. There were 65 out of 75 non-UK students (86.6%) who indicated that they would have to work hard to learn programming (Table 5.38).

R4. Region of origin has no effect on opinions about the motivation in computer programming.

According to the findings discussed in Section 5.5.3.1, there were 47 out of 52 International students (90.4%) who felt motivated to learn to program versus 22 out of 22 EU students (100%) and 83 out of 93 (89.2%) UK students who felt the same. There were 69 out of 74 non-UK students (93.2%) who agreed that they were motivated (Table 5.36).

This was not a homogeneous group, so it might be expected that there could be some variability if a future cohort were to be observed and the research was replicated. Out of 169 total respondents, there was a disproportionately low number of females (27), students over age 21 (29), students with "a lot or extensive" prior programming experience (22) and few students from non-UK countries (23) it is possible that these results would be likely to vary with a more balanced population.

C1. Student enrolled in different degrees have no difference on opinions about the motivation in computer programming, the willingness to work hard and the ability have mastered programming by the exam.

According to Section 5.5.4, current evidence shows that there are no relationship between the courses they selected will affect their attitudes and beliefs (Table 5.41 – Table 5.45).

Based on the sample size and disproportionately low number of females and respondents over 21 years of age, it is unclear as to the representativeness of these results of the general population, but they may be more representative of first year programming student populations.

5.6.3 Summary

A short questionnaire was given to first year undergraduates who were learning to program as part of their course. The survey collected background information about each student's age, gender, origin (UK, EU or International), their course and their prior knowledge of programming as well as a number of questions related to how they viewed the programming course they were about to take. These questions concerned their motivation as well as a number of questions related to how hard or easy they would find the course and their confidence. The aim of the study is to examine the students' initial attitudes towards learning to program at university and then to examine any relationships between their attitudes and such factors as their background (e.g. age, gender and origin), the course they will take, their motivation and their prior experience of programming.

The analysis of the survey of the student responses has shown a variety of outcomes, which could be evidences for Research Question A:

A. Considering first year undergraduates: before being formally introduced to learning to program at university, do any differences exist, within specific sub–groups of students, between the attitudes to and beliefs about the task of learning to program?

When asked why they took the course, respondents with prior programming varied their answers, depending on the amount of experience they had. Those with little experience thought the course would be difficult (options C, D). Those with a lot of experience thought it would be easy (option B). Age was a determining factor for those who took the course because they were motivated to learn (option A). Whereas, based on Region of Origin they were most worried about the difficulty of the course. The five most significant results describing the key motivation and attitude questions, based on categorical background information, are marked ***** in the following table:

	STATEMENT				
	A	**B**	**C**	**D**	**E**
Age	*				
Gender					
Origin				*	
Course					
Prior experience		*	*	*	

Table 5.51 Reasons for Taking the Course by Demographics

Key to statements:

A. I feel motivated to learn to program.
B. I think learning to program will be easy for me.
C. I think I will have to work hard to learn to program.
D. I'm worried that learning to program will be difficult.
E. I’m sure I will have mastered programming by the exam.

6 Weekly Survey

The previous chapter described the preparation and design of the initial survey and how survey results varied by demographic variables and group characteristics. The data showed that attitudes and motivations about learning computer programming varied by group categories. This chapter describes how those measurements were monitored on a weekly basis using a participant survey. This short online survey was administered between October 2012 and December 2012 in students' weekly practical labs, as a means to trace the students' learning progress. The survey was designed to investigate responses relevant to the second research question (Section 3.1.3 & 3.4) which was:

B. Whilst learning to program in this connected world, to what extent do different students:
    a. Gain or lose confidence and motivation?
    b. Modify their approaches to learning as they progress?

As students' progress through the programme, there are dynamic changes the occur involving new material, personal circumstances and conditions beyond the student's control that affect motivation levels and feelings of competency and satisfaction. The weekly survey was designed to track these changes.

In order to maximise the survey response rate, questions were designed to be concise and easy to answer. Students were only required to simply tick any response that applied.

Having analysed the responses, it could then be summarized that their most challenging topics are "Testing and Debugging", "Array and API" and "Overloading".

6.1 Design

Based on the general design as discussed in Section 3.1, a paper-form questionnaire with closed-ended questions was the preferred tool to use in this survey as well, because it the most efficient way to get the most information the easiest and quickest.

6.1.1 Weekly Observation

Observation plays an essential role in education research. Scientific observation is an important channel to obtain and accumulate perceptual materials. Information teaching activities include interaction between teachers and students, collaborative activities between students and human-computer interaction. It encompasses not only the interaction activities between teachers and students, but also the external behaviour and appearance exhibited by teachers and students using information technology skills.

In this study, observation could offer investigators subjective impressions. It is a great opportunities to observe what students do (Section 2.1.5) in their weekly practical programming labs, aiming to see how they modify their approaches to learning as they progress (Research Question B.b). It could also be helpful to design the following surveys. It is suggested that the investigators could make weekly log to record observation. Meanwhile, it will also be helpful to ask for demonstrators and lecturers about what they have observed to broaden the view.

However, observation can't provide comprehensive learning behaviours from our students. Thus, weekly survey is still necessary.

6.1.2 Weekly Survey

Since students tend to be very busy in weekly labs, short questionnaire with closed-ended questions is suggested. Students would be more willing to provide their thoughts if they can do this by just simply ticking from options for a small amount of times.

Students can fill in this online questionnaire on their own will. Meanwhile, they will still be told that they are free to withdraw anytime they want if they feel uncomfortable for some questions. The responses should be able to support the evidence for Research Question A & B.a. In details,

- **What is your feeling about today's lab?**

This question allows students to select from *"Hard to begin with", "Easy", "Mixed easy & difficult", "Difficult", "Easier than I expected", "Harder than I expected", "Interesting"* and *"Challenging".* They are also able to provide not-listed feelings in "*other*". The responses should be able to show how their attitudes towards each lab and identify the most difficult lab/topic for students, which might be useful to gather more in-depth data in future experiments.

- **Until now, how do you feel about Java programming?**

Students will be asked to tick from "*Love it! :-D*", "*Like it! :-)*", "*Ho Hum… :-(*" and "*Don't like it. :-((*". Such descriptions and emoticons can promote participants' willing and encourage more responses. These responses could offer how students' attitudes change whilst they learning progress.

- **Based on your progress so far, what grade do you think you can get in the final examination?**

The options for this question are *"First Class (≥70%)", "2.1 (60%-69%)", "2.2 (50%-59%)", "Pass (40%-49%)"* and *"Fail (<40%)".* This question allows investigator to figure out how their final expectation change during the module.

The final questionnaire used in this phase of the study can be found in Appendix C.

6.2 Preparation and Implementation

The respondents were the first year undergraduate students who took COMP1004 (Programming Principles) in AY 2012-13 in the University of Southampton.

6.2.1 Weekly Observation

Prior to the observation, the following technical preparations must be performed:

6.2.1.1 Determine the content of the observation

The main purpose of observation is to record response variables resulting from experimental variables and significant behavioural changes. However, the study themes and the content to observe and record are always different. This study focuses on how students behave while learning to program in their weekly practical labs, including how they learn online and with textbooks, what they do when they encounter difficulties and how they communicate with their classmates, demonstrators and lecturers.

6.2.1.2 Determine the scene of observation

The investigator acts as a post-graduate teaching assistant while observing how students learn to program in their weekly practical labs for COMP1004 (Programming Principles).

6.2.1.3 Prepare observation instruments

Modern observation instruments mainly include recorders, optical cameras, digital cameras, television cameras, video recorders, closed-circuit television devices, and image and sound processing multimedia computers. Prior to the observation, it is not only necessary to prepare the required equipment, but also to check their availability, understand their properties, master their methods of operation and ensure their accuracy. For this study, the observer selected pen and paper and a sound recorder to take notes about certain behaviours.

6.2.1.4 Design observation log

An observation study must first observe and record. Subsequently, the results of observation must be organised, including statistics and text processing, in order to make the material system more accurate, essential and natural in preparation for further analysis and research.

The design of observation record sheet must be concise, scientific, structured and easy to operate. Prior to conducting the observation, the sheet must be check carefully for possible errors. With a detailed and complete sheet, the observer is able to think while observing, as it means that the observer only needs to fill in numbers and symbols, thereby making appropriate and detailed records easier during observation.

For this study, the weekly observation record sheet was designed as following (Figure 6.1):

Weekly Observation Record

Time		Lab #		Attendent rate	

Content	

Observing Item		Description
Main Confusion		
Reasons for confusion		
Learning Behaviours	Novices	
	Pre-experienced	
How to get help	Novices	
	Pre-experienced	
Comments & Analysis		

Figure 6.1 Weekly Observation Record Sheet

Whilst demonstrating in weekly practical labs, the observer takes notes in the printed record sheets with prepared pens. If there are too much information to write down but the time is limited, the observer could use a sound recorder to record students' behaviours first and transfer those voice into notes on the sheet after the lab. Then the observer will process all the information and create a new sheet with well-organized notes if necessary.

However, due to some marking labs, the observer will be busy with marking and giving feedback to students' lab work. In this case, although the observer will have no time to create any hand-writing notes, there might be some short sound recordings as observation data.

6.2.2 Weekly Survey

The online questionnaire will be able to give out participants on Google docs weekly from Friday 5th October 2012, whilst students were signing out at the end of each week's practical lab. The questionnaire shall remain open until the end of this module. The data and information collected from the questionnaire shall be kept until the 1st September 2013 when it will be destroyed.

6.3 Responses

6.3.1 Weekly Observation

There are ten completed weekly observation record sheets in all and the following (Figure 6.2) is an example:

Weekly Observation Record

Time	05/10/2012	Lab #	1	Attendent rate	≈100%

Content	Intro, compiling, variables; Object comparison & autoboxing

Observing Item		Description
Main Confusion		>40 students → small problems: 1. >80% questions: language: e.g. naming principle 2. Some novices: structure of the code
Reasons for confusion		Not familiar with how Java works. fundamental issues: e.g. how to write, compile, run
Learning Behaviours	Novices	≈30% attendees: 1. Reading set book/notes/slides for reference 2. "copying from examples" e.g. follow the spec closely
	Pre-experienced	≈10 student left early Talking to them → lab was easy finished in a short time
How to get help	Novices	1. Prefer: printed/online slides, printed/personal notes, textbook 2. Online tutorials (they found themselves) 3. Experts (e.g. PGTA, pre-exp-stu) 4. Little use of Google
	Pre-experienced	≈20 students: 1. Easy lab, very few questions 2. Only a few → used Google
Comments & Analysis		1. "copying from drawings" e.g. follow the spec 2. Surface learning (novices). Deep learning? little evidence 3. No one gave up. Syntax → ok 4. Not so much Web → don't know what to search 5. Highly motivated when problem solved

Figure 6.2 Example of Weekly Observation Record Sheet

Having observed by the writer and discussed with other teaching assistants, the observation data could be organized in a big table (Appendix E).

To summarize, the weekly observation data could be generated as:

- Lab 1 - 3

Due to the similar learning behaviours, the observation results from these three labs could be summarized together. Lab 1 to 3 explained basic concepts of Java, including compiling, variables, classes, if-statement, methods, object interactions, etc. Most of the students had a good start. The specification/instructions were clear, so basically, the questions from students focused on the concepts. Key errors/confusions were centred in detailed problems, like naming principles. Some students were confused with logics. In this stage, syntax was not a big problem, because some students, especially novices, were struggling with the structure of the code and how Java works.

In terms of their learning behaviours, pre-experienced students finished this lab and left much earlier than novices. There were various ways for novices to get help when struggling: 1) they firstly reviewed the lecture slides, textbooks and notes. They did not search (e.g. Google) much, but some students were watching online tutorials (e.g. live programming demonstrating videos); 2) they asked teaching assistants and pre-experienced students frequently. They preferred helpers to explain Java using metaphor, because in this stage, programming might be too abstract for novices and real world practices would be helpful for them to understand; and 3) few novices asked lecturers for help. Most of the time, they would not talk to the lecturers unless the lecturers asked whether they were OK or not first.

Strugglers only achieved surface learning at this stage. They have not built up a complete understanding about how Java works, and are more likely simply copying from drawings, without understanding the in-depth reason why. Nonetheless, none of the students gave up, even if they got stuck. They had to be willing to ask, and although it might take a long time for novices to complete this lab, as long as they succeeded, they felt happy and highly motivated.

- Lab 4

This lab practised loops and array list. This is the first marking lab, so teaching assistants were busy with marking the previous three labs. In this case, strugglers might have to ask their classmates for help and search online for help.

- Lab 5

This lab was mainly about arrays, HashSet, HashMap and relevant APIs. It was a very busy lab, because: 1) due to the marking, many students had to leave their questions until this week; 2) it might be a challenging topic for students and not easy for some novices to understand new theories in a shorter time; and 3) the specification was not as detailed as previous labs. At this stage, students were supposed to do some of their own computing thinking. Apparently, some students were not ready for this change.

Since students received their first formal feedback for their work the prior week, they started to know that surface learning only was probably not enough to succeed in the learning to program course any more. Questions from students were becoming more in-depth, which showed that they started to adopt deep learning approaches to understand programming. Most of the students followed the feedback well, so good feedback was considered to be very important, which might lead students to adopt better learning approaches and activities.

It is reflected that the feedback provided from last lab affect students' learning approaches, especially for novices and students who have a little programming experience. Those students still confused about what is the "best" way for them to learn to program, so they will follow the feedback strictly.

- Lab 6

This lab introduced how to write test harnesses and how to debug with IDEs. The key issue for this lab was that, many students were confused about the specification and how to create test scripts, which reflected that students needed to improve their computing thinking.

In this week, students said that this lab was hard to begin and they tended to ask people for help rather than searching online, because this was their first time to use debugger and they had no idea about what to search. Thus, they preferred experts who could explain relevant theories from the beginning in order to finish the lab work. However, once they obtained a basic understanding about debugging, they used the Web first if they help needed again.

The students may have other deadlines from other modules. This might cause some problem, but did not affect their programming learning.

- Lab 7

Object-oriented theories were introduced to the students beginning this week. In this lab, students were asked to practise inheritance, method overwriting, polymorphism and expectations.

However, this was also the second marking week, so all the teaching assistants were busy with marking and giving feedback to the students' previous lab work (lab 4 - 6). Thus, if the students were getting stuck, they had to ask their classmates or search on the Web. After the lab, some students commented that the specification was easy to follow, so it was easier than expected for them to finish the code.

- Lab 8

This lab introduced method overloading and interfaces. It was busy as expected because some students left their questions for this week due to the marking.

Most of the students who got confused about this lab had a surface understanding about basic object-oriented programming theories. They later concluded that the specification of lab 7 was really clear. They sometimes provided the correct code, but without taking a deep learning method to obtain a strong understanding about the theory. Teaching assistants still needed to remind the students to learn in depth.

- Lab 9

This lab practised reading from external files, modelling loaded data and writing files. Teaching assistants were all busy answering questions from students, but the questions were mainly about lab 8 due to the delay of marking lab in week 7.

At this stage of programming learning, most students have already obtained a sound understanding of programming. If they got stuck, most of them would go online and search for help. Sometimes they may know what to code but not clear about the detailed reason why, and they would ask their friends and teaching assistants to sort out these issues. Most of the time, teaching assistants would lead these students to adopt a deep learning approach, in order to gather a perfect understanding.

- Lab 10

There were no designed tasks in this lab. Apart from marking lab 7 to lab 9, teaching assistants would answer any questions about this module, including all previous labs and coursework. Some strugglers reflected that, in order to get pass for their exam, they gave up advanced OOP theories but enhanced basic theories.

In addition, it is worth mentioning that, many students may use online group discussion (e.g. Facebook group) for some modules, but not for programming. Some students reflected that it was difficult to describe programming problem online by typing. It was much easier to describe their problems by online voice chatting and face-to-face discussion.

6.3.2 Weekly Survey

These surveys were taken at the end of each week's lab. The number in the brackets is the actual number of students who responded to this attitude question. The N values indicate the number who selected that particular option and the percentage indicates what proportion of those who completed this question selected that option. The final questionnaires can be found in Appendix D.

- **Attitudes towards Practical Labs**

	Easy		**Difficult**		**Mixed easy and difficult**		**Hard to begin with**	
	N	**%**	**N**	**%**	**N**	**%**	**N**	**%**
Week 1 (119)	39	32.8	9	7.6	33	27.7	22	18.5
Week 2 (122)	31	25.4	6	4.9	47	38.5	18	14.8
Week 3 (117)	27	23.1	20	17.1	27	23.1	26	22.2
Week 4 (120)	23	19.2	21	17.5	44	36.7	13	10.8
Week 5 (115)	16	13.9	29	25.2	41	35.7	19	16.5
Week 6 (116)	14	12.1	29	25.0	35	30.2	20	17.2
Week 7 (112)	32	28.6	10	8.9	36	32.1	6	5.4
Week 8 (104)	29	27.9	5	4.8	43	41.3	6	5.8
Week 9 (103)	22	21.4	11	10.7	36	35.0	9	8.7
Week 10 (100)	27	27.0	12	12.0	13	13.0	6	6.0

Table 6.1 Weekly Evaluation of Lab Difficulty

	Easier than expected		Harder than expected		Interesting		Challenging	
	N	%	N	%	N	%	N	%
Week 1 (119)	15	12.6	5	4.2	46	38.7	20	16.8
Week 2 (122)	12	9.8	5	4.1	37	30.3	21	17.2
Week 3 (117)	5	4.3	23	19.7	46	39.3	29	24.8
Week 4 (120)	8	6.7	13	10.8	36	30.0	27	22.5
Week 5 (115)	5	4.3	16	13.9	39	33.9	33	28.7
Week 6 (116)	3	2.6	20	17.2	33	28.4	33	28.4
Week 7 (112)	25	22.3	2	1.8	42	37.5	20	17.9
Week 8 (104)	12	11.5	4	3.8	31	29.8	19	18.3
Week 9 (103)	9	8.7	7	6.8	34	33.0	20	19.4
Week 10 (100)	5	5.0	6	6.0	19	19.0	11	11.0

Table 6.2 Weekly Attitudes Toward Labs

- **Attitude towards Java**

	Love it		Like it		Ho Hum		Don't like it	
	N	%	N	%	N	%	N	%
Week 1 (119)	37	31.1	66	55.5	9	7.6	7	5.9
Week 2 (122)	40	34.8	57	49.6	13	11.3	5	4.3
Week 3 (113)	28	24.8	57	50.4	20	17.7	8	7.1
Week 4 (116)	32	27.6	67	57.8	13	11.2	4	3.4
Week 5 (108)	28	25.9	56	51.9	19	17.6	5	4.6
Week 6 (110)	24	20.7	64	55.2	16	14.5	6	5.5
Week 7 (108)	38	35.2	56	51.9	10	9.3	4	3.7
Week 8 (97)	32	33.0	48	49.5	13	13.4	4	4.1
Week 9 (98)	35	35.7	50	51.1	10	10.2	3	3.1
Week 10 (94)	37	39.4	45	47.9	6	6.4	6	6.4

Table 6.3 Weekly Attitudes Toward Java

- **Beliefs in Progress**

How students perceived their progress on the course was measured by the grade they expected to achieve at the end of the course depending on how they felt at the end of each class. The results are shown in the table below.

	First		2.1		2.2		Pass		Fail	
	N	%	N	%	N	%	N	%	N	%
Week 1 (119)	46	38.7	55	46.2	9	7.6	6	5.0	3	2.5
Week 2 (115)	42	36.5	51	44.3	13	11.3	5	4.3	4	3.5
Week 3 (112)	37	33.0	50	44.6	9	8.0	11	9.8	5	4.5
Week 4 (112)	39	34.8	54	48.2	10	8.9	6	5.4	3	2.7
Week 5 (109)	34	31.2	52	47.7	12	11.0	8	7.3	3	2.8
Week 6 (109)	33	30.3	50	45.9	15	13.8	8	7.3	3	2.8
Week 7 (108)	45	41.7	48	44.4	7	6.5	6	5.6	2	1.9
Week 8 (98)	32	32.7	52	53.1	5	5.1	6	6.1	3	3.1
Week 9 (98)	32	32.7	47	48.0	12	12.2	5	5.1	2	2.0
Week 10 (96)	31	32.3	44	45.8	11	11.5	5	5.2	5	5.2

Table 6.4 Weekly Assessments Final Expectation

6.4 Data Analysis

6.4.1 Weekly Observation

The ebb and flow of student perceptions of their progress, or the lack thereof, is to be expected over a period of weeks. The course materials and designed experiences vary from week to week and the students' prior knowledge and experience make each week's learning objectives more or less difficult depending on individual capabilities. The data reflect this. The weekly ratings of relative ease or difficulty of that week's learning experience were inversely proportional. As perceived difficulty increased, the degree of perceived ease decreased and vice versa. Overall, their liking for Java increased as the course progressed. There was a small percentage who expressed their belief from the beginning that they would likely fail the course, whereas, those who expected a grade of 2.1 remained fairly stable throughout the entire course. Those who expected to receive a First only declined slightly from their first 2 weeks' ratings. These are all retrospective self-reports and, as such, are subject to a variety of influences that have almost nothing to do with actual performance in the learning experience. These measures are important, however, because they are able to show trends over a period of time and can provide insights into what factors may have caused variations over time.

6.4.2 Weekly Survey

The central focus of the study is related to how students' attitudes and beliefs may have changed over the ten week duration of the course. Analyses were made for student responses week by week on the three areas: attitudes about the lab session, attitudes about Java and their belief about their level of performance. In order to understand the pattern of change over the period of the ten week course, graphical means are employed.

As one might expect, the weekly ratings varied depending on as to the difficulty of that week's module content as perceived by the student. As the following data will show in greater detail, student mastery of course content seemed to increase until the week 7 module; thereafter, the pattern of responses on different variables was mixed. The average of "easy" ratings on were on average nearly 30% for the final 4 weeks while the mixture of "easy/difficulty" average was slightly higher. The sessions rated "hard to begin with" were at their lowest (5%) for the last 4 weeks. The variability suggests that some students were beginning to experience mastery while some students continued to struggle.

6.4.2.1 Attitudes towards Practical Labs

The proportion of students who indicated that the session was easy dropped consistently for the first 6 weeks, but the number jumped from 12.1 in week 6 to 28.6 in week 7.

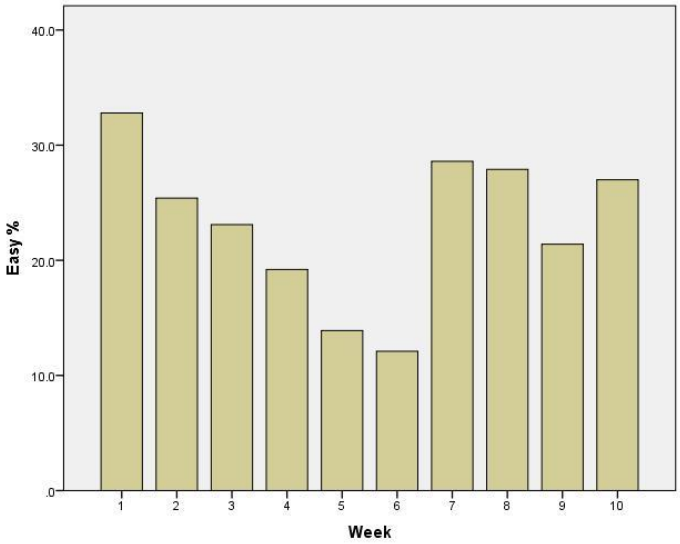

Figure 6.3 Weekly % of Sessions Rated Easy

The proportion who found the session difficult was relatively low in the first two weeks but jumped to over 17% in weeks 3 and 4 before rising again in weeks 5 and 6. However in weeks 7 and 8 the proportion dropped dramatically. In summary, it appears that the lab classes were found most difficult in the middle section of the course.

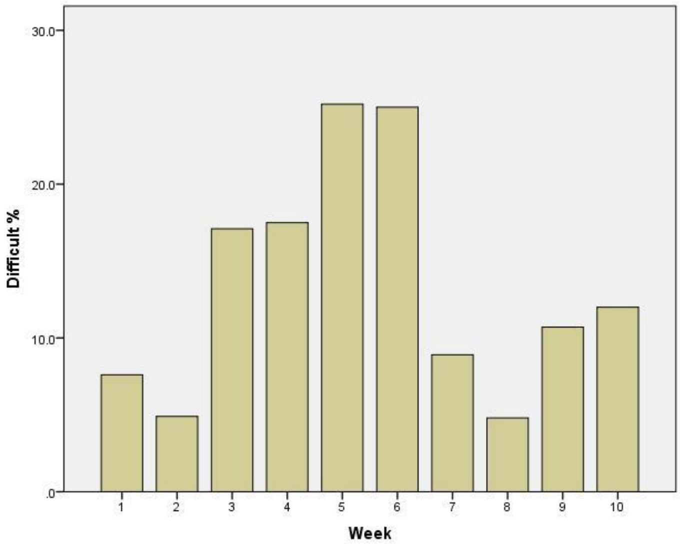

Figure 6.4 Weekly % of Sessions Rated Difficult

The proportion of students who found each class to be a mixture of easy and difficult, although fluctuating, did not show the clear patterns of the first two measures.

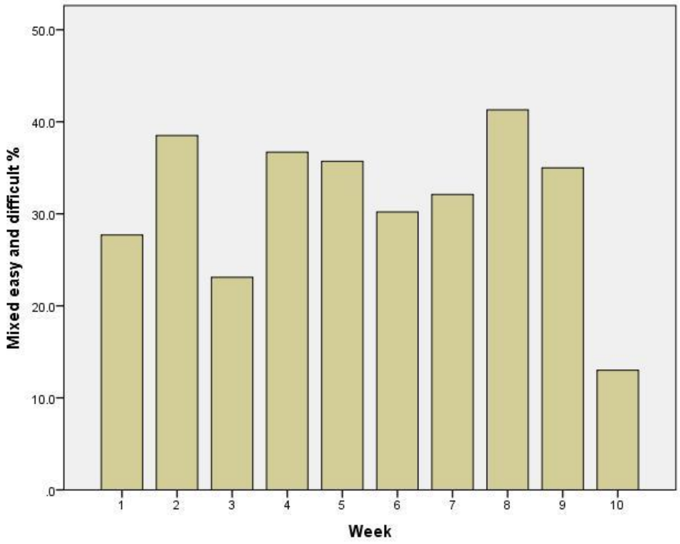

Figure 6.5 Weekly Sessions % Rated Easy/Difficult

The proportion who found the classes 'hard to begin with' varied from week to week but generally fell over the period of the course.

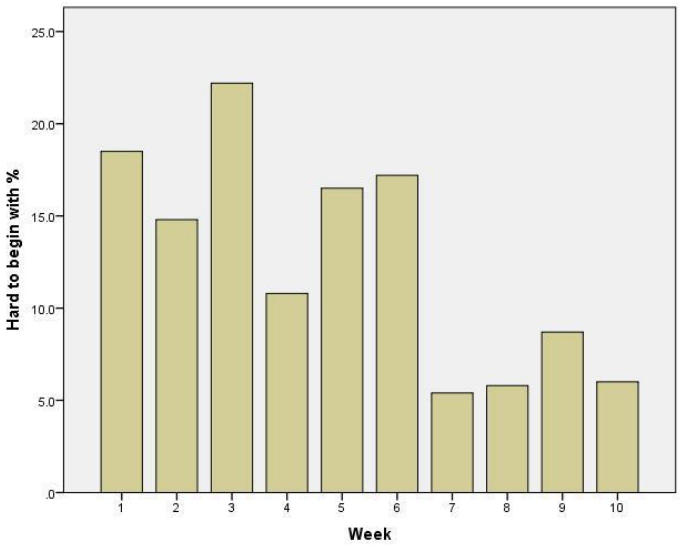

Figure 6.6 Weekly Sessions % Rated Hard to Begin With

The 'Easier than expected' measure falls over the first six weeks until in week 7 there is a sudden rise (from 2.6% to 22.3%) after which the downward trend resumes.

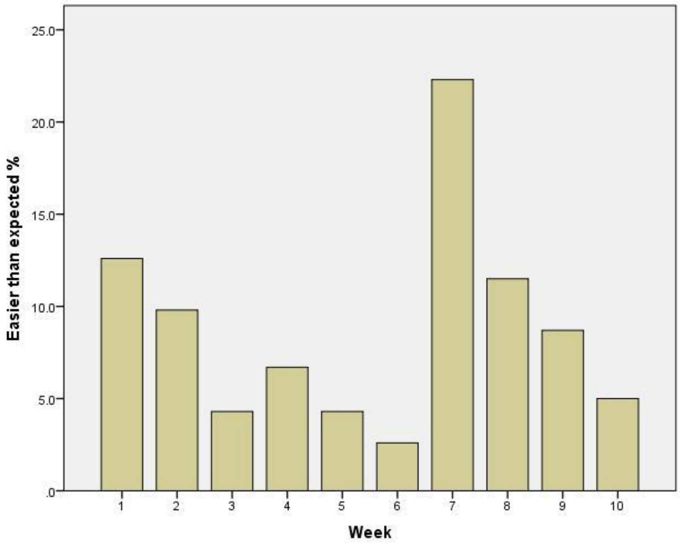

Figure 6.7 Weekly Sessions % Rated Easier Than Expected

The 'Harder than expected' measure jumps in week 3 to 19.7% and these higher values remain for the following four weeks. However, there is a dramatic drop in week 7 from 17.2% to 1.8%.

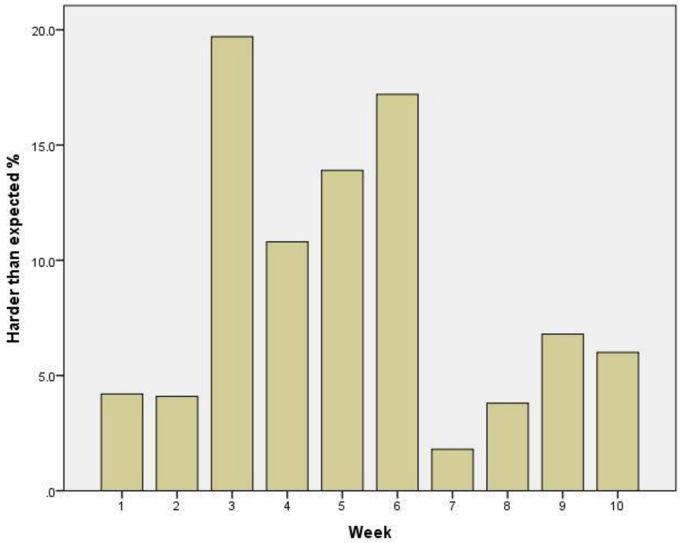

Figure 6.8 Weekly Sessions % Rated Harder Than Expected

The proportion of students who found the classes 'Interesting' varied from week to week but had an overall average of 32%. There was a drop in the last week to 19.0%.

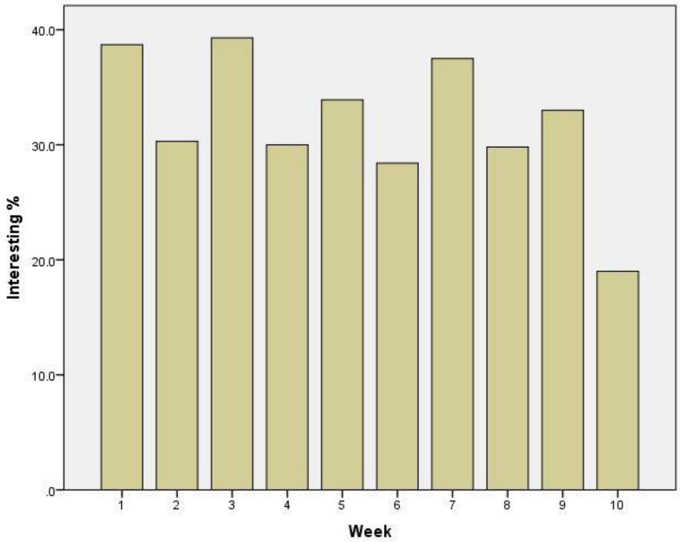

Figure 6.9 Weekly Sessions % Rated Interesting

The proportion of students who found the classes 'Challenging' peaked in weeks 2-6 and had an overall average of 21%. There was a drop in the last week to 11.0%.

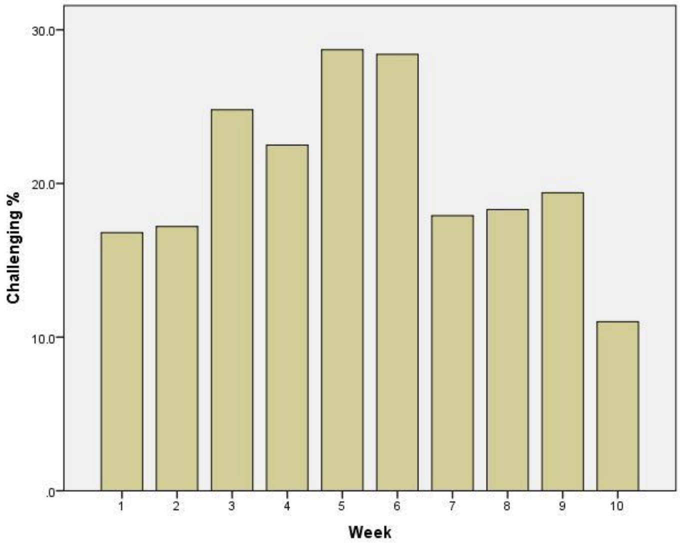

Figure 6.10 Weekly Sessions % Rated Challenging

To conclude, combined with the results of weekly observation, the top 3 difficult labs are: Lab 6 (Testing & Debugging), Lab 5 (Array & API) and Lab 8 (Overloading). However, there is no evidence so far to show the detailed reasons, which need to be explored in future experiments.

6.4.2.2 Attitude towards Java

Taking the responses overall the breakdown of attitudes to Java can be seen in this graph:

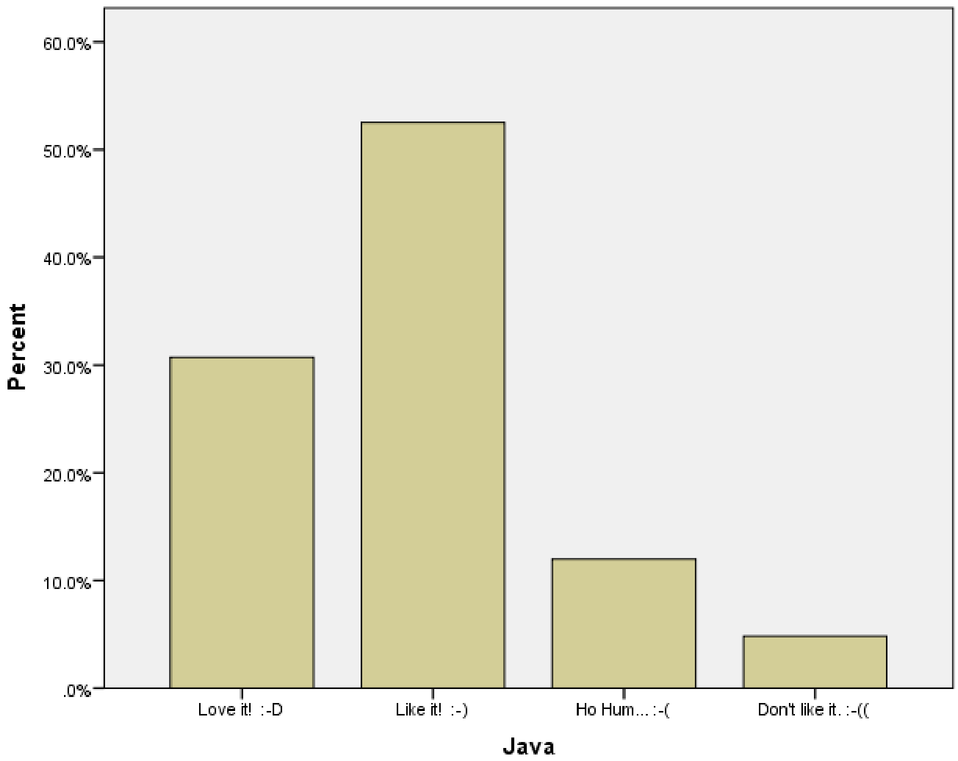

Figure 6.11 Overall Attitudes toward Java

However, as the values in the table show, attitudes change from week to week. In order to see any changes during the course the following stacked bar graph can be used. For each week, it shows the relative sizes of the four choices for the group.

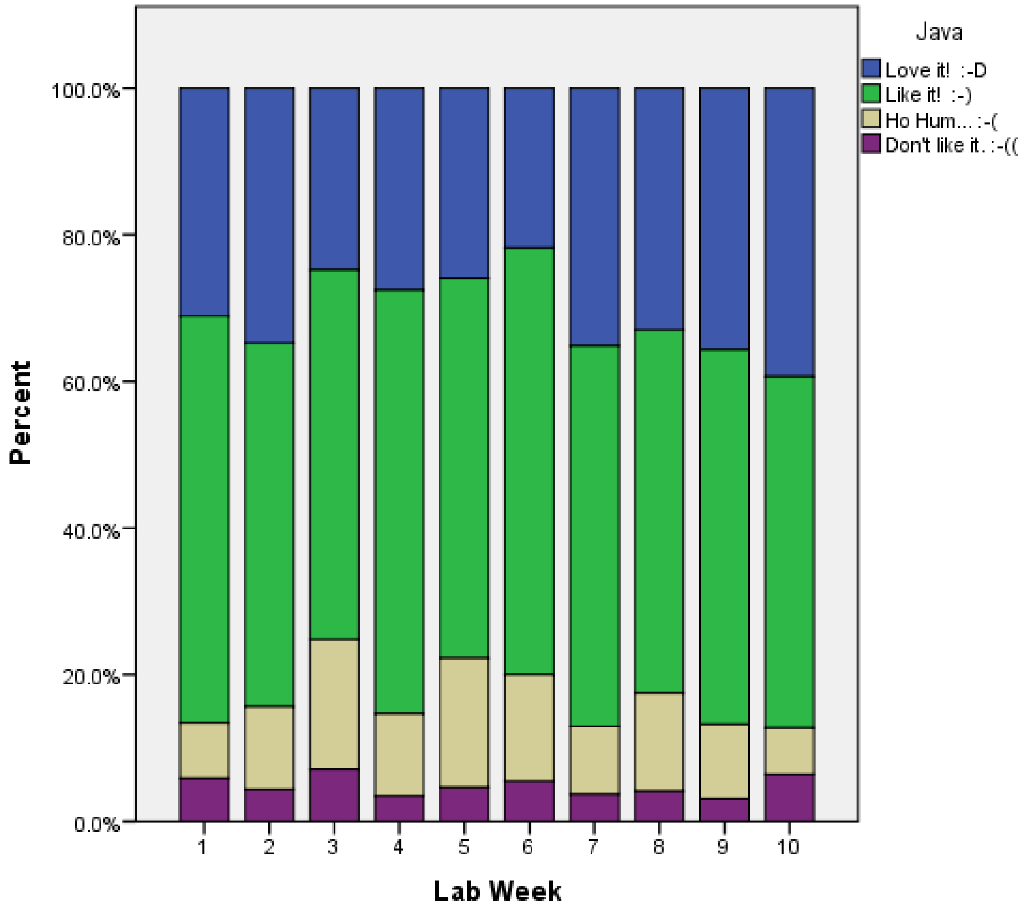

Figure 6.12 Attitudes toward Labs by Week

For each week we can see the proportion of the total number of students whose attitudes matched each of the four options. For each week the proportions are not very different from the overall proportion shown in the above bar chart. The great majority either said they 'Love it' or 'Like it'. The proportion who said they 'Love it' increased over the ten weeks with the proportion in Week 10 reaching nearly 40%.

The proportion who said they 'Don't like it' varied from a low of 3.1% in week 9 to a maximum of 7.1% in week 3.

6.4.2.3 Belief in Progress

Taking the responses overall the breakdown of attitudes to Java can be seen in this graph.

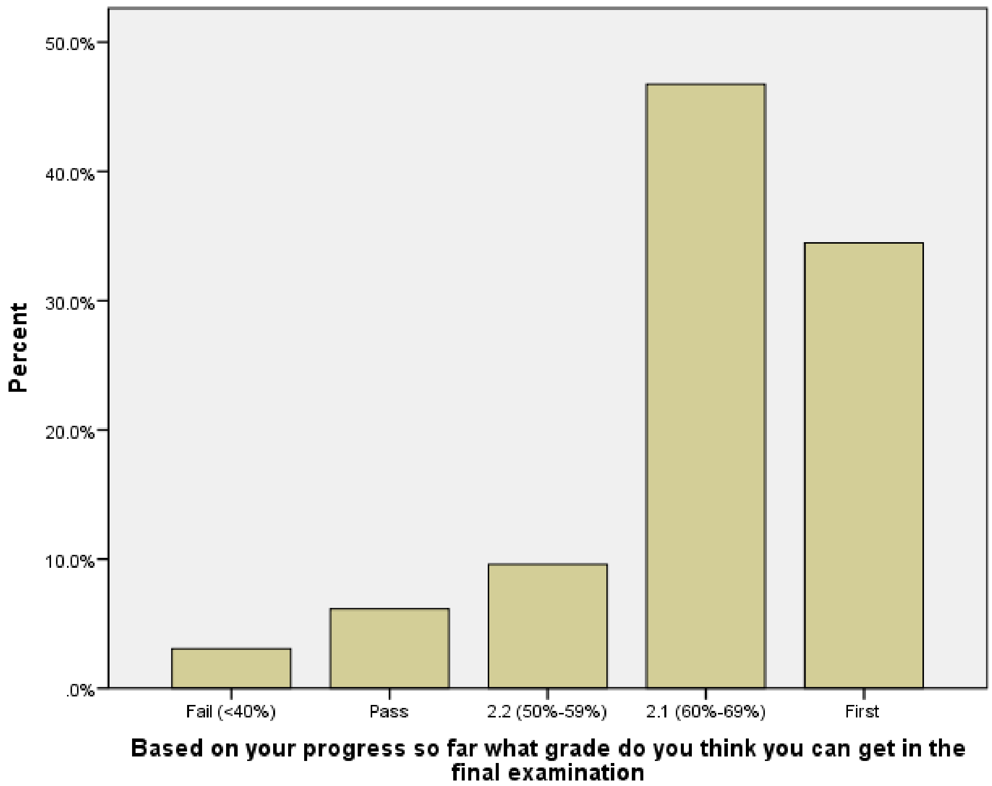

Figure 6.13 Final Exam Grade Expectations by Percent

As the values in the table show, belief in their progress changes from week to week. In order to see any changes during the course the following graph can be used.

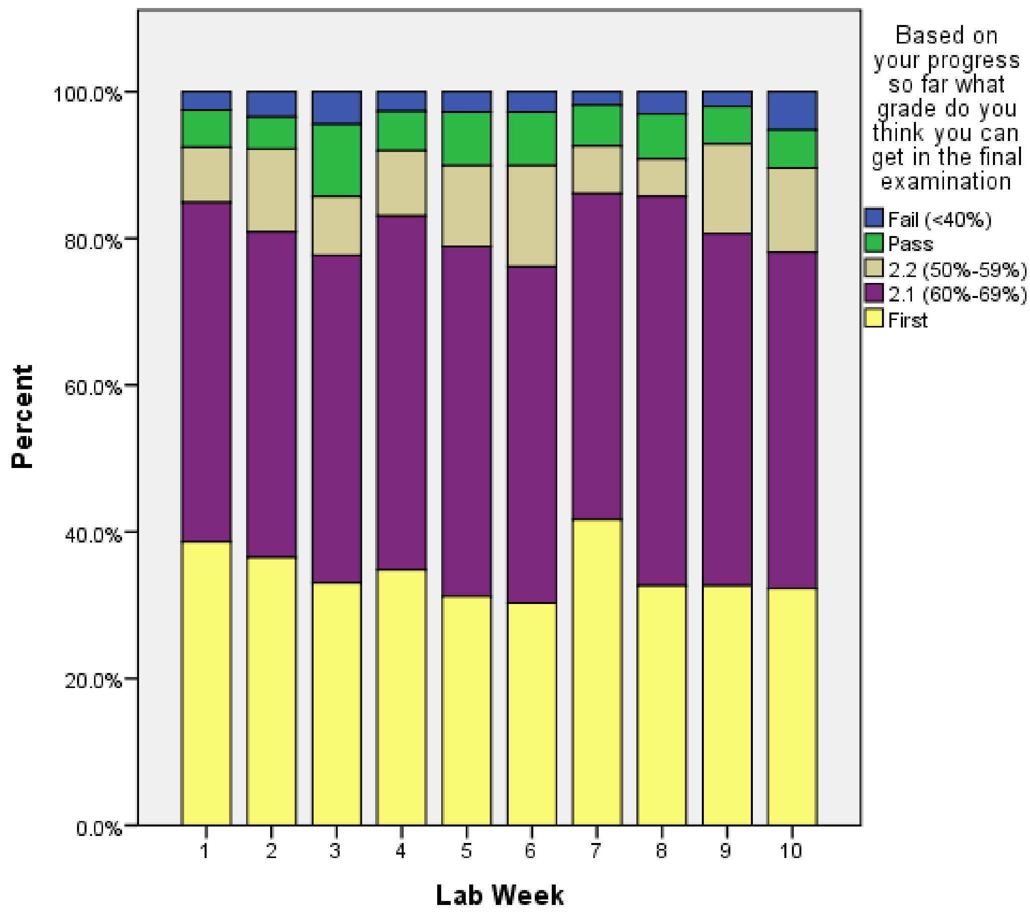

Figure 6.14 Final Exam Grade Expectations by Week

For each week the graph shows the proportion of the total number of students who believed that would obtain each grade in the five degree level options. For each week the proportions are not very different from the overall proportion shown in the earlier bar chart, the categories classified in order of decreasing size: 2.1, First, 2.2, Pass, Fail. The proportion who thought their work merited a First slowly declined over the first 6 weeks but in week 7 it was boosted from 30.3 to 41.7.

6.4.3 Conclusion

As one might expect, situations can change over a 10 week period and self-ratings of attitudes and beliefs about one's performance and expected outcomes in a computer lab are no exception. The amount of effort required to master assignments from one week to the next varied with the assignment, the skill set and the amount of effort required to complete assigned tasks. Students with lesser computer backgrounds obviously struggled harder than those who had prior programming experience. The written comments from the weekly survey, as well as the item ratings, that there were a variety of factors that affected their weekly performance ratings, depending on their degree of difficulty with the subject matter whose ratings plummeted mid-course (week 5 and 6). Despite the degrees of difficulty, the overall ratings of loving the course improved over the 10 week period. The weekly survey was an effective way to determine student reactions to the labs and reveal individual differences as well as the effect course content had on student attitudes.

6.5 Summary

The outcome from weekly observation and weekly survey could be evidence for Research Question B:

B. Whilst learning to program in this connected world, to what extent do different students:
   a. Gain or lose confidence and motivation?
   b. Modify their approaches to learning as they progress?

The survey recorded three main measures related to learning to program: attitudes about class sessions (eight measures), attitudes about Java (four options) and belief in one's progress (degree of grade). It can be seen that there are changes from week to week in each of these. All of the values recorded in the tables and graphs are summaries for the group of students. However very few of these are major changes - most are slight changes between weeks.

There are some exceptions and, in particular, it does appear that week 7 was appreciably different from other weeks. There were significant increases in measures of Easy, Easier than expected and Interesting with decreases in measures of Difficult, Hard to begin with, Harder than expected and Challenging. Also in week 7 it can be seen that there were increases in Love it and Like it with reference to Java, while the students' predictions of expected grades at the end of the course showed increases in First and 2.1.

What the overall patterns do not show are the individual experiences of the course and any interactions between student attitudes and beliefs. That is something that could be investigated in the future. For example, course designers may want to consider the order in which some modules are introduced into the course curriculum. It is logical to assume that some concepts must be mastered before introducing new, possibly more complex ones. However, some topics might be able to be introduced in a different sequence and obtain different, if not better, results. This could be a subject of future experimentation and a subsequent survey of previous students. Regardless, periodic data collection in close proximity to the events being measured provides a longitudinal record of perceptions about important components of the course continuum. Thus, the weekly assessments can contribute a phenomenological history of the course as it was experienced in real time and provide insights as to what might be done differently in the future and why.

7 End of Term Survey

The overall design of this research was to obtain qualitative and quantitative measurements of the study as it progressed. The measurements occurred before, during and after key components of the project were implemented. Previous chapters outlined the procedures and measurements that occurred with the pilot study and the student self-assessments that were made about their thoughts, feelings and attitudes throughout the class sessions and labs. Another significant milestone in measuring the effects of the course is the end of term survey that is described in this chapter.

The meta-goal of this study is to conduct and analyse how the Internet affects first-year undergraduate students learning to program in this connected world.

The outcome of this study could be evidence for Research Question B & C (Section 3.1.4 & 3.4):

B. Whilst learning to program in this connected world, to what extent do different students:
    a. Gain or lose confidence and motivation?
    b. Modify their approaches to learning as they progress?

C. How do students integrate their formal and informal learning activities? To what extent do students:
    a. Rely on formally provided materials and exercises?
    b. Develop individual approaches to learning which integrate real world and online activities and materials?

There are 46 completed responses from an online questionnaire survey conducted to investigate to what extend the Web impacts on the students' learning to program once the students have finished the introductory module.

The most popular online platform was a public search engine, while the formally provided learning environment comes second. Participants tend to use the Web the most when they were introduced to a new theory or looking for help when they got stuck. In terms of the reason, participants believe that the Web is convenient to use, and it is the best place to find information.

Diagrams with discussion will be described, as well as the plan of conducting a further NGT meeting.

7.1 Design

This survey was principally directed toward use of the internet as a tool for learning. This addressed one of the primary aims of this study which was to determine the role the internet played in supporting the learning process as a supplement or substitute to the materials and resources that were provided with the course materials.

Students tend to be busy with their final examinations at the end of a semester. Thus, a short online questionnaire mainly with closed-ended questions is preferred in this stage to collect some data, which could identify what students do with the Internet for learning to program, and provide "threads" for future experiments to answer the research questions.

The details of the questions are,

- **Do you have any programming experience before you started this degree?**

According to the responses from similar question in the Initial Survey (Section 5.1), students tend to be confused about how to distinguish their prior programming experience level between "a lot" and "expert". As a result, the options for this question will be designed as "*None*", "*Some*" and "*A lot*". The responses could be helpful to figure out whether prior experience is a factor which would affect students' learning behaviours with the Internet.

- **When do you use the Internet when learning to program?**

The options for this question are "*None*", "*Before lectures/labs*", "*During lectures/labs*" and "*After lectures/labs*". From the responses, academics may obtain a general view that when is the best time to offer online learning help and maybe offer suitable kinds of learning resources for each particular timing.

- **What online platform(s) do you use?**

The respondents are supposed to select any that apply from the following list:

- ☐ None
- ☐ Formally provided ones, e.g. ECS intranet, Edshare, student wiki
- ☐ Public searching engine, e.g. Google, Bing.
- ☐ Online live programming videos, e.g. watching tutorial videos in YouTube
- ☐ Social networking, e.g. Facebook.
- ☐ Forum/BBS
- ☐ Online courses
- ☐ Others, please specify: ________________________

The responses should offer an overview about what favourite online platforms are. Academics could try to enhance those platforms if possible, to promote students' learning.

- **How do you use the Internet?**

The options are:

- ☐ None
- ☐ Learning a new theory involved in the lectures, e.g. inheritance, debug, array.
- ☐ Learning extra theories, e.g. Enum.
- ☐ Review
- ☐ Getting some help when struggling for formal provided labs/coursework
- ☐ Others, please specify:________________________

The responses could remind academics what kind of learning materials which students may prefer.

- **Why do you choose the Internet to help you learn to program?**

This question is designed to identify why students like to use the Internet nowadays. The potential reasons might be:

- I never use the Internet. Please specify the reason.
- I don't have other choices.
- Convenient.
- I don't have to rely on other resources, e.g. textbook.
- Try to avoid asking "silly" questions to other people, especially the lecturers, because it might make them consider me to be "stupid" and not hard-working.
- Others, please specify: ____________________________.

- **Do you have any suggestions on how this module could be enhanced by adopting the Internet (probably in terms of improving the teaching strategies, providing more online resources or platforms to learn and giving feedback, etc)?**

This is an only open-ended question (Section 3.2.1) which allows the respondents to provide their attitudes towards how the Internet affects their learning which are not list in above questions.

The final questionnaire used in this phase of the study can be found in Appendix D.

7.2 Preparation and Implementation

The respondents were the first year undergraduate students who took COMP1202 (Programming I) in AY 2012-13 in the University of Southampton. Participants were not specifically chosen and the questionnaire was publicised widely to students across the university by the researchers and via word of mouth. If recruitment had not been sufficient, the investigator was prepared to ask others to help spread the survey through word of mouth. Students were not forced into taking part in the study; they took part at their own free will. No incentives were offered.

The online questionnaire was visible to participants from 11th December 2012. The questionnaire remained open until 15th January 2013. The data and information collected from the questionnaire were kept until the 25th January 2013 after which it was destroyed.

7.3 Responses

The students were asked if they had experience of computer programming prior to the course. The responses were:

None 12 (26.1%), Some 25 (54.3%), A lot 9 (19.6%)

They were then asked on what specified period (before/during/after lectures/labs) they used the Web when learning to program. In terms of the number of occasions, the values ranged from never to as many as three possible times offered.

Number of specified periods use the Web when learning to program	Frequency	%
0	1	2.2
1	12	26.1
2	17	37.0
3	16	34.8
Total	46	100.0

Table 7.1 Frequency of Internet Use

This indicates a very high use of the Web with 71% using it on at least two occasions. The one person who didn't use the Web at all gave the reason: '*Because I can't understand what is being said on the website, forums or any other sites*'.

The numbers and percentages of those using the Web on different occasions broke down as follows:

- Before lectures/labs 28 (60.9%)
- During lectures/labs 30 (65.2%)
- After lectures/labs 36 (78.3%)

The students were asked **What online platform(s) do you use when learning to program?** The numbers of different platforms used are shown in the graph.

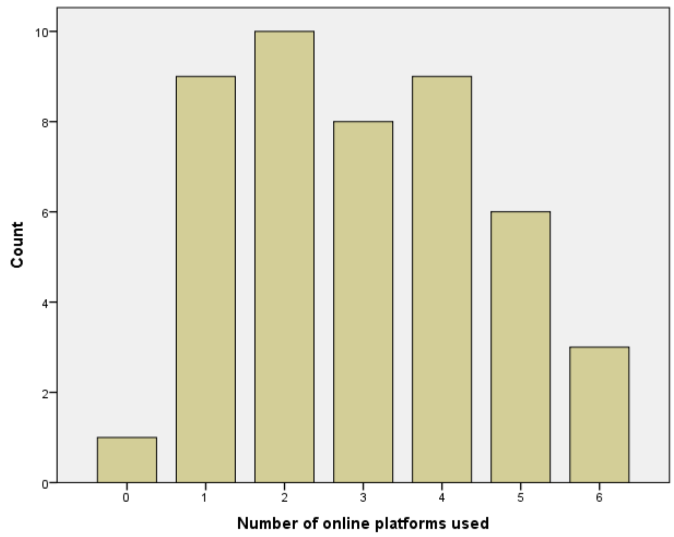

Figure 7.1 Number of Platforms Used to Access Internet

This illustrates that although many students use only one or two platforms (approximately 40%), there are also many who use up to six different platforms.

The popularity of each platform measured by the numbers using each type can be seen in the following table. The list has been arranged in order of popularity.

Platforms	Number	Percentage
Public searching engine, e.g. Google, Bing.	42	91.3%
Formally provided ones, e.g. ECS intranet, Edshare, student wiki	25	54.3%
Forum/BBS	19	41.3%
Social networking, e.g. Facebook.	17	37.0%
Online live programming videos, e.g. watching tutorial videos in YouTube	15	32.6%
Online courses	15	32.6%
Others	2	4.3%

Table 7.2 Platforms Used to Access the Internet

Although public search engines lead the group with over 90% using them and over half using the formally provided ones, there is also a substantial use made of all the other platforms. The examples provided in the 'Others' response were:

- *StackOverflowSlideShare (or similar) for different topics (programming related, but not only java or what is in class/lectures), and,*
- *API*

The students were then asked **How do you use the Web when learning to program?** The numbers of different uses made by the students are shown in the graph.

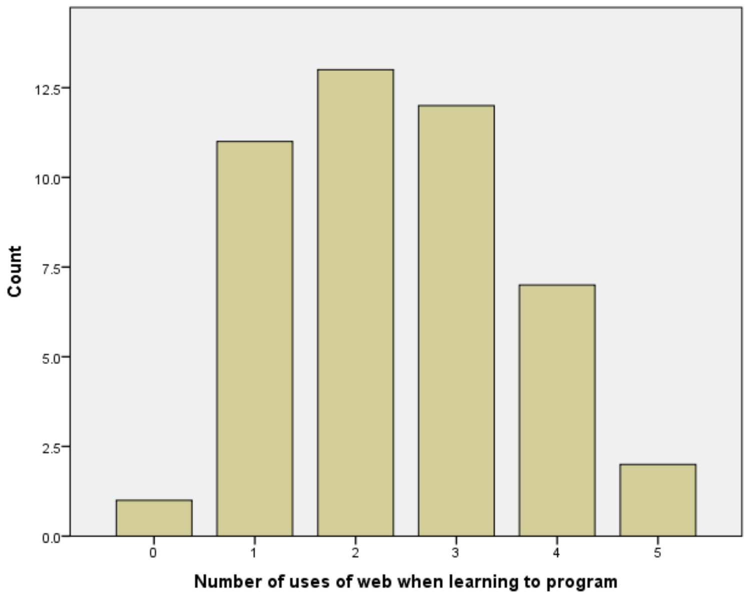

Figure 7.2 Uses of the Web When Learning to Program

It is clear that many students make use of the Web for a variety of applications with some using it for up to 5 different purposes. The one person who had zero uses was the student who earlier said they never used the Web.

The popularity of each possible use of the Web can be seen in the following table (in order of popularity).

Use of the Web	Number	Percentage
Getting some help when struggling for formal provided labs/coursework	29	63.0
Learning a new theory involved in the lectures, e.g. inheritance, debug, array.	28	60.9
Learning extra theories, e.g. Enum.	27	58.7
Revision	18	39.1
Others	9	19.6

Table 7.3 Reasons for Using the Web to Learn Programming

Clearly, the first three uses are very close together in their level of popularity. It should be noted that these choices applied throughout the course, whereas revision was in preparation for examinations.

The examples given in 'Others ' were:

- *Extending knowledge of the theory involved in the lectures,*
- *I use it as a reference guide,*
- *Learning things related to my own personal programming projects,*
- *Learning how to use the covered material to fulfil my current needs,*
- *Sites like stack overflow for complex bugfixes,*
- *Checking syntax and api methods,*
- *Looking up Java syntax, as I have programmed in other languages except java before,*
- *I'm at the point when I think I know how to program and I can pick up a new programming language fairly easy - I usually read/see things about good software design and what's the "cool" way to solve some problems in different situations,*
- *Lots of time reading forums :P.*

The students were next asked **Why do you choose the Web when learning to program?** The numbers of different reasons given are shown in the graph.

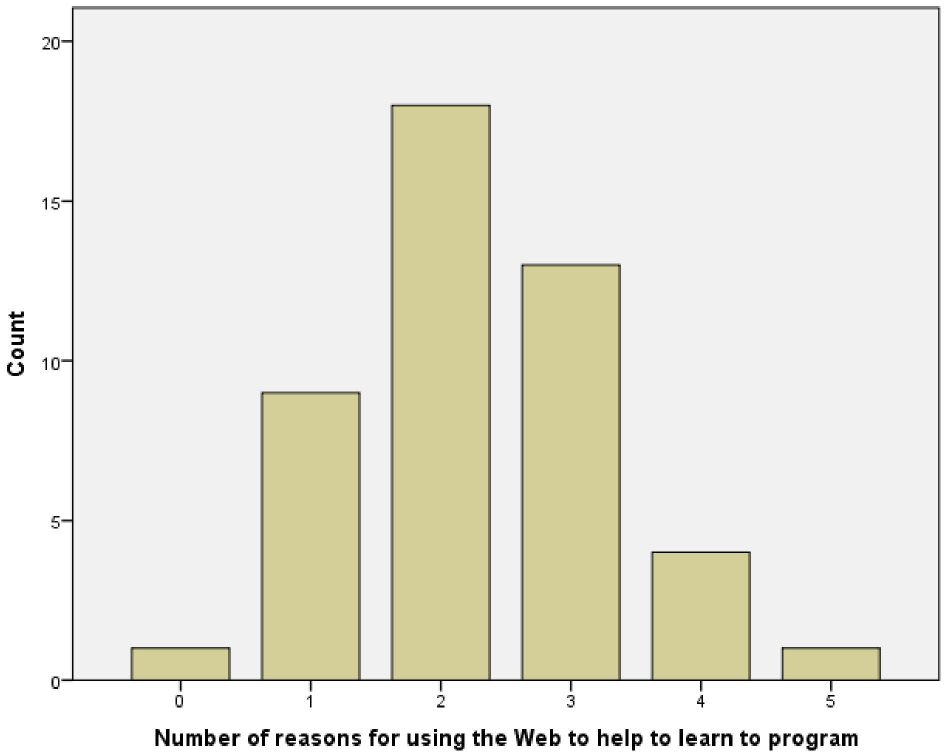

Figure 7.3 Number of Reasons Given for Using the Web

It is clear that many students have a variety of reasons for using of the Web with 36 (nearly 80%) giving two or more reasons.

The popularity of each possible reason for using the Web can be seen in the following table (in order of popularity).

Reasons for choosing the Web to help to learn to program	Number	Percentage
Convenient.	43	93.5%
It's the best place to find information.	31	67.4%
I don't have to rely on other resources, e.g. textbook.	14	30.4%
Try to avoid asking "silly" questions to other people, especially the lecturers, because it might make them consider me to be "stupid" or not hard-working.	11	23.9%
Others	4	8.7%
I don't think the information is available anywhere else.	2	4.3%

Table 7.4 Reasons Given for Using the Web

Clearly convenience is the overwhelming reason mentioned by students with over 90% but also over two thirds consider it to be the best place to find information.

The reasons given in 'Others ' were (three responses written in):

- *I think it's better to try and solve the problem myself through checking the web rather than just be lazy and ask other people,*
- *I usually search the web first when I get stuck. It's generally faster than looking up in a (printed) textbook and saves me to ask stupid or obvious questions and I save time,*
- *The web seems to be the best place to ask as many questions as possible. Because Google doesn't give you the feeling that you're annoying them and that your questions are stupid. Whereas with a person you can't asked them questions 24hours a day. The internet is always there and is full of examples and information. Especially with languages like Java, because they are so widely used everyone has similar issues and you find so much help on the web.*

The questionnaire concluded with two open questions. The students were asked if they had any suggestions on how the module could be enhanced by adopting the Web. A large number, nineteen students, responded with suggestions; the write-in answers were:

- *I think the module should not adopt the Web too much because it is good for students to extend their programming knowledge by simply skimming through the textbook to find the desired resource; however, it could be good to have a lecture/some slides in each lecture where some links to good places to find feedback and help online (to mention one, java tutorials from Oracle or StackOverflow) so that students can have some sources to refer to knowing that they are actually useful and don't get you the wrong way.*
- *No, I think the module is good as it is.*
- *An anonymous forum would probably help others by assisting them with simple "embarrassing" programming questions.*
- *Kirk (1203) often leaves extra links to more detailed information at the end of the slides so if there is a particular topic covered in the lecture which you find interesting you can go onto the slides and find links to more information on that topic.*
- *Not really, no.*
- *Student wikis are pretty good idea, but to make them work better students would need to receive more incentive (maybe virtual reputation points)*
- *I believe this course works perfect so far. I think it would be efficient if students are asked to look up some of the topics covered during lectures and investigate what sort of problems are facing and how they are solving them. Thus, after few months when we face similar problem we would most probably have forgotten the exact solution but either the idea will pop up in our heads or we will remember that we can't find the solution to our problem in a particular place.*
- *Video lectures please :)*
- *It may be helpful to include more commented code snippets to demonstrate the correct syntax for the different techniques that we are taught in the lecture slides (or just in addition to them on the student wiki.*
- *Not really, I think people should just Google Java tutorials and read through them. Doesn't take too long and you can learn quite a bit?*

- *In the lecture slides/on the course page it would be good to have textbook references for each section of the course. It would also be good to have additional programming tasks that are optional for beginners, intermediates and experienced programmers.*
- *I think that giving students online resources and advice to look online for documentation, tutorials, blog posts and slides (slides from slideshare or something, not only from Uni) will be good.*
- *I don't think it's the role of the module to use the web, more the role of the students.*
- *The more online resources the better.*
- *Links in the slide shows so we can use your recommended links rather than Googled ones.*
- *Providing more online challenges so that student can practise their programming in an engaging way.*
- *More examples.*
- *I think that the web is already used to great effect in this module and a lot of web use for this module is self-determined in the first place.*
- *I would suggest not omitting code from lecture slides as they are a great source for ref.*

Finally the students were asked if they had any other comments. Their responses were:

- *Possibly make the programming labs more welcoming to students who have no experience in programming before.*
- *The programming labs should have been more open-ended, suggesting a way to do it, but allowing the student to go about it a different way if they want. (Encouraging creativity)*
- *I think it's a very well-done module considering the difficulty of teaching programming in a lecture. I don't think attempting to integrate with the web would go down too well apart from perhaps recommending resources, which unless I'm mistaken is probably already done.*
- *Otherwise a fantastic course. The fact that there are postgrad students available during programming labs is one of the best practices. They always explain things thoroughly...it is definitely one of the reasons why we program so good.*
- *Nope. they need the basic right first, but we are progressing very fast. Therefore quite a number of them have a hard time catching up.*
- *It would be really helpful if all of the machines on campus (ECS and iSolutions) used the same version number of the software we are learning with (primarily JDK 1.6 vs 1.7)*
- *Some parts of the lab specs were a bit confusing but usually asking demonstrators or students for clarification did help.*
- *Some lab write ups were noticeably unclear / incoherent, the same goes for the coursework.*
- *Great lecture content, always worth attending lectures as they are fun also!*
- *A big ask but to split the large lecture into those who have programmed and those who have not, then build from there.*

7.4 Data Analysis

The data from this survey provided some additional insights in answering the question as to the students' reliance on course supplied instructional materials versus student-selected use of supplemental resources such as online activities and materials. The results of the questionnaire provided valuable additional insights about how students modified their approach to using materials of their choice in addressing learning solutions. All students used self-selected supplemental resources at one time or another and to varying degrees. The question was not asked, but it would be interesting to examine if there were any differences between the responses given to the questions according to the amount of programming experience students had before starting the course. This could be a question to be added to future studies on this subject.

7.4.1 Time to Use the Web

This two way table shows the percentages using the Web at different times according to the level of prior programming experience.

When using Web	Level of prior programming experience			
	None	**Some**	**A lot**	**Overall**
Before lectures/labs	33.3%	64.0%	88.9%	60.9%
During lectures/labs	83.3%	64.0%	44.4%	65.2%
After lectures/labs	75.0%	76.0%	88.9%	78.3%

Table 7.5 When Web Was Used by Prior Programming Experience

It is clear that there are differences present:

1) Those students who have a lot of prior experience of programming make much greater use of the Web before lectures than those with no experience, nearly 90% compared to 33%.
2) During lectures those without background experience in programming use the Web a lot more than those who have experience.
3) Use of the Web after lectures does not differ so dramatically with the majority of all students (over three-quarters in each case) using the Web, with a greater proportion for those with prior experience.

7.4.2 Purposes for Using the Web

This two way table shows the percentages using the Web for different purposes according to the level of prior programming experience.

Purposes for using Web	Level of prior programming experience			
	None	Some	A lot	Overall
Learning a new theory involved in the lectures, e.g. inheritance, debug, array.	66.7%	64.0%	44.4%	60.9%
Learning extra theories, e.g. Enum.	33.3%	64.0%	77.8%	58.7%
Revision	50.0%	40.0%	22.2%	39.1%
Getting some help when struggling for formal provided labs/coursework	75.0%	76.0%	11.1%	63.0%
Others	25.0%	8.0%	44.4%	19.6%

Table 7.6 Uses of the Web by Prior Programming Experience

It is clear that there are many differences present:

1) For example, of those without background experience in programming, three quarters use the Web in getting help when struggling with their formal work (compared to only 11% for those with a lot of programming experience). However even those with some experience use the Web a lot for this purpose.
2) An even greater proportion of those students who have a lot of prior experience of programming use the Web to learn extra theories, nearly 80% compared to 33% of those with no prior experience of programming.
3) Again in using the Web for Revision, half of those with no prior experience use it for this reason while only a fifth of those with a lot of experience do so.

7.4.3 Reasons for choosing to use the Web

This two way table shows the percentages using the Web for different reasons according to the level of prior programming experience.

Reasons for choosing to use the Web	Level of prior programming experience			
	None	Some	A lot	Overall
I don't think the information is available anywhere else.	8.3%	4.0%	0.0%	4.3%
It's the best place to find information.	41.7%	80.0%	66.7%	67.4%
Convenient.	91.7%	92.0%	100.0%	93.5%
I don't have to rely on other resources, e.g. textbook.	33.3%	28.0%	33.3%	30.4%
Try to avoid asking "silly" questions to other people, especially the lecturers, because it might make them consider me to be "stupid" or not hard-working.	41.7%	16.0%	22.2%	23.9%
Others	16.7%	4.0%	11.1%	8.7%

Table 7.7 Reasons for Using the Web by Prior Programming Experience

It is clear that there are more similarities than differences present:

1) For example, the main reason for choosing to use the Web, convenience, was the same for all levels of experience, with each group showing over 90% choosing this reason.
2) The proportions in each group who gave as a reason not having to rely on books was very similar in each group, around one third.
3) There were considerable differences in response to the reason that the Web was the best place to find information with 40% of the non-experienced students selecting this reason compared to double the proportion in the group of students who had some experience of programming.

7.5 Summary

The outcome of this study could provide evidence for Research Question B & C. It is clear that the students learning to program make considerable use of the Web in assisting their learning. They use it before, during and after lectures on a variety of platforms, making many different uses of it and for numerous reasons. It appears to be a very useful part of the learning activity. Further analysis showed that there are differences in their use of the Web in learning to program, according to their prior background level of experience in learning to program. It appears that choice and convenience are the primary criteria for determining what sources are used and when they are accessed. Given the time constraints most students have, it is no surprise that they prefer the educational support medium that they can access when they need it.

8 NGT Study

Based on the general design as discussed in Section 3.4.1, the Nominal Group Technique (NGT) was the preferred tool to use in this data collection point, because it the most efficient way to get additional information the easiest and quickest following the end of term survey.

The end of term survey was useful for determining the extent to which students use the Web as an adjunct to other media for providing solutions to issues they encounter in the course of attending lectures and labs. Responses indicate that student of all competency levels use the Web extensively, with the exception of those who have limited programming experience. They use it mainly for its convenience (24/7) and the fact that it is unobtrusive to fellow students or lecturers and doesn't embarrass them before others. Written comments suggested that Web tutorials are good supplements to the course materials and are widely used as needed. With this information as background, the nominal group technique (NGT) is a useful follow up to drill down on some of the issues that have surfaced from the various surveys.

An NGT meeting was conducted with the objective to collect 1st year undergraduates' detailed perspective when they were introduced to programming in Java, with special attention given to learning approaches. On the basis of the preliminary findings from previous surveys, responses from this survey could provide detailed evidence for all of the three research questions (Section 3.1.5 & 3.4), which are:

B. Whilst learning to program in this connected world, to what extent do different students:
   a. Gain or lose confidence and motivation?
   b. Modify their approaches to learning as they progress?

C. How do students integrate their formal and informal learning activities? To what extent do students:
   a. Rely on formally provided materials and exercises?
   b. Develop individual approaches to learning which integrate real world and online activities and materials?

According to general design of this research (Section 3.7.2), any unclear points from this survey could be investigated in the future interviews.

8.1 Design

In order to answer the previous three research questions, the following questions were designed as follow up:

- **Part 0. Warm Up.**

In this part, the participants were asked to select from three personal questions for the purposes of testing selection choices and gathering some demographic data.

0.1 What is your gender?

Select from 1 to 3, while 1 means female, 2 means male and 3 means prefer not to say.

0.2 What is your origin?

Select from 1 to 3, while 1 means UK, 2 means EU and 3 means International.

0.3 What is your course?

Select from 1 to 3, while 1 means Computer Science, 2 means Software Engineering and 3 means Information Technology in Organisations (ITO).

- **Part 1. Does background matter?**

Participants were be asked to select from 1 to 5, and 1 to 5 where 1 means strongly agree, 2 means agree, 3 means disagree, 4 means strongly agree and 5 means not applicable, respectively. The reason why to offer an option to select a 5 (not applicable) is that, every participant will press one button while making a choice. Without that option there would likely be missing data and make it more difficult for researchers to analyse some participants' study background since they did not select for some questions. The following questions are designed to understand the relationship between learning background and learning outcomes.

1.1 For all, your backgrounds affect your learning very much.

Select from 1 to 4, since this question is for all the participants.

1.2 For those who have had some prior experience of programming before being introduced to program at university, that experience has been useful for your learning.

Select from 1 to 5. Novices are supposed to select 5.

1.3 For those who have no experience of programming before being introduced to program at university, your lack of prior experience put you at a disadvantage.

Select from 1 to 5. Pre-experienced participants are supposed to select 5.

1.4 For those who have a good math skill before being introduced to program in the university, this maths' skill has been useful for your learning.

Select from 1 to 5. With computer science and software engineering students in the University of Southampton, they can only take the degree if they have an A in A level maths. However, even with an A some may consider their maths ability/skill to be less than others. Also students in ITO do not have to have maths. It is worth mentioning that students at many other universities are not expected to have strong maths so it may be either a point for data collection, or it may be a point for discussion.

1.5 For the rest, your lack of maths' skill put you at a disadvantage.

Select from 1 to 5. The participants who have good maths' skill are supposed to select 5.

- **Part 2. What obstructed the learning progress?**

This part is built on the basis of the results from the weekly survey (Section 6.4.2). The following questions were designed to gather some qualitative data in terms of the reason why the participants got stuck when learning some topics.

2.1- 2.3 According to previous survey results, the top challenging labs are: Lab 5 (Arrays and API), Lab 6 (Testing and Debugging) and Lab 8 (Overloading). Could you specify the reasons for each why you consider they were challenging?

Participants received some pens and A5 size index cards to write down their thoughts. They were given approximately seven minutes to work individually. The investigator then spent about eight minutes collecting a summary of the items raised which were initially recorded on a white board by a reporter. Having collected and summarized the responses, the participants used the voting devices to select their priority.

- **Part 3. Any learning approaches?**

This part was designed to find out the detailed learning approach that participants were using whilst learning to program (related to the second research question).

3.1 What are your usual learning approaches?

3.2 What are your preferred learning approaches when you get stuck? E.g. you found an aspect of programming difficult as in the examples we have just discussed?

The response method and duration are the same as Part 2.

- **Part 4. Any learning activities?**

This part, combined with different question types, was designed to observe the participants' learning activities, intending to find out how they integrate their formal and informal learning activities (related to the third research question) and their actions when facing to some difficult situation. Since there were more than one type of question in this part, participants followed the researcher's instructions to select or write down their thinking.

4.1 You usually rely on formally provided materials and exercises.

Select from 1 to 4.

4.2 What are your physical world ("offline") learning activities?

Tick any that apply from the following:

Reading textbooks; printed notes; group study; library materials; getting help from your classmates or friends; ask demonstrators; ask lectures; others (please specify)

4.3 What are the online learning activities you use?

Tick any that apply from the following:

Discuss in social networking (please specify the platform); search from public searching engine (e.g. Google); watch online demon programming video (e.g. YouTube); learn and discuss on Forum/BBS; using school resources (e.g. ECS Intranet); online chatting (e.g. Skype); online course; browsing Wikipedia; online resource collection (OER); others (please specify).

For this question 4.2 and 4.3, participants will receive one-A4 page questionnaire which they are expected to answer. At the bottom of the questionnaire, the participants will be asked to write down what proportion of time they spend on each category of learning activities.

4.4 What is your individual approach to learning which integrate real world on online activities and materials?

This is a really difficult topic to pursue with this tool. However, it is included to help elicit a realistic framework of approaches to learning and learning activities which can then be later explored in greater details through semi structured interviews. Thus, the participants were only asked to write down their thoughts on the cards and not on a priority basis.

4.5 How do you respond when you receive diverse or maybe even contradictory points of view?

Since utilizing diverse and contradictory points of view is one important aspect in Integrative Learning, the participants were asked to offer their own approach to dealing with these situations. The response method and duration were the same as Part 2.

8.2 Preparation

The respondents were the first year undergraduate students who took COMP1004 (Programming Principles) in AY 2012-13 in the University of Southampton. The invitation of this event was created on Eventbrite and sent out on the 20th May, 2013, which was 10 days before the NGT meeting. In the invitation, the consent form and Participant Information Sheet (PIS) were attached, stating the policy for anonymity and the issue of withdrawal, etc. In this case, all the final participants took part at their own free will. Also, no incentives were made.

One whole set of Zappers containing 40 voting devices were on loan from the university library loan desk on the 29th May, 2013. Consent forms attached with PIS were printed in advance, packaged with answer sheets for Section 4, pens and cards labelled with different colours. Each package contained one piece of consent form (with PIS at the back of the paper), one Zapper voting device, one answer sheet for Section 4, one pen and eight cards (labelled with different colours for organizing purpose).

8.3 Event Flow

This NGT meeting was scheduled to get officially started at 13.00 on the 30th May, 2013. Since there were 16 responses for the invitation, a classic lecture room which is capable for up to 25 people was booked beforehand. Before the participants arrived at the meeting room, direction signs leading to the meeting room were posted. Meanwhile, a lunch with soft drinks was ready as promised.

The event finally started at 13.20 due to some late participants. There were 14 participants including one walk-in participant. Fortunately, the participants had different backgrounds and their programming abilities all varied, which offered the research more data and led to a much more substantial result. The meeting processes were strictly adhered to.

A. **Registration (13.00 - 13.20).** Each participant came to the researcher to register, receive a package, and a Zapper voting device. The participants then signed the consent forms (attached with PIS) and returned those forms to the researcher. They were free to refuse to sign and withdraw, if they were unhappy with the PIS and changed their minds.

B. **Briefing (13.20 – 13.30).** The researcher gave a welcome introduction to all the participants. There were three main points for this brief:
   a. State the purpose of this survey.
   b. Declare that this survey is 100% anonymous, so they can just respond with their actual opinions without any concerns.
   c. Emphasize their rights: they can withdraw at any time – or decline to answer any questions.
C. **Warm Up (13.30 - 13.35).** Explain how to use the Zapper voting devices and have them tested by asking the participants to answer the questions in Part 0. Replacement devices are available if necessary.
D. **Part 1 (13.35 – 13.45).** Answer the questions in Part 1: "does the background matter?"
E. **Part 2 (13.45 – 14.20).** Answer the questions in Part 2: "What obstructed the learning progress?"
F. **Part 3 (13.20 – 14.40).** Answer the questions in Part 3: "Any learning approaches?"
G. **Part 4 (13.40 – 14.55).** Answer the questions in Part 4: "Any learning activities?"
H. **Ending (14.55 – 15.00).** Leave any comments about their programming learning experiences and this event on the last non-labelled card in the package. Give out a 4GB USB memory disk to each participant as a thank you. Ask them to write down their name and email address on a card if they are happy to attend future individual interviews.
I. **Clean-up (15.00 onwards).** The researcher must have the meeting room cleaned before leaving.

To summarize, this event lasted less than two hours. During this period, the participants were free to have their lunch and ask any questions about the survey. Reminding them with their right to withdrawal and keeping smiling are also important, because all these behaviours will help the participants to relax and offer more data. It is worth mentioning that there was one late participant who came when the meeting was at Stage E when the other participants were writing down their thoughts for question 2.1. The late participant was asked and agreed to select the missing questions after the meeting finished.

8.4 Responses

Having collected and carried out some statistics analysis, the responses from this event could be illustrated as follow:

8.4.1 Part 0. Warm up.

Fortunately, the Zapper voting devices were all working well during the meeting. The following table and diagram shows the responses for the three warm-up questions in Part 0.

Questions and Answer		#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13	#14
0.1 GENDER	1 Female	√							√						
	2 Male		√	√	√	√	√	√		√	√	√	√	√	√
	3 Prefer Not To Say														
0.2 ORIGIN	1 UK			√	√	√	√	√	√	√	√	√	√	√	√
	2 EU	√													
	3 International		√												
0.3 COURSE	1 Computer Science		√		√	√	√	√			√	√	√	√	√
	2 Software Engineering	√								√					
	3 ITO			√					√						
	4 Not Listed														

Table 8.1 Responses for Part 0 in NGT Meeting

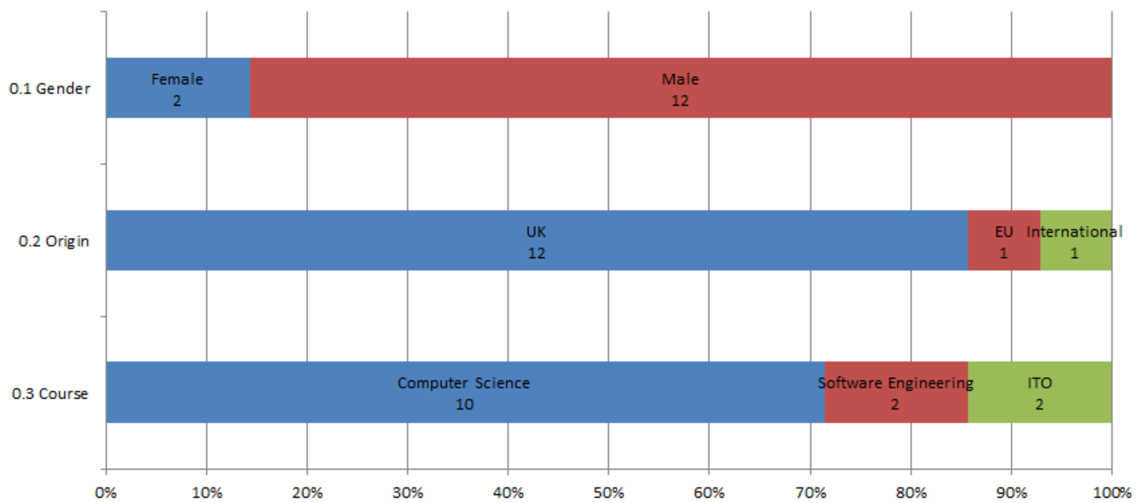

Figure 8.1 Responses for Part 0 in NGT Meeting

8.4.2 Part 1. Does background matter?

Responses from the participants are summarized in the following table, in which 1 means Strongly Agree, 2 means Agree, 3 means Disagree, 4 means Strongly Disagree and 5 means Not Applicable:

#	Q1.1 Backgrounds Affect Learning	Q1.2 Prior Experience Helps	Q1.3 No Experience Obstructs	Q1.4 Good Maths Helps	Q1.5 No Maths Obstructs
#1	2	1	5	3	5
#2	2	1	5	3	5
#3	2	2	5	3	5
#4	1	2	5	3	5
#5	2	5	3	4	5
#6	1	5	2	2	5
#7	2	1	5	3	5
#8	3	2	5	5	3
#9	2	2	5	3	5
#10	2	1	5	3	5
#11	2	5	3	1	5
#12	1	1	5	5	2
#13	2	1	5	2	5
#14	2	1	5	3	5

Table 8.2 Responses for Part 1 in NGT Meeting

Alternatively, the results are illustrated in much more intuitional diagrams:

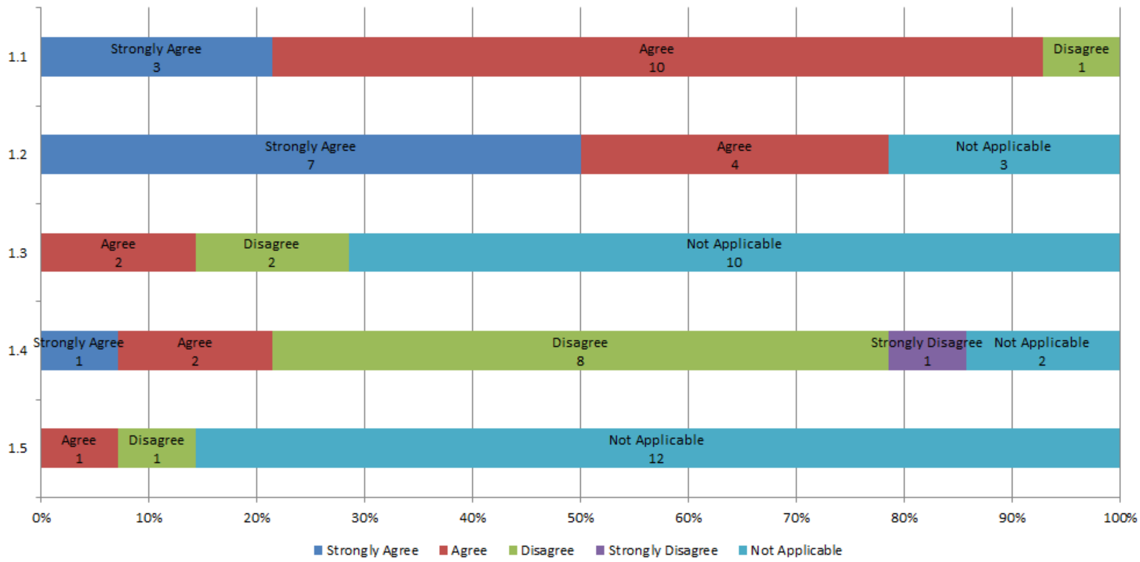

Figure 8.2 Responses for Part 1 in NGT Meeting

8.4.3 Part 2. What obstructed the learning progress?

Having collected and quickly read through the cards by the researcher, the following eight items were generated according to the participants' opinions about why they consider Lab 5 (Array and API) to be challenging:

1. These are brand-new concepts which caused lots of confusion.
2. Bad timing: the theories were explained in the lectures after the lab.
3. Lack of explanation/experience.
4. High-perceived workload VS limited effort.
5. Not happy with the programming environment (Eclipse).
6. Delayed from last lab.
7. Unclear/Confused specification.
8. Expectations

Before the participants started to rank, they were asked to confirm that the researcher has recorded all the items they just mentioned. If negative, the participants would receive another card to write down the missing points. For this question, the researcher recorded all the items and the participants were then asked to rate these responses which were the Top 3 reasons for them. In about 15 seconds, the participants finished their ranking. By assigning their Top 1 reason 3 marks, the second one 2 marks and the third with 1 mark, the priority results are shown in the following table, with the Top 3 reasons highlighted:

#	ITEM 1	ITEM 2	ITEM 3	ITEM 4	ITEM 5	ITEM 6	ITEM 7	ITEM 8
#1	1		2				3	
#2	2		3	1				
#3	3			1		2		
#4		2		1		3		
#5	2			3				
#6		3	2					1
#7		2	3				1	
#8	3		1				2	
#9			3	2			1	
#10			2		3			
#11	1		3			2		
#12	3			1			2	
#13		3	2				1	
#14	3			2		1		
SUM	**18**	**10**	**21**	**11**	**3**	**8**	**10**	**1**

Table 8.3 Priority Results for NGT Question 2.1

Having considered participants' detailed responses on collected index cards and the voting results, the top three reasons for Lab 5 (Array and API) are:

1. The use of some new concepts was not clearly explained.
2. It is difficult to understand some new concepts and API, which caused lots of confusion.
3. Failing to work hard enough with the increased workload.

Continuing to the rest of the questions in Part 2, the responses from the cards were combined with the previous eight items and the participants agreed that those items covered their thoughts. Using the same voting and statistics methods, the following two tables represent the priority results for Question 2.2 (Lab 6, Testing and Debugging) and Question 2.3 (Lab 8, Overloading), with the Top 3 reasons highlighted) as well:

#	ITEM 1	ITEM 2	ITEM 3	ITEM 4	ITEM 5	ITEM 6	ITEM 7	ITEM 8
#1			3			1	2	
#2	3		2					1
#3	2			1		3		
#4	3		2				1	
#5	2		1	3				
#6	3			1		2		
#7	1				2			3
#8	2		3		1			
#9	3			1		2		
#10			2		3			
#11					3		2	
#12	3		1	2				
#13			3			2		1
#14				3			2	1
SUM	**22**	**0**	**17**	**11**	**9**	**10**	**7**	**6**

Table 8.4 Priority Results for NGT Question 2.2

#	ITEM 1	ITEM 2	ITEM 3	ITEM 4	ITEM 5	ITEM 6	ITEM 7	ITEM 8
#1	1		2				3	
#2	3					2		1
#3	2		3			1		
#4	3			1			2	
#5	3	1				2		
#6				3		2		
#7	3				2		1	
#8	3				1		2	
#9	3					2	1	
#10			3		1		2	
#11	3		2			1		
#12		3		2				1
#13	3			1		2		
#14				1			2	3
SUM	**27**	**4**	**10**	**8**	**4**	**12**	**13**	**5**

Table 8.5 Priority Results for NGT Question 2.3

Same as the means on how the reasons are generated for Lab 5, the top three reasons for Lab 6 (Testing and Debugging) are:

1. Struggle with the new concepts which brought confusion. And, lack of help made this situation even worse.
2. Due to lack of prior explanation/experience, this lab was hard to begin.
3. It is a large learning course since it was the first time to use debugger.

For Lab 8 (Overloading):

1. The nature of the new concepts was hard to learn, and did not understand how those theories worked during the lab.
2. The specification is misleading.

3. Delayed from last lab.

8.4.4 Part 3. Any learning approaches?

Similar to the processes in Part 2, the participants agreed that the following nine items covered all their usual learning approaches when introduced to program:

1. Read the specification
2. Practice by writing code
3. Apply learning so far
4. Read lecture slides/textbooks
5. Search online, e.g. Google
6. Ask friends/Have a discussion
7. Ask experts
8. Reflection
9. Learn in sections

The ranking results are shown in the following table:

#	ITEM 1	ITEM 2	ITEM 3	ITEM 4	ITEM 5	ITEM 6	ITEM 7	ITEM 8	ITEM 9
#1					2	3	1		
#2	3			2	1				
#3		3		2				1	
#4			1		2	3			
#5	3			1		2			
#6					1	3	2		
#7	3	1			2				
#8		3			1			2	
#9		2			3				
#10	1	3	2						
#11		1	2		3				
#12	3	2			1				
#13		3			2		1		
#14				3	1	2			
SUM	**13**	**18**	**5**	**8**	**19**	**13**	**4**	**3**	**0**

Table 8.6 Priority Results for NGT Question 3.1

In terms of the responses for the learning approaches the participants used when they were stuck while learning to program, the previous nine items describe their normal learning approaches that they used. Apart from that, participants tended to feel frustration and lose motivation, if they still failed to understand some theories after several attempts. It is not a surprising phenomenon, because the feeling of learned helplessness is common for students. Thus, “lose motivation/feel frustration” could be added as the 10th item for the participants to rank, although it is a kind of “behaviour” rather than a real “learning approach”.

The list of the learning approaches used when stuck while learning to program is as follows:

1. Read the specification
2. Practice by writing code
3. Apply learning so far

4. Read lecture slides/textbooks
5. Search online, e.g. Google
6. Ask friends/Have a discussion
7. Ask experts
8. Reflection
9. Learn in sections
10. lose motivation/feel frustration

The priority results are shown in the following table:

#	ITEM 1	ITEM 2	ITEM 3	ITEM 4	ITEM 5	ITEM 6	ITEM 7	ITEM 8	ITEM 9	ITEM 10
#1	3				1		2			
#2	3			1	2					
#3	1				3			2		
#4					2	3		1		
#5					2	3		1		
#6		1			2	3				
#7					2	3				1
#8					3	1		2		
#9					2	1				3
#10					3	2				1
#11					3		2			1
#12					3	2	1			
#13					3	1	2			
#14				1	3	2				
SUM	**7**	**1**	**0**	**2**	**34**	**21**	**7**	**6**		**6**

Table 8.7 Priority Results for NGT Question 3.2

8.4.5 Part 4. Any learning activities?

The first question uses a four-point Likert scale and the responses are illustrated as follows:

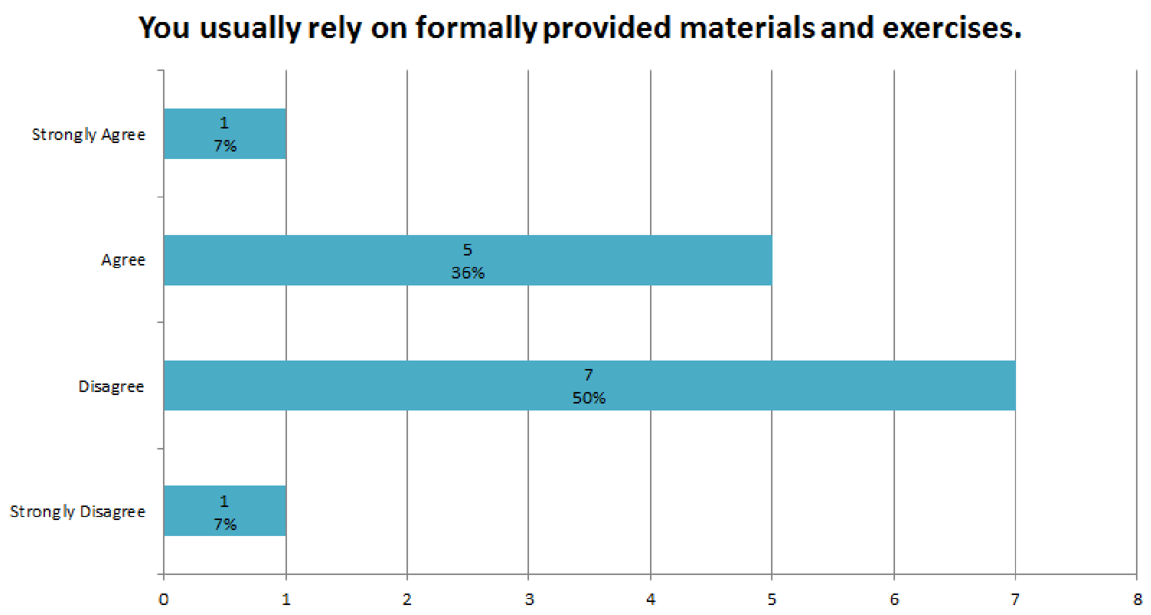

Figure 8.3 Responses for NGT Question 4.1

Surprisingly, contrary to the weekly observation during the practical labs, participants expressed that they were not very reliant on formally provided materials and exercises (e.g. weekly practical labs and coursework). Due to this unexpected discovery and with the objective of figuring out whether those exercises obstructed participants' learning, the researcher temporarily added another type of question to this one and asked the participants to select from 1 to 4:

4.1 Formally provided materials and exercises affect your learning in a positive way.*

The results are illustrated as follows:

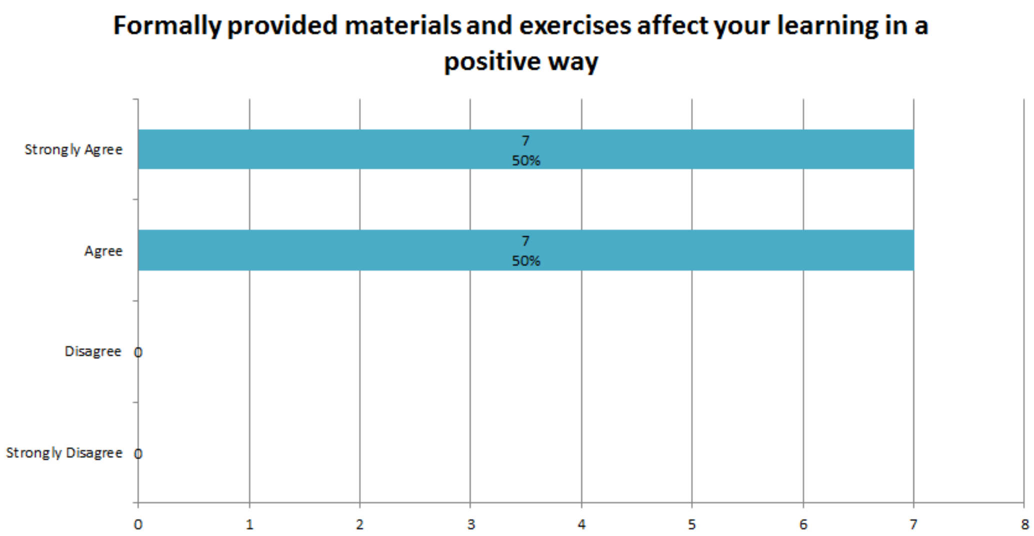

Figure 8.4 Responses for NGT Question 4.1*

The following two questions were answered on one A4-size questionnaire and the responses are reported in the following table:

Physical World ("Offline") Learning Activities		Online Learning Activities	
Selection	**Activities**	**Selection**	**Activities**
5	Read textbooks	10	Discuss in social networking
0	Read printed notes	14	Search from public searching engine, e.g. Google, Bing
8	Group study	7	Watch online demon programming video, e.g. YouTube.
0	Read library materials	5	Learn and discuss on Forum/BBS
10	Get help from classmates or friends	10	Use school resources, e.g. ECS Intranet
8	Ask demonstrators	6	Online chatting, e.g. Skype
5	Ask lecturers	1	Online course
		9	Browse Wikipedia
		1	Online resource collection (OER)
Others:			
1	Keep practice	2	Check Java docs/API
1	Apply learning so far	3	Check Stack Overflow

Table 8.8 Responses for NGT Question 4.2 and 4.3

In terms of the proportion of time participants spent on these activities, the numbers are reported as follows:

Physical World ("Offline")	5%	30%	10%	5-10%	50%	25%	20%
Online	95%	70%	10%	10%	50%	75%	70%
Ratio	1:19	1:2.33	1:1	1:1-1:2	1:1	1:3	1:3.5
Physical World ("Offline")	20%	80%	15%	20%	2.3%	1%	7%
Online	80%	20%	85%	80%	24.9%	99%	93%
Ratio	1:4	1:0.25	1:5.67	1:4	1:6.48	1:99	1:13.29

Table 8.9 Proportion of Time Spent on Learning Activities

According to the weekly observation, respondents often come up with various responses for Question 4.4 (What is your individual approach to learning which integrates real world on online activities and materials). However, the reason why the research still asked this question of the participants is that, the responses may offer ideas about how to refine this "big" question to "smaller" ones, so that the research could design more easy-to-answer questions in this area in the future. As a result, the participants only needed to provide their insights and not select on a priority basis.

As part of integrative learning, it is important to understand how our students respond, when they receive diverse or maybe even contradictory points of view, while learning to program. The last question is designed to gather such insights from the participants. After summarizing the cards, the responses are shown as follows:

1. Argue/Discuss with friends.
2. Do some in-depth research.
3. Evaluate each point of view critically.
4. Ask for another opinion.
5. Go with intuition.

Afterwards, the voting results are shown in the following table:

#	ITEM 1	ITEM 2	ITEM 3	ITEM 4	ITEM 5
#1		1	2	3	
#2	2	1	3		
#3	1	3	2		
#4			2		3
#5		1		3	2
#6	3	2	1		
#7	1	2	3		
#8		3			
#9	1	3	2		
#10		2	3	1	
#11					3
#12		2	3	1	
#13	3			2	
#14	3	2	1		
SUM	**14**	**22**	**22**	**10**	**8**

Table 8.10 Responses for NGT Question 4.5

8.5 Data Analysis

8.5.1 Part 0. Warm up.

In general, the respondents have different backgrounds and achievements. Comparing the results from the initial survey group and the NGT meeting, they are shown as follows:

	GENDER		ORIGIN			COURSE			Pre-experienced?	
	Male	Female	UK	EU	Int'l	Computer Science	Software Engineering	ITO	Yes	No
Initial Survey	142	27	94	23	52	53	7	14	129	40
NGT Meeting	12	2	12	1	1	10	2	2	11	3

Table 8.11 Comparison in Demographic Aspect

The number of students included in the two data sources is 183 which is a respectable number for a study of this type.

8.5.2 Part 1. Does background matter?

Generally, there is no evidence showing that a particular background will lead to a certain learning approach and perspective. Participants sharing a similar background may have different preferences, whilst those with different prior experience may prefer a similar learning approach. For example:

1. Almost all the participants (13 out of 14) agree that their backgrounds affect their learning style very much, although the only one (#8) who disagrees with that still believes that her prior experience of programming, before being introduced to programming at university, has been useful for her study.

2. Gender is not a factor that appears to affect learning preferences.

3. Previous experience does help. 100% of the pre-experienced participants believe that this learning experience has been useful for their learning.

4. The proportion of novice participants in programming is similar to that between the initial survey (23.66%) and the NGT meeting (21.43%). Normally, it is suspected that there would be a quarter of the novice students every academic year attending UK universities. However, divergent views exist in terms of whether the lack of previous experience puts these participants at a disadvantage. Two novices believed that this lack did not obstruct their learning, whilst the third one holds the opposite opinion. Detailed insights could be investigated in the future interviews.

5. Some academics worry that the classroom language might be a problem for our international students. Nonetheless, according to participants' responses, there is no relationship between their first languages and their learning approach/preference. Still, the research will ask some questions related to this problem in the future interviews.

6. Although Maths is a compulsory skill for computer science students and they must have an A for A-level maths to take the degree, 9 out of 12 participants who had received a good Maths' skill before taking the degree believe that this skill is not necessarily helpful for their learning. For the other two participants who did not have good Maths' skills, only one of them states that this lack

put him at a disadvantage. Thus, it is not clear that strong Maths' skill is definitely a necessary for learning to program from the students' view.

8.5.3 Part 2. What obstructed the learning progress?

The top reasons why students consider Lab 5, 6 and 8 the most challenging have been listed in Section 4.4.4. Subsequently, more insights were obtained from participants' responses:

1. The biggest obstacle for the students was how to understand new concepts in a short time, and it occurs often. Participants consider it as one of the top three most challenging labs. The total priority value increases from 18 for Lab 5 to 27 for Lab 8. Some participants also identified several new concepts that were introduced in the lecture after they had finished the lab.

   Taking some responses from the participants as an example:
   - *"The fact that, HashSet/Maps hadn't been explained at that point greatly added to troubles."* --- Lab 5
   - *"Quiz was difficult, first time we were made to think about scope."* --- Lab 6
   - *"Didn't understand how an interface worked when going into …"* --- Lab 8

2. There is an increasing negative effect caused by delaying some learning progress. Increasing numbers of students stated that, if they failed to understand the concepts from previous labs, that it would certainly delay their learning progress. For example:
   - *"Falling behind or understanding of previous labs had a detrimental impact on subsequent learning."* --- Lab 6
   - *"Based on lab 5, if you struggled with lab 5 which is a basic for this lab then it made it hard to comprehend what to code."* --- Lab 6
   - *"As it relied on 'last' week's work as I was behind this further, put me behind as I was stuck on Lab 7."* --- Lab 8

3. Lack of prior experience/explanation has greatly influenced students' learning in the beginning. However, this impact was reduced over time. Further questions were asked in the individual interviews to investigate the potential reasons for that, which may include strengthened ability of understanding programming and whether the explanation was actually getting clearer.
   - *"Introduced new learning that I did not have previous experience."* --- Lab 5
   - *"Debugging requires deeper understanding experience, which was lacking."* --- Lab 6
   - *"Lack of experience -> I didn't know how to properly overload the class constructor."* --- Lab 8

4. The impact from workload and self-effort is stable. The total priority value is 11, 11 and 8 for the three labs, respectively. It is encouraging to see that some students demonstrated a response that showed an internal locus of control. They believe that gains should depend on individual performance, which is a good sign, because, in this case, students would be highly motivated and learn better (Section 2.4). For example:
   - *"Didn't work hard enough."* --- Lab 5
   - *"Didn't work as hard to learn things."* --- Lab 6

5. In terms of the programming environments (Eclipse), participants complained more about the testing and debugging environments than other aspects.
   - *“Eclipse has an awful debugger. I’m used to VStudio so found it hard to adapt.”* --- Lab 6
   - *“Difficult to understand how to use the debugging tools – still quite unsure!”* --- Lab 6

6. Another argument was focusing on the specification. Although the total priority values fluctuate, they always lie in a high level (10, 7 and 13). Since many participants usually learn by replying on formally provided exercises, a comprehensive and easy-to-read specification is essential for these students. Otherwise, students may get de-motivated.
   - *“Specification of task not very clear – difficult to understand.”* --- Lab 5
   - *“The program was easy but the specification made it difficult.”* --- Lab 6
   - *“Misleading specification.”* --- Lab 8

8.5.4 Part 3. Any learning approaches?

This section summarizes the participants’ usual learning approaches and preferred ones when getting stuck. Then, the potential relationship between these two kinds of learning approaches is analysed.

- **Usual Approaches**

1. Although searching Google receives the highest priority, only two participants selected it as their first-choice in a learning approach. Strategically, it is considered to be auxiliary and secondary. Meanwhile, both of the participants who considered Google as their top learning approach, selected “practice by writing code” as their second choice.
   - *“If have a problem go to API; if still have a problem, Google.”*
   - *“Going to lectures. Searching online.”*

2. The second highest priority was in “practice by writing code”, whilst four participants considered it to be the top learning approach.
   - *“Learn by doing. I will try and do it.”*
   - *“Practicing almost like how one would practice for a sport.”*

3. “Ask friends and have a discussion” and “re-read the specification” tied for the third place. However, more participants (three and four, respectively) chose these two as their top learning approach rather than Google.
   - *“Understanding the concept by talking to my course mates.”*
   - *“Find the given resources and read through them.”*

4. In general, usual learning approaches for participants are more likely to be simple and individual learning methods instead of complex ones.
   - *“Practice. Ask a friend. Ask another friend. Ask someone better than me. Search online. Read lecture notes again.”*
   - *“Initial learning usually done from textbook/reference text, then actually applying the gained knowledge in a scenario where it would be useful.”*

5. In terms of the formally provided lecture slides/notes and textbook, half of the participants mentioned reading and reviewing those on their cards. However, only four participants selected

this approach as one of their top three choices and the final priority for it is pretty low. Detailed reasons were gathered from the individual interviews.

- **Approaches Used When Getting Stuck**

According to Table 8.8, the most popular learning approaches when participants got stuck are "searching on Google" and "ask friends and have a discussion". In details:

1. 100% participants selected Google as their top 3 learning approach when they got stuck while learning to program, in which 50% respondents believed that they would Google once they need some help.
    - *"Google. YouTube tutorials. Help from some with experience."*
    - *"Searching online for the solution to that problem."*

2. Those who selected Google as their first choice mainly argued that "ask friends and have a discussion" and "reflection" would be their second and third choice.
    - *"If at first I don't succeed, I ask Google. Failing that, I ask somebody with superior ability. Failing that, I ask StackOverflow (unless my Google search brought me their first, of course)."*
    - *"Use Google to see if other people have had same issue. Come back to it later with fleshed mind. … Ask course mates to see if they can suggest a way through problem."*

3. 10 out of 14 participants prefer to ask their friends and have a discussion about the problem, where four of them believe this to be their first choice. Interestingly, all of these four participants selected Google as their second choice.
    - *"Ask someone. Search."*
    - *"Ask colleague how they worked it out, for example, if a certain lecture website helped them. Search online for help."*

4. Apart from "searching Google" and "asking friends and have a discussion", priorities of other items are all at a much lower level.

5. Lecture slides and textbooks are still not popular. Although four participants stated this approach on their cards, the final rating is still low and only two participants adopted that as their top three learning approaches and both of them selected it as the third choice. One participant described reading the textbook was a "very rarely" activity. Detailed reasons are expected to be gathered from individual interviews.

6. Participants may feel frustration and lose motivation if they failed to solve the problems they face while learning to program. Due to the learned helplessness concept, this is not an unexpected phenomenon. Academics need to pay more attention to such issues, because once they have successfully dealt with a challenge, students will be highly motivated and their learning will be promoted as observed in the weekly practical labs.
    - *"Problems arise if initial understanding is not sufficient; with very advanced concepts where background knowledge is required but not specified this inhibits movement to the implementation phase and causes frustration."*

- **Relationship Between Two Kinds of Learning Approaches**

1. 100% of the participants selected Google in their top 3 learning approaches when they got stuck while learning to program, in which 50% of the respondents also believed that they would Google once they need some help.

2. Those who selected Google as their first choice indicated that "ask friends and have a discussion" and "reflection" would be their second and third choices.

3. 10 out of 14 participants prefer to ask their friends and have a discussion about the problem, where four of them believe this to be their first choice. Interestingly, all these four participants selected Google as their second choice.

8.5.5 Part 4. Any learning activities?

1. 8 out of 14 participants state that they do NOT usually rely on formally provided learning materials and exercises. However, all the respondents agree that those resources promote their learning. Since this result is different from the weekly observations, detailed insights should be investigated in the future interviews and observations.

2. There are seven kinds of "offline" learning activities declared which received 38 selections in total. 10 out of the 14 participants managed to get help from their classmates and friends whilst learning to program. The second favourite activities are "group study" and "asking demonstrators". It is unexpected that "reading textbooks" and "asking lecturers" have received the fewest selections. Reasons could be investigated in future interviews.

3. Online learning activities have received almost double selections (68) as "offline" ones. "Searching from public searching engine" is the most widely adopted activities, since all of the participants have selected this item. Apart from that, "discussing in social networking" and "using school resources" were considered to be the second most popular ones. Other online resources such as Wikipedia, demonstration programming videos and relevant forum/BBS were selected as the third most preferred online learning activities.

4. In terms of how to integrate physical world activities with online activities and materials, the participants offered a variety of ways:

   a. Four participants prefer adopting online activities (such as Google for solutions) when struggling with "offline" resources.
      - *"Resort to online materials for help when I am struggling with offline materials."*
      - *"I use online world learning to fix issues in physical world learning."*

   b. Four participants meet their friends and have a discussion about what they have read from books and online materials. Then they upload their findings to the Web.
      - *"We met up in small groups to discuss and to through code we found online."*
      - *"Put group study stuff online."*

   c. Some students like to learn online first, then apply them into practice. Apart from listening to the lectures, physical world activities are more likely to be treated as a "back-up" strategy.

- *"I read just about all resources I read online then step away and think through what I have read away from the source to work out the how and why of what I have learnt."*
- *"Physical world activities generally serve as a backup or secondary fall back to the wider variety (in micro and macro scale) of information available online".*

d. Similar to the findings from the End Session Survey, one respondent states that he/she did not pay much attention to the offline because online materials are much easier and quicker to find.

e. Real world learning activities are not totally abandoned. One participant argues that the online resource cannot offer him/her the expected answer, so he/she prefers to study in the "offline universe". Potential reasons could be investigated in future interviews.
   - *"The online world is generally pretty bad at answering the question you asked. Usually, if I want a specific answer to a specific problem, I need to look in the offline universe."*

5. It is obvious that participants all have their own preferred ways to address diverse or maybe even contradictory points of view while learning to program.

   a. According to the selections, participants prefer to "do some in-depth research" and "evaluate each point of view critically" most often.
      - *"Try to understand reasoning behind each viewpoint, then make an informed decision."*
      - *"Differentiating between two sources is usually done by experimentation. I first try the simplest answer, then the more complex answer whichever works best, sticks."*

   b. Although "ask/discuss with friends" was selected in the third place, there are three participants voting this item as their first choice.
      - *"Check with other people to see what they think is correct."*
      - *"Ask my friends (either on Facebook group or offline)."*

8.6 Summary

A close look into the item ratings and written comments indicate that learning occurs in a variety of ways depending on the circumstances and the individual. Preferences vary as to how individuals deal with the need to solve a sticky problem. At some point it becomes evident that additional resources may be needed to solve the problem. At this point, individual differences come into play. Whether to use a Web source, consult with another student or rely on course materials is a matter of personal preference and the data about which yields better results is scant. In this study, the variance between the end of term survey, internet survey and NGT group sometimes yielded contradictory results. That can be attributable to differences in the way the information was obtained and in some instances small sample sizes. In any event, it is important to note that the course provided some guidance as to what to do when pressed for a solution. Course materials were helpful to some. Others preferred to use different sources. It seems obvious that this could become the object of another research study, either as an extension or a repeat of this study at some point or it could incorporated into a new study.

The data collected from this study could be evidence for Research Question B & C. In the next phase of this research, individual interviews will be conducted. According to the data analysis so far, especially the unclear points from this survey, the following questions should at least be included:

1. Is it true that no prior programming experience will affect learning very much?
2. Is it a problem that learning to program in a foreign language for those students whose first language is not English?
3. Does maths' skill really matter?
4. Why textbook is not popular?
5. To what extent do students rely on formally provided exercise?
6. Why some students prefer NOT to ask lecturers when help needed?
7. Is it common that the Web can't offer students what the "answers" they required?

9 Individual Interviews

The original intent of this study was to include both qualitative and quantitative data in the search to answer the research questions. It was important, therefore, to conduct in-depth interviews as a valuable contribution to the knowledge gained in this study. The process and results of the individual interview are presented in this chapter.

The respondents were the first year undergraduate students who took COMP1202 (Programming I) in AY 2012-13 in the University of Southampton. Invitations, attached with the consent information, PIS and consent forms, were sent out via email to the volunteers recruited from previous NGT meeting first from 23th September, 2013. After about two weeks, if only a few students agreed to attend, invitations were to be sent to all the other undergraduates who have finished the introductory programming module in the school of ECS in the University of Southampton from 7th October, 2013. In this case, all the final participants took part at their own free will. No incentives were made.

This interview was designed to expand the research outcome from the previous NGT meeting, which could provide detailed evidence for Research Question B & C (Section 3.1.6 & 3.4):

B. Whilst learning to program in this connected world, to what extent do different students:
    a. Gain or lose confidence and motivation?
    b. Modify their approaches to learning as they progress?

C. How do students integrate their formal and informal learning activities? To what extent do students:
    a. Rely on formally provided materials and exercises?
    b. Develop individual approaches to learning which integrate real world and online activities and materials?

It focused on understanding the in-depth perspectives from the students who had just finished the introductory programming module in the University of Southampton, including the relationship between prior skill background (e.g. Maths) and the final learning outcome, the learning approaches used (textbook VS Web), etc. The participants were undergraduates in the school of ECS from the University of Southampton who provided their contact detail during a previous NGT group discussion.

9.1 Post-Programme Interview Protocols

There were eight of the 20 students contacted who agreed to participate in the interviews – 3 females and 5 males. They were each briefed as follows:

Interviewer: Before we start I need to ensure that you are over 18 and fully understand this interview is only for academic research, and the personal data will not be used for commercial use.

Respondent: Yes okay.

The generic questions consisted of the following lead questions with appropriate follow up questions. These are logical questions to ask in support of the primary research questions, which were essentially designed to determine ***whether personal background factors affected approaches to learning and the use of online and offline resources***.

The interview questions are intended to further drill down on the original research questions as well as follow up on the issues that were raised in the NGT exercise which suggested that students' prior educational experience, lecture/lab experiences and other factors affect their self-styled learning choices. These are the follow up questions that were derived from those prior results (Section 8.6).

1. **Motivation:** What is your motivation for learning to program? Did your motivation remain the same during the whole semester?
2. **Language:** (For non-UK origin only) How does the foreign language (English) affect your learning?
3. **Maths Skill:** Can you describe your maths' skill before the module? How much do you think it influenced your learning of programming?
4. **Gender:** Do you think gender is a factor that affects learning to program?
5. **Prior Experience:** How does your prior programming skill (none or lots) affect your learning?
6. **Formal Learning Activities:** How much do you rely on the formally provided learning materials and exercise?
7. **Textbook:** What do you think about the set textbooks?
8. **Help:** Who do you rather ask/turn to when you need some help? (Why not lecturers?)
9. **Giving Up:** Have you given up learning some topics? What caused that situation and why you decided so?
10. **Online Learning:** If you are using online resources, what do you do? How do you integrate your physical world and online learning activities? (time spent, content, how to balance these two.) Does feedback matter?
11. **Expectations:** Any expectations of the academics? (What support or resources do you expected for the academic staff?)

NB. The description and analysis of the equations in the rest of this chapter will use the question stubs as an index the questions.

Each of the interviews was recorded with their consent and verbatim transcripts were made for each interview which were approximately 20 minutes each. This chapter will provide insights into the questions that were asked of each interviewee and their contributions to the overall research agenda for this study.

9.2 Tag Cloud Overview

One of the ways to give an overview of results is through the use of word tag clouds expressed by their font size and colour relative to the other tags making them more easily recognized (Lohmann et al. 2009). Tag clouds are visual representations of social tags that are based on word associations. In this case the words are from the combined verbatim interviews which consisted of nearly 7,000 words in a single document. Tag clouds are a visual representation for text data whose importance is shown by font size or colour. It is a quick way to show the content of the 8 interviews and provide a basis for showing the relative importance of word and phrases alphabetically.

The tag cloud illustrated below was derived from the verbatim interviews for all 8 of the personal student interviews and is shown in Figure 9.1. The words that stand out are "learning" and "programming" which were used 128 and 94 times respectively. The entire interview was about the students' experiences in learning to program in java and their prior experience with basic. Respondents were asked a number of questions about what their academic experience with the lectures and labs and the learning exercises associated with them. Attention was also paid to texts and use of online support via the internet for additional resources.

One can also see that they discuss motivation, feedback and challenges they have experienced. This can be a particularly useful concept for getting a gestalt overview as to what one can expect to see in the individual interviews which provide the details and context for the comments that were made. Moreover, the individual respondent differences can be noted and provide a better insight into the learning experiences that occurred over the duration of the lectures, labs and assigned learning exercises.

Figure 9.1 Word Tag of Most Frequent Words Used in Personal Interviews

The comments made by the 8 interviewees embellish the concepts that are found in the tag cloud overview. These will be addressed on a case-by-case and question by question basis for each interviewee. A summary analysis and discussion is then follows.

The tag cloud shows that the respondents spoke extensively about a learning situation in which they were learning computer programming. There were lectures and learning modules that required problem-solving that often used online resources. There are academic challenges that affect their motivation levels and for which they need help.

The specific conversation upon which the tag cloud is based comes from the interviews that were conducted with 8 respondents who had taken the computer programming course. Their anonymised answers are shown on Section 9.3 below. Although the questions were usually asked verbatim as shown and in sequence, there were 2 or 3 interviews in which the line of questioning led to follow up questions that took the conversation in a different direction. In these instances, the 20 minute time constraint prevented all of the questions to be completed. The questions were not avoided, they were simply not specifically asked. This happened infrequently and is not considered to undermine the findings.

9.3 Responses

Each of the interviewees contributed some unique insights into their learning experience, as well as echoing the comments of the other students. In the accounts below each respondent is assigned an ID to maintain their anonymity. The abstracts summarise some of the perspectives of some interviewees and provide insight into their views on each of the interview questions that follow:

- **HS**

HS actually enjoys programming because he enjoys watching something that he made work. He enjoys working out what's wrong with it when it doesn't work. So it's the enjoyment of having something he created. In terms of the maths' skill (Section 8.6), he explained that:

> *"Sometimes it becomes challenging, but no I don't think I ever stopped enjoying programming. I don't feel that a good knowledge of maths is a requirement for programming. Possibly the logic and analysis skills are applicable, but the mathematics in itself I don't feel is of great use."*

He didn't watch any online video tutorials, it was purely textual. He used the exercises more as an inspiration than just something to be done. He considers that good programming is good programming. He thought that if his code style was standard, correct and made no errors then he would still receive high marks regardless of whether he followed a formally prescribed method or not. He thinks the Programming One module was very well planned, at the beginning he found that it was a little bit too easy. In that time he did his own learning. As the module's difficulty increased it scaled very well, so there were always extensions he could access, the course work had extra marks for adding extra features. He thinks he could still challenge himself with perhaps more advanced programming.

He did not read the text books. It wasn't because of difficulty or not challenging, it's because he has always had, in terms of a personal learning style, found it very difficult to learn programming out of a text book. It's just his style, for him programming is a very interactive experience. He writes some code, he runs the code, he watches the result, and he checks that the result is what he wants it to be. It's a sort of iterative approach to programming and that's the way that he learns, it's very difficult for him to learn just by reading out of a page of just static.

There were books, there was the textual tutorial and as I mentioned before, I subsequently found out it was very difficult. It was much easier to program by experience rather than reading, so I think that's why my earlier attempt to learn was not successful. He doesn't remember giving up on any topics now. There were some topics which he learned that he was less comfortable with than others, but he never gave up on any of them.

- **M**

He said that there were a few motivations at work in his case; programming is an enjoyable experience, it felt like a creative experience and, of course, for him there was an element of wanting to do well in your subject and get good grades. Another factor about learning to program was the prospect of transferring that learning to a successful job. He said, if anything, his motivation increased; the more he learned, the more he could see what was possible. Once he got to the stage where he felt fluent enough to work with ADIs and the use of third party libraries, he found that there was so much more he could do.

He believed that Maths is important, but maybe not when he first started learning. When he first started learning it seemed like it was a lot of learning key words, learning how they operate with each other, learning basic constructs. He thinks knowledge of maths comes in at a later stage, when trying to design efficient algorithms and that sort of thing, and when he's a bit more experienced in programming.

He thought the text book was really good. He could really follow it in the book. He thought practice was very important for programming, particularly in the early stages when he was still learning the pattern, the words and how it all fits together. So, he used the text to supplement the coursework and the tutorials, just to get more practice, not necessarily for any concepts, but to just to go over the concepts they had already learned in more detail. He thought it was really a good text and that it was really well written.

At the start of the year he learned mostly offline but, as the course became more and more difficult, he went increasingly more online. That is because he thought the text book did cover everything he needed to know, but as it became more advanced, there was more and more stuff was not within the realm of the text book. He thought the lecturers are quite busy, so that unless it was something he couldn't solve by any other manner, he would email the lecturer after that. But if it can be resolved by using a helper or a text book or something, he thinks it's almost a waste of a lecturer's time to get involved in something so trivial.

All the other six respondents (A, DK, J, K, Li & Lu: three male and three female) preferred not to have their perspectives published as above. However, they were still happy to offer their responses as evidence for answering the research questions (Section 9.4, Figure 9.1).

9.4 Data Analysis

The interviews added yet another dimension to the insights gleaned from the data analysis. The interview results were consistent with was found previously with the survey data and nominal group technique. The themes that emerged from the interviews are summarised below.

NB. In the highlighted extracts below like responses have been clustered and wording has been highlighted by the author to suggest emerging themes. Codes for those themes are show on the right hand side of each table.

1. **What is your motivation for learning to program? Did your motivation remain the same during the whole semester?**

   Five of eight enjoy programming. Two took the course to please family. One thought it was good grounding for a job.

SUBJECT	RESPONSE	CODE
A	**Like** it.	Pleasure
DK	I **like** programming	Pleasure
M	I **enjoy** it. Is a good **grounding**.	Pleasure, planning
HS	I **enjoy** programming, working out what's wrong, and creating something.	Pleasure
J	It is **interesting**	Interest
K	To **please** family	External pressure
Lu	**Jobs** at the beginning, then just want to **pass** and **please** family.	Planning, survival, external pressure
Li	At the **beginning, positive**, could do it well. Missed one lecture and felt being behind motivated to "**pass exam**."	Transition, survival

Table 9.1 Responses and Codes for Motivation (Question #1)

2. **(For non-UK origin only) How does the foreign language (English) affect your learning?**

   None felt language was a barrier.

SUBJECT	RESPONSE	CODE
A	N/A	N/A
HS	English is my first language.	N/A
J	N/A	N/A
Lu	N/A	N/A
M	N/A	N/A
K	It is **OK** with me.	Comfortable
Li	**No problem**.	Comfortable
DK	**A little**. I didn't ask English speaking experts for help, because I'm afraid that my English is **not good enough to describe** my problem and understand their explanation. The lectures and learning materials are **fine**.	Comfortable

Table 9.2 Responses and Codes for Language (Question #2)

3. **Can you describe your maths' skill before the module? How much do you think it influenced your learning of programming?**

   All felt they had good math skills but most of them (six out of eight) weren't sure if that helped or not.

SUBJECT	RESPONSE	CODE
A	**Good as required**. I believe it is **helpful for creating computing thinking**.	Helpful, belief
DK	I believe I have strong math skills. But I don't know whether it is helpful, at least I didn't realize it. Maybe something potential? **Not sure**.	Uncertain
M	**Good**. Got an A in A level maths. **Not sure** whether it was helpful.	Uncertain
HS	I took mathematics A level and I took Further mathematics. Have **no feeling** about whether it was helpful.	Uncertain
J	**Good** as required. **Didn't realize** this was helpful or not.	Uncertain
K	I'm pretty **good at** maths, but I **don't know** if that helped.	Uncertain
Lu	**Good**. **Don't know** if it was helpful.	Uncertain
Li	**Good** as required. **Don't believe that helped**.	Belief

Table 9.3 Responses and Codes for Maths Skill (Question #3)

4. **Do you think gender is a factor that affects learning to program?**

   None thought gender was an issue other than the belief that males enjoyed computer stuff more.

SUBJECT	RESPONSE	CODE
A	**Don't know**. It should be the same.	Uncertain
DK	**Don't have a feeling**.	Uncertain
M	**Don't know**.	Uncertain
HS	**Don't know**.	Uncertain
K	**I didn't see any difference**. I have lots of male classmates and we mainly did the same learning.	Negative
Li	**Maybe, but not sure**. Don't feel anything special.	Uncertain, positive
J	Maybe boys like to program better than girls, because **boys like computers**. **Not sure**.	Uncertain, positive
Lu	Maybe. Think **guys may learn better** they have more interests in this area.	Positive

Table 9.4 Responses and Codes for Gender (Question #4)

5. **How does your prior programming skill (none or lots) affect your learning?**

   Those with prior experience with Basic or Python though it helped. Those without prior experience thought it might have helped.

SUBJECT	RESPONSE	CODE
A	**Positive**. Helped me **a lot** to understand how java works and how to create code.	Positive
HS	I have some, and I **believe it helped**.	Positive
J	Some Visual Basic. Yes **it helps** to build some computing.	Positive
Li	**A little** Python. It **helped to understand** some basic concepts but **not for advanced topics**	Limited
M	**Yes in some ways**, but in other ways no. Java is object oriented, Python is sort of imperative	Limited
DK	I don't have any, but **should be helpful**.	Novice, positive
K	I don't have any but it **should be very helpful** if I do.	Novice positive
Lu	Don't have any, but **should be helpful** if I did.	Novice positive

Table 9.5 Responses and Codes for Prior Experience (Question #5)

6. How much do you rely on the formally provided learning materials and exercise?

Five of the group thought the program materials helped very much. Two of the others found some of the exercises helpful. One didn't use them at all because he had prior programming experience. Those who did not rely on those materials much are all well programming-experienced students, whilst those who highly replied are mainly novices and students who have a little programming experienced.

SUBJECT	RESPONSE	CODE
DK	**A lot**. I think these will help me **get a better mark** for final exam. But I do practice **online for extra** exercise.	High reliance, Conscientious dependence
K	**Very much**. It took me lots of time and I'm sure I can **get a good mark** if I finish these well.	High reliance, Conscientious dependence
Li	**Very much**. Believes they **help to pass** exam. Had **no time to find other** places to practice.	High reliance, Conscientious dependence
Lu	**Very much**. It takes lots of time already and **these should help**.	High reliance, Conscientious dependence
M	I **worked through all** the work that's given. I worked through the tutorials and the coursework and used the book a lot.	Conscientious dependence
HS	I would **rate it a 3 out of 5** because it was **useful at the start** and then I went off on my own doing my own research.	Some dependence
J	I **didn't think this much**. If you ask, **maybe not that relies on**. I just tried my best to finish them, and I can practice for fun as well.	Semi independent
A	**Not so much**. I got experience and I know how to practice apart from these.	Low reliance, independent

Table 9.6 Responses and Codes for Formal Learning Activities (Question #6)

7. What do you think about the set textbooks?

Four of them found them helpful in different levels. One used a textbook in his own language. One didn't read it at all and the two others scanned through it after lectures.

SUBJECT	RESPONSE	CODE
M	I thought they were **really good**. I could really follow it.	Positive
K	I think it is **really good**. I read it, **not a lot**, but **frequently enough**.	Positive
Li	Seemed to **help in the beginning** but didn't use it half way through.	Limited Positive, evolving behaviour
Lu	Used it to **learn after lectures**, but didn't read it frequently.	Mixed/unreliable?
J	I didn't read it much I'm sorry. **At the beginning I did** although.	Limited Positive, evolving behaviour
DK	I've got a good textbook but **written in my mother language**, which helps a lot. It has lots of examples which make programming easier to understand.	Alternative substitute, customised
HS	I **did not read** the textbooks.	Negative
A	**Barely** read it.	Negative

Table 9.7 Responses and Codes for Textbook (Question #7)

It appeared from the interviews that students customised or personalised their textbook use. However it is clear, as with the surveys, that for many the textbook was not seen as relevant.

8. Who do you rather ask/turn to when you need some help? (Why not lecturers?)

Half of the group asked classmates for help, plus demonstrators and lecturers occasionally. The rest used Google for help.

SUBJECT	RESPONSE	CODE
A	Mainly from **demonstrators** and **lecturers**.	Formal support
J	**Lecturers**, **demonstrators**, **friends**. All the people I can. And **Google** of course.	Formal, informal, peer, online, mixed, customised
K	**All the people** I could ask.	Personal customised
Lu	Asked **experienced classmates** a lot. A little bit **shy to ask demonstrators or lecturers**.	Trust peers,
Li	Normally **Google** for help. Rather **ask female friends**. More convenient. Lecturers are not first choice. **Feel stupid if asking a silly question**.	Trusted peers,
M	I usually **Google** things first. If **I misworded something, I might talk to a demonstrator**.	Authoritative trust
HS	When I have a problem to solve, I go looking for **tools** I need to solve it. Most of the time comes from just looking at **Java documents**.	Authoritative not personal
DK	I **didn't ask people** frequently. I didn't talk much.	Not personal

Table 9.8 Responses and Codes for Help (Question #8)

Some interviewees (Li & Lu) also reflected that, rather than discussing codes on Facebook group by typing all the words, they preferred to have face-to-face discussion. The reason is that they thought it was difficult for them to describe the problems they met. It was much faster for them to sort out their problems by orally expressing whilst showing their code to the people they asked. This phenomenon is similar to the observation in the practical labs (Section 6.3.1).

Again the students are using both the university infrastructure and their own online practices to customise their support. High value appears to be placed on peer learning and support, perhaps fear of 'exposure' of ignorance, undermines students'' unwillingness to approach academics for formal assistance. However, since there student peers are more readily available, it may be possible to see this practice as useful and realistic, and indeed it mirrors practice in the working environment to which professional programmers are likely to be exposed.

9. Have you given up learning some topics? What caused that situation and why you decided so?

Only one of the interviewees gave up one lab. All the others had ever given up on a topic, but found a solution.

SUBJECT	RESPONSE	CODE
Lu	Gave up last lab. I **didn't feel missing one lab would fail** the course.	Judged withdrawal
A	**Never**.	Persistent
DK	No, **never**.	Persistent
HS	**Never**.	Persistent
J	**Never**.	Persistent
K	**Never**.	Persistent
Li	**Haven't given up** but for advance OOP topics and labs 8 and 9 didn't have a deep understanding. Feels like if first 7 labs could be learned, would pass exam. I would spend more time on them.	Persistent
M	I **don't think** it is something I have ever done.	Persistent

Table 9.9 Responses and Codes for For Giving Up (Question #9)

10. If you are using online resources, what do you do? How do you integrate your physical world and online learning activities? (time spent, content, how to balance these two.) Does feedback matter?

Everyone used Google frequently but also got help from friends. Everyone valued feedback as a way to learn and avoid wasting their time if in error, and it was the most important factor for some interviewees that affect how they integrate physical world and online learning activities.

SUBJECT	RESPONSE	CODE
HS	Online tools, especially Java documents.	Independent learning, personal trust, information seeking strategies
J	When I need help and learn theories. I spend half online and half offline. I'd rather show programming to others online. I go to lectures, then practice offline and if I need help, I may ask offline or Google online.	Customisation, independent learning, personal trust, information seeking strategies
A	I use online resources a lot, for getting help and learning theories. Feedback is important. I may lose some content from lectures but the feedback could remind me what was missing and what I have done was right or not.	Independent learning, information seeking strategies
DK	I Google a lot for help. There are also lots of useful BBS where I can learn. Mainly online, Offline is mainly lectures and textbooks, practice. Getting help online, and sometimes learn online as well. Feedback is important to correct my learning strategies.	Customisation, independent learning, information seeking strategies
K	Mainly getting help online, and online live demonstrating videos help. I don't know how I integrate, maybe from feedback. I follow the feedback very much.	Customisation, independent learning, information seeking strategies
Li	Just Google or check API. Use it for getting help. Spends 70% online learning resources to answer particular questions. Use textbook to learn theories. Feedback is very important. Tells me what to do and save time, Helps to switch between physical world and online learning activities.	Customisation, independent learning, information seeking strategies
Lu	I Google for help but I often don't know how to describe my problem properly. Online videos are helpful for that. Feedback is important to improve my learning.	Customisation, independent learning Personal trust, information seeking strategies
M	Google for help. Feedback affected my learning activities I guess.	Customisation, independent learning No clear strategy

Table 9.10 Responses and Codes for Online Learning (Question #10)

11. **Any expectations of the academics? (What support or resources do you expected for the academic staff?)**

    None expected more from the academics than they received. They mainly solved their problems in other ways and only consulted lecturers or demonstrators as a last resort.

SUBJECT	RESPONSE	CODE
Li	Maybe some anonymous online platform where questions could be posted and no one judges, just helps.	Fear of judgement, possible lack of confidence, caution
M	No, lecturers are busy. I wouldn't bother them unless there was no other way.	Implied respect
A	No	N/A
DK	No	N/A
HS	No	N/A
J	No	N/A
K	No	N/A
Lu	No	N/A

Table 9.11 Responses and Codes for For Expectations (Question #11)

The answers to these questions are remarkably consistent which lends a lot of validity to the interview process and the uniformity of the outcomes of the course. There are no diametrically opposite responses to the questions.

Some responses suggested fear of judgement, and implied respect for the time of the teachers. However, it is not possible to gain insights from those responses which are limited to a simple negative.

9.5 Summary

Assistance: Both groups made similar choices in their sources of assistance – online, classmates and friends were the preferred sources for assistance in solving a problem. Neither group relied predominantly on the course materials for learning to program.

Course materials: They utilised the course materials that were provided to them when they felt they needed to, but they chose whatever type of support they needed on a case-by-case basis.

These learning behaviours provide a basis for answering research questions B and C. Students do modify their approaches to learning to meet their needs and they rely on the course materials and exercises when it serves their purposes. In this manner they are clearly taking responsibility for their own learning and being accountable for their own results, a very mature behaviour.

From these interview data, one could conclude that the respondents enjoyed computer programming and believed that they had received a good learning experience overall.

They were generally satisfied with the course materials, but sought to personalise their learning by drawing from additional resources as needed. Additional resources typically included friends, fellow students or online resources.

They didn't have any strong opinions as to whether prior programming experience or math skills were an advantage in learning programming, however, no one believed they hurt. For some, fear of embarrassment that they might ask silly questions or be unable to accurately state what their problems were caused them to seek online help or help from other students, rather than consult lecturers or demonstrators. Each of them had positive remarks for the value of feedback. It saved them time and corrected their thinking so that they can avoid making future mistakes.

10 Conclusion and Future Work

This chapter focuses on the findings of conducted experiments, what was learned during the course of the study and what the implications are for any future work that might be conducted to add to the body of knowledge.

The results of the study show that the personal backgrounds of students do have an effect on how they learn and the type of learning resources they draw from. With this knowledge, educational institutions are better equipped to understand students' needs and perhaps tailor educational experiences in a way to maximise student learning.

The basic research design of this study embraces the principles of a robust empirical investigation of an important subject of inquiry. What and how do first year undergraduates in computer programming learn to program in a connected world. This study was designed to use a combination of qualitative and qualitative research techniques that use various measures of the educational process. The teaching methods used a combination of lecture, computer laboratory, experiential learning and various reference media.

The research measurements included questionnaires, direct observation, personal interviews and participatory data gathering techniques. The intent was to collect data using a variety of methods, collected at significant times and integrated in order to present as comprehensive view as possible of the learning experience. The goal of the inquiry was to answer the following questions:

A. Considering first year undergraduates: before being formally introduced to learning to program at university, do any differences exist, within specific sub–groups of students, between the attitudes to and beliefs about the task of learning to program?

B. Whilst learning to program in this connected world, to what extent do different students:
   a. Gain or lose confidence and motivation?
   b. Modify their approaches to learning as they progress?

C. How do students integrate their formal and informal learning activities? To what extent do students:
   a. Rely on formally provided materials and exercises?
   b. Develop individual approaches to learning which integrate real world and online activities and materials?

These questions provided the basis for the design of the study and the criteria for the measurements that were taken over the course of the study.

These assumptions were made about the nature of the learning experience and measurement process:

- The introductory programming module at universities is a system and process with a beginning and end that can be measured.
- Measurements can be taken at various times in the module that can provide insight into the effectiveness of the learning process and its effects on students.

- Assessing the interactive effects between the various delivery systems, the instructional media used and student responses is an important method for determining the module's effectiveness in fulfilling its obligations to the student and to the university.
- Multiple measurement techniques, when combined, can provide a more reliable and valid characterisation of what actually transpired in the module of the educational process, as opposed to what was intended.

10.1 Answers to the Research Questions

The analysis of experiments/surveys aims which was to provide answers to the questions so that the information obtained could be used to improve the teaching-learning process. Information gained as to the effectiveness of the instructional system and the students' response to it will be used to modify the system to better serve the interests of key stakeholders – the students, the instructors and the university.

The following information will consist of a statement of each research question and the methods that were used to answer the questions.

Question A **- Considering first year undergraduates: before being formally introduced to learning to program at university, do any differences exist, within specific sub–groups of students, between the attitudes to and beliefs about the task of learning to program?**

This question is based on the assumption that we live in a dynamic environment and that there is an interaction between the learning programme (stimulus) and the student's response, which is influenced by his or her background, attitudes and beliefs. The response is unique to each individual, but among like-minded individuals or groups with similar backgrounds, the response to a course of instruction, which is a shared experience, can be similar. This is consistent with the process mentioned by Kolb in which the learning process is described as a cycle of learning activities that move through various stages until the cycle is completed (Kolb 1984; Kolb 1985). In the present situation, the instructor initiates a lecture or an assignment involving such topics as code review, live programming demonstration, or coding activities. After which the student is expected to respond in some expected ways. At this point, the unique information-processing capability of the student comes into play. What happens next is a function of the student's background, attitude, and beliefs that guide his or her actions. It is described the optional learning strategies as being up to the learner to engage in surface learning or deep learning (Biggs et al. 2001). The choice to engage in rote learning that involves minimal effort, memorisation and the goal of simply passing is an example of surface learning. Deep learning is exemplified by a commitment to develop a comprehensive understanding of the subject matter and to excel, not just get by.

There are several indications of how students' assessment of their personal abilities affected their outlook on future performance in the programming course. Since behaviour is affected by preceding events, one would expect that future performance would draw from those past experiences. In the present study, students were asked what their prior programming experience was and what their outlook for the course was.

The following hypotheses were made about Question A using the results obtained from the initial student survey responses as reported. Based on the sample size and disproportionately low number of females and respondents over 21 years of age, it is unclear as to the representativeness of these results of the general population, but they may be more representative of first year programming student populations.

1. Beliefs and Prior Experience

According to Section 5.6.1, the following statements (B1 – B5) have been tested:

B1. Students that have prior programming experience will not inevitably result in confidence of their abilities to master the final exam and more motivated to learn to program.

B2. Students that have no or little prior programming experience will worry that they have to work hard and learning to program will be difficult.

B3. Students who express a motivation to learn will be more confident of their ability learn to program.

B4. Students who believe that programming is easy will express less motivation to work hard.

B5. Students who believe that programming is easy will express more confidence in having mastered programming by the exam.

B1 to B3 provide insight into the different mindsets of the students. These data suggest that motivation is a stronger factor than prior experience when it comes to determining the amount of confidence students have in their ability to learn to program. Respondents with little or no prior programming experience expressed more confidence to master programming by the final exam than did those with a lot or extensive experience (83.8% vs. 81.8% respectively).

While prior programming experience may have made the course easier from some students, it probably means that, as time went on, motivated students acquired the requisite knowledge and experience and the gap between the two disappeared, a tribute to the success of the learning experience.

Amongst the respondents who believed that programming were easy, 78.2% of them believed they would still need to work hard, whilst 91.1% of them expressed confidence that they would master programming before the final exam.

2. Demographic Factors

According to Section 5.6.2, the following statements (A1-A4, G1-G4, R1-R4 & C1) have been tested:

A1. Age has some effects on opinions about the difficulty in computer programming.

A2. Age has some effects on opinions about the ability have mastered programming by the exam.

A3. Age has some effects on the willingness to work hard to learn to program.

A4. Age has some effects on opinions about the motivation in computer programming.

Although current evidence could not support the assumption that age has a relationship with students' willingness to work hard for learning to program, age does make a difference in terms of students' motivation when considering opinions about the level of difficulty for learning to program or the ability to master exam. Respondents under age 21 felt motivated (93.5%) whilst 79.3% of them over age 21 who were motivated. This is unusual in that it apparently contradicts the generally held view that mature students are more highly motivated than recent school leavers.

G1. Gender has no effect on opinions about the difficulty in computer programming.

G2. Gender has no effect on opinions about the ability have mastered programming by the exam.

G3. Gender has no effect on the willingness to work hard to learn to program.

G4. Gender has no effect on opinions about the motivation in computer programming.

Current evidence shows that there were no differences that gender made as to initial motivation, mastering programming by exam time or willingness to work hard to learn.

R1. Region of origin has some effects on opinions about the difficulty in computer programming.

R2. Region of origin has no effect on opinions about the ability have mastered programming by the exam.

R3. Region of origin has no effect on the willingness to work hard to learn to program.

R4. Region of origin has no effect on opinions about the motivation in computer programming.

Region of origin did make a difference in terms of the attitudes of level of difficulty in these respondents. Students from the European Union stated that programming was easiest (69.6%), followed by UK and International Students (58.5% and 57.7% respectively). Current data collected for this project could not support that there are any relationships between region of origin and, students' opinions about ability to master final exam, their motivation and their willingness to work hard to learn to program, but it might be something to include in a future study.

This was not a homogeneous group and the sample size was small and skewed in some respects. Out of 169 total respondents: there were only 27 women. There were just 29 students over the age of 21 and those students who self-identified as having "a lot or extensive" prior programming experience numbered just 22. Similarly there were 23 EU students and 52 international students. Taken together it seems likely that the research finding might be different with a larger and more balanced population. Many of the summary statistics were not statistically significant, so conducting the study on a larger population with a demographic composition nearer the actual proportions could conceivably result in some different conclusions.

C1. Student enrolled in different degrees have no difference on opinions about the motivation in computer programming, the willingness to work hard and the ability to have mastered programming by the exam.

Current evidence shows that there is no relationship between the degree that students enrolled and their initial attitude, motivation and beliefs.

Question B **- Whilst learning to program in this connected world, to what extents do different students:**

a) Gain or lose confidence and motivation?

b) Modify their approaches to learning as they progress?

This question has to do with the motivational patterns of respondents when dealing with learning issues that were encountered during the training programme. If they encounter an issue that is not able to be resolved quickly or easily, how do they deal with it? What resources do they seek out for help and what problem-solving solutions do they use? The answers to these questions come from the pilot study, the responses from the initial, weekly and end of term surveys, the Nominal Group Technique meeting and individual interviews. The following information was obtained from several sources and is indicative of the range of responses that were made.

From the pilot study (Section 4.4.2.2), an international student commented as follows: “It becomes much easier to learn new stuff. I found looking at a tutorial on YouTube very useful since I'm International student going to lecture really wasn't very helpful (just in the beginning).” This is an example of a student looking to an outside source (YouTube) to deal with questions or issues rather than using course materials or contacting the lecturer. This was not only more convenient but avoided potential embarrassment by exposing their lack of knowledge.

Weekly observation (Chapter 6) showed that, getting stuck with some theories may cause loss of confidence. Once the problems have been solved, the students would be highly motivated. However, if they failed to sort those out, the learned helplessness they suffered may lead to de-motivation.

The weekly surveys produced some examples of internal attitudinal changes over the course of the 10 weekly labs. For example, when asked how the respondent would rate the difficulty of each lab (e.g. easy, difficult, mixed easy/difficult and hard to begin with), the weekly ratings showed increases and decreases week by week with the middle weeks (weeks 4 – 6) showing a general increase in perceived difficulty. Comparing week 1 with week 10 (section 6.3.2) shows the following: those who rated the 2 weekly sessions easy decreased from 32.8% to 27.0%; those who rated the sessions difficult increased their ratings from 7.6% to 12.0%; the mixed/easy ratings declined from 27.7% in week 1 to 13.0% in week 10; those who rated the lab sessions as hard to begin with showed a decrease in difficulty from 18.5% to 6.0%.

Similar fluctuations were found when assessing the weekly sessions in terms of a 4 point scale: (easier than expected, harder than expected, interesting and challenging). The interpretation of those results (Section 6.3.2) is that at the end of 10 weeks, the respondents found the sessions not as easy as expected, harder than expected, less interesting and less challenging.

The easy/harder measurement suggests that they had gains in competency that affected their perceptions as to how easy or how hard programming is currently. The interest/challenging dimension suggests that their exposure to programming affected their sense of challenge and interest as well.

Their attitudes toward Java in terms of liking/loving it didn’t change much. Those who liked it/loved in from the beginning continued at about the same rate. Those who felt “Ho Hum” or “Don’t like it” didn’t change significantly either.

When asked what the estimated their final grades would be (i.e. First, 2.1, 2.2, pass or fail), the comparison between week 1 and week 10 shows: expected First grades fell (38.7% to 32.3%);

expected 2.1 grades were relatively stable (46.2% vs. 45.8%); expected 2.2 grades increased (7.6% vs. 11.5%); pass ratings were relatively stable (5.0% vs. 5.2%); and expected fail grades doubled (2.5% vs. 5.2%). In interpreting these results one needs to consider that week 1 estimates were based on personal opinions with no relevant data to base them on. After 10 weeks the students had actual experience upon which to make those judgments.

These attitudes about the sessions, about Java and perceptions about their progress in terms of expected final grade are indicative of changes occurring in students' internal perceptions, perhaps resultant of their knowledge, skills and understanding changing over this time span. The module had an effect on each student, perhaps some more than others. With these internal changes in attitudes and outlooks toward personal progress, one can surmise that they are also accompanied by changes in their problem-solving skills and use of external resources.

Whilst most students expressed satisfaction with module materials, many found the need to supplement them with other resources, such as those available from the internet. For example, over 70% of respondents to the end of term survey said they consulted the internet frequently (section 7.3) in reference to lectures/labs - before (60.9%), during (65.2%), and after (78.3%). An overwhelming number of students (90%+) use Google or Bing, nearly double the second place choices (e.g. university provided materials via the ECS intranet, Edshare). The majority of students (over 60%) use the Web when they are struggling with a problem arising from a lab or coursework or their learning a new theory. This is supported by the data gathered during the weekly observation. The amount of students who searched online when struggling increased from Week 5 onwards (Section 6.3.1 & Appendix E).

Other uses of the Web as were noted previously include: extending knowledge of theory, personal programming projects, dealing with bugfixes, syntax and other current issues. These are illustrative examples of the way students adapt their learning method to suit the circumstances, which is an aspect of Research Question B. These adjustments obviously vary with the circumstances as they evolve over time. An indication of this was reported earlier but is shown in the Figure 10.1 below in graphic form. One can see that students had different reactions to each week's learning experiences. The trend lines are in the direction one would prefer in an end to end learning experience – becoming easier, less difficult and not as hard to comprehend as before.

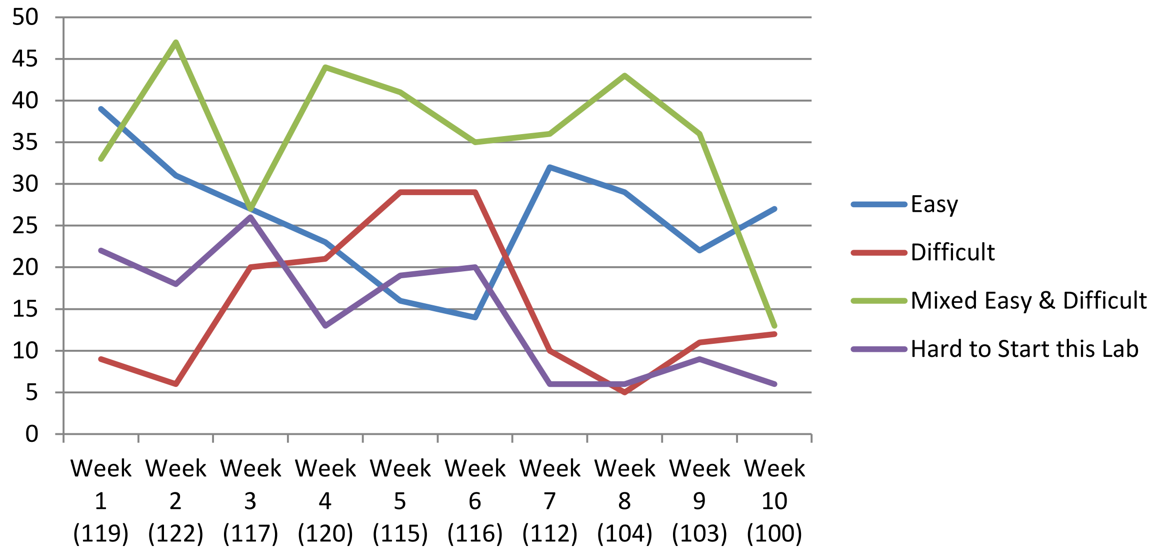

Figure 10.1 Example of Changing Attitudes Towards Lectures and Personal Abilities

The written comments in the open ended questions (Section 7.3) largely gave the established learning and teaching methods positive marks and constructive suggestions as to what can be done to improve the outcomes. They ranged from making no changes at all to suggestions for supplemental resources, largely drawn from the Web.

The evidence suggests that motivations wax and wane according to the situation at any given point in time but students make adjustments as they see fit and largely end up with a positive experience. Meanwhile, conflicting deadlines, the impact of which was noted via the observations in the teaching labs, may influence, but it is not a serious problem

In addition, weekly observation and individual interviews also shows that the formal feedback (including those for practical labs, coursework, etc.) is an important factor which might lead the students to modify their learning approach. Appropriate feedback prompted students to customise their learning and gain a supported transition towards independent learning. This suggests that the students are presenting behaviours consistent with Marton and Säljö's (1976) deep approach to learning (Section 2.1.3).

<u>Question C</u> **- How do students integrate their formal and informal learning activities? To what extent do students:**

a) Rely on formally provided materials and exercises?

b) Develop individual approaches to learning which integrate real world and online activities and materials?

The answers to these questions involve an examination of the usage of programme supplied reference materials and the use of adjunct materials and the Web. From the information obtained in this study, it shows variability and flexibility as to how students use the standard course materials and how they supplement them as needed to meet their own specific needs. Some found help in other textbooks (e.g. in their first language), but the major adjunct learning resource is the Web for its convenience (90%+) and ease of finding information (where, of course students may have been making use of this in their first language too). Use of and reference to additional resources were identified in the NGT group who, like the previous surveys, also ranked Google searches as a first step in solving thorny problems, closely followed by asking a friend or expert (Section 8.4.4). This is the same conclusion students in the initial survey came to.

An equal number of students appeared to rely on formal information and materials as did on informal/self-identified sources of support. According to the interview (Section 9.4), students who have a sound foundation of prior programming experience relied less on formal materials, compared to novices and students with a little programming experience.

The table below shows that even students with more programming experienced use the Web on a frequent basis.

When using Web	Level of prior programming experience			
	None	Some	A lot	Overall
Before lectures/labs	33.3%	64.0%	88.9%	60.9%
During lectures/labs	83.3%	64.0%	44.4%	65.2%
After lectures/labs	75.0%	76.0%	88.9%	78.3%

Table 10.1 Use of the Web in Conjunction with Lectures and Labs

This table shows how students with varying levels of expertise use the Web.

1) Those students who have a lot of prior experience of programming make much greater use of the Web before lectures than those with no experience, nearly 90% compared to 33%.
2) During lectures those without background experience in programming use the Web a lot more than those who have experience.
3) Use of the Web after lectures does not differ so dramatically with the majority of all students (over three-quarters in each case) using the Web, with a greater proportion for those with prior experience.

It may be, that once students progress through their degree programme, the pattern of usage observed in those with prior experience becomes widespread across the student body. However it was not possible to identify that from the collected data.

Using the Web is a method of choice for problem-solvers who are seeking clues or ideas when tackling new or difficult problems, but whatever outside resources are used require critical evaluation of the findings as well. This was the consensus of the NGT group (section 8.4.5). Critical analysis of any information is a sign of maturity, especially in computer programming. Research Question C appears to be answered in the affirmative. Students do show evidence of adapting their use of resources over time and use the Web and personal contacts for assistance as the situation requires.

The interviews showed that students make whatever changes are necessary in order to learn what they need to learn. This manifests itself their decisions to utilise the module’s instructional materials or seek other texts or online help. This typically happened with students who had prior programming experience with Basic or Python or who simply found resources they found more useful. One respondent (Li) said that she stopped using the text halfway through the course and another student (HS) said he never used it.

Focusing on Question C.b: to what extent do students develop individual approaches to learning which integrate real world and online activities and materials?

Most students were unclear about the question, because they have no awareness about whether they were integrating “something”. Nonetheless, some typical learning “modes” could be identified as following:

1. Some students learn in real world (e.g. going to lectures, reading textbooks, etc.) at the beginning. They will get help online when help needed (e.g. confusion about theories and practical exercises).
2. Some students preferred to learn theories online. However, when help needed, they would rather ask experts (demonstrators, lecturers, classmates, etc.). The main reason is that they have no idea how to describe their problems and what to search aiming to get the information they need. Some argued that it would be much faster if they can show their codes to someone face-to face.

Apart from these, many students (Section 9.4) indicated that, feedback is the key factor, which not only influence their modification of learning approaches (Research Question B.b), but also affected them to switch from online learning activities to “offline” ones. The main reason is that, when they first introduced to program, they have not generated the most appropriated way to learn to program. They would follow the feedback strictly in order to at least get more marks.

10.2 Evaluation

The overall research design appears to have provided a means to gather interesting and useful data for each of the research questions. There is ample evidence to support the notion that student backgrounds, attitudes and beliefs do affect their approaches to learning to program (Q1), albeit some only slightly. On a percentage basis, the greatest difference on any of the factor tested by the hypotheses was regarding A1 (Age has some effects on opinions about the difficulty in computer programming) and R1 (Region of origin has some effects on opinions about the difficulty in computer programming). A1 showed a difference of 13.8 percentage points between respondents under age 21 (62.1%) versus 48.3% of respondents over age 21 who felt programming was easy. The second largest difference was also in regard to the ease of learning to program that showed an 11.9% difference in ratings by EU students (69.6%) versus 57.7% by International students. Most other difference scores were in usually in the 3 – 5% range or less.

There is also substantial evidence to support the notion that students change their learning approaches to suit the situation (Q2). This is evident in how they choose to approach solving problems. Rather than using course-supplied materials only, for the majority of students, their first recourse is to seek online help, mostly on Google, followed by fellow students and friends. The fact that they seek out lecturers or demonstrators only rarely is a commentary not only on their fear of embarrassment but their willingness to solve their problems in their own way and take responsibility for the results.

Fear of embarrassment is a function of individual personality as well as the nature of the social context the lecturer and others are able to construct. The teacher is ultimately responsible for creating a learning environment wherein students feel comfortable in exploring any learning issue freely and openly as much as possible.

As to whether students gain or lose confidence or motivation, the collected data is largely circumstantial and inferential. The survey questionnaires were self-reports. The scales may not have accurately measured their true feelings or precisely measured the implied behaviours. Such is a limitation of these type of studies; however, all of the measurements taken over the course of the study are quite consistent. Use of alternative texts and online resources were noted somewhere in every sampling of study methods that were taken, including the personal interviews.

The ease and flexibility of access to solutions was confirmed on several occasions. This speaks to their conscious and deliberate acts to solve problems in their own way, which is an aspect of motivation which is obviously the basis for different courses of action that suits the need and the situation.

The integration of formal and informal activities (Research Question C) occurs every time a student deviates from using lectures, labs and course-supplied materials to involve online, social networks and other methods to solve learning problems. As has been reported, this is a sign of maturity and a manifestation of what has been termed “self-efficacy” and “self-regulated” behaviour. These behaviours are characterised by students assessing the situation, determining a course of action and staying with their plan until they have achieved their learning objectives. Based on all the data points, one can assume that this is occurring from one degree or another by each student. The precise metrics for measuring these behaviours are beyond the scope of this study but would seem to be a valuable addition to a future study as a follow on to this one.

However, there are some issues that are worth noting and are appropriate for comment and analysis, in terms of some things that might have been done differently, in hindsight, that may have influenced the results positively or negatively.

1. The findings from each data collection source are remarkably consistent.

The surveys and interviews corroborate one another. They all point to the same conclusions with each measurement. Respondents assume responsibility for their own approaches to learning and use whatever resources they chose in order to address whatever learning issue they are confronted with. This is consistent with what Lefcourt referred to as “locus of control” (Lefcourt 1982). Students who modified their approaches to learning to use the supplemental materials of their choosing were exercising internal controls. They were also exhibiting self-regulatory behaviour that Zimmerman described as when an individual organises his or her learning strategies to accomplish the desired results within the time frame available (Zimmerman 2000).

2. The pilot study was a great idea as a means of testing the questionnaire before administering it to a larger population.

As a rule, once a survey questionnaire has been administered, there are no second chances to correct an error that may have occurred in its design. In this case, the offer to participate in a pilot study was limited in its participation, due to a scheduling conflict that was attributed to the lateness on the project's administrative approval. This delay subsequently caused a conflict with other student requirements. If this had been avoided, there would have likely been a much larger participation rate than the 8 students who were involved in the pilot study. This may have had a significant effect on the amount of pre-project information available for inclusion in the remainder of the study.

3. **It is possible that more effective use might have been made of the NGT results if they had been done earlier in the study.**

The NGT is a powerful tool for engaging groups in more in-depth analyses of issues where there are no ready answers and for gathering information resides in the minds of groups who share a common connection or interest. In this case, perhaps the NGT technique could have been conducted a bit earlier based on the assumption that the results may have provided insights into issues that might have affected some questions that were later asked in questionnaires and in the subsequent interviews.

4. **The data gathering and statistical procedures that are used have an effect on what information is available for analysis and interpretation.**

For example, aggregating interview data is problematic without using a very sophisticated analytical technique like thematic analysis, discourse analysis or one of the other qualitative analysis methods. In which case, great care needs to be given to following protocols so that themes can be more easily identified and analysed for message content. There is also the possibility of using new technologies to fully automate survey administration and analysis that could make use of the most powerful statistical analyses possible. Crosstabs are instructive for displaying inferential statistics like means and frequencies. There are also powerful statistics that link individual ratings with a broad array of independent variables that could be incorporated into the data analysis model that includes such analyses as multivariate analysis, regression analysis, factor analysis and validity checks.

5. **There are also other learning programme evaluations (in both the public and private sectors) that might be used as models for future studies.**

Future programme evaluation techniques might include the design and construction of a valid and reliable measurement instruments that uses behavioural anchored rating scales that are easier to verify than "Ho Hum," "Interesting," "Challenging," and "Love it or Like it." Since these terms were not defined, their meanings may have been interpreted differently by each individual thus making it difficult to be assured that the results are always reliable.

As previously noted, concepts such as "self-efficacy," "self-regulation," and "deep learning" have been used to differentiate learners and approaches to learning and could be used as variables for determining who excels in the modules and why. With valid and reliable instruments to classify students, it might also be possible to create alternative modules that are designed based on qualities of the student as well as how best to present the instructional material.

This project might be replicated as a whole or in part for the University's future programming classes, or in programming classes at other institutions. The results of this study would likely serve as a basis for modifications that could be made to achieve better results for students who participate in these modules.

10.3 Contribution

This thesis explains in detail a three-year long research investigation of first year undergraduates' approaches to learning to program at the University of Southampton.

The introductory programming module that was studied consisted of lectures, demonstrations, labs, textbooks and other learning aids.

The research traces student progress within the module using observations, surveys, interviews and group discussions. These were used together to drill down and triangulate findings which addressed three research questions focusing on "what students do" with three specific focus areas of:

- Differences in students' background and prior experience.
- Attitudes and approaches to confidence and motivation.
- The comparison and relative importance of formal and informal learning resources.

A detailed presentation of the findings was made in Section 10.1, key contributions are listed below:

- Current evidence shows that gender and the degree types have no effect on students' initial attitudes and beliefs towards learning to program. Before the module, students under 21 felt more motivated than those over 21. International students worried the most about that learning to program would be a difficult task, whilst EU and UK students have less worries.
- Students tended to lose confidence and motivation when they were struggling with some theories. Once they addressed that problem, they appear to have become highly motivated. However, if they failed to sort those issues out, this appears to have led to a loss of confidence and motivation – demonstrating learned helplessness.
- Students with little or no prior programming experience appear to rely much on formally provided exercises, whilst pre-experienced students have more interests to learn in their own way.

The study yielded some important insights about how students respond to a standard course curriculum and how they make personal adaptations to learn programming in ways that make the most sense to them. In so doing, the research offers some potentially valuable insights into the educational process involved whilst teaching undergraduate student introductory computer programming courses.

Following the structured framework of evaluation provides a systematic way to gather evidence which can be used to review or change the overall learning design in the following ways:

1. Identifying what additional efforts are being made by students to adapt their approaches to learning and resources to meet the challenges they encounter.
2. Analysing how the collected information might be used to modify the programmes, if needed.
3. Making modifications to teaching strategies. For example, aiming to maximise the benefits of the Web, academics might consider providing two alternative and complementary online platforms for novices and pre-experienced students. One which focuses on basic theories might be more suitable for novices, whilst another offering more challenging tasks and complex explanations might provide additional challenge and stimulation appropriate for pre-experienced students.

These proposals are in effect a suggestion to mirror within online resources the differentiated approaches which are often found within the face to face context of lecture theatre and laboratory.

A further contribution from this work exists in the research framework (Figure 11.1) which may be valuable to academics seeking to evaluate and update educational approaches. The framework is probably most relevant to introductory modules, and may be useful, not only in computer science, but across a far broader range of academic disciplines.

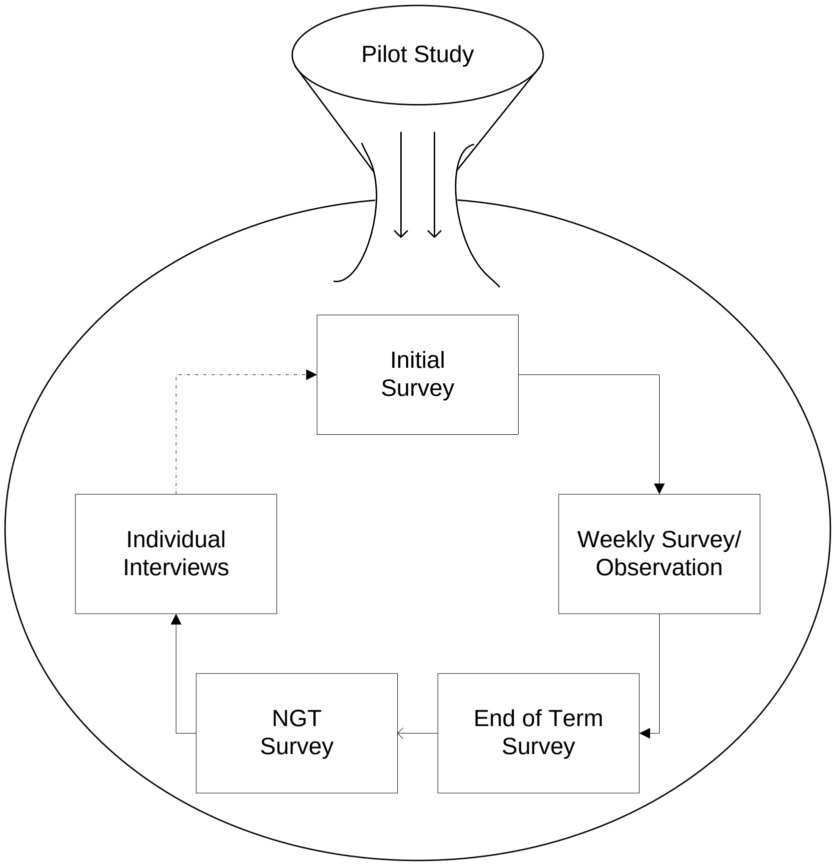

Figure 10.2Research Framework

The particular strengths of the framework are listed below:

1. It is efficient. This framework allows academics to understand what their students do in an introductory module at universities within just one semester timespan. It can be quickly applied when needed, and all the methods of data collection are straightforward to prepare and establish.
2. It is generic. There are no aspects of the process which are specific to the discipline studied in this thesis. This research framework can quickly applied to other introductory modules at universities. Minor changes may be relevant to match the specific structure of the module components. For example, academics in finance or management adopting the framework may not have identical timings or practical labs, so the weekly survey might be modified as after-lecture paper form surveys depending on the detailed curriculum structure.
3. It provides a mechanism to establish regular review, which may be essential given the general environment of rapid technological change. Understanding whether student's approaches and attitudes are changing, and tracking the extent and nature of that change may be more effective

than periodic review, and can enable fine tuning of approaches rather than more disruptive radical overhaul and change.

10.4 Future Work

According to all the responses from the series of surveys, there is tentative evidence that it might be possible to identify specific "learning modes" which describe what our students do into categories. For instance (details in Section 10.1),

1. Some students read textbook, lecture slides and provided notes to learn new theories. They will go and get help online when they get stuck. They will not ask anyone until they cannot get useful answers from the Web, and they would rather ask their friends to avoid looking "stupid" by the lecturers.
2. Some students do not rely on the Web at all. They prefer reading printed materials and asking whoever they can get help from.
3. Some students may love to learn online but prefer not to discuss at online platforms, because they believe that they are not capable to describe their problems clear. They would rather have face-to-face discussion with experts (e.g. demonstrators, friends, etc.) by showing their code.
4. Novices and students who have a little prior programming experiences have a higher reliance on formal provided materials and exercises, than those students who already had a sound foundation of programming skills before university.

The current evidences are not strong enough to firmly identify all the modes. However, it will be interesting to have it investigated in the future.

Meanwhile, it might also be important to examine why some students fail the final examinations. What kind of students may fail and the specific learning experiences of those students are worth being investigated.

In terms of the research methods, future research might include a better technique for profiling students than self-reports or some comparison and calibration of self-reported attributes with more established objective measures. Perhaps there could be a design that used a more sophisticated identification of student attributes for greater accuracy and comparability. Another possible change is to divide the group into 2 treatment groups who receive slightly different educational experiences. For example, it might be fairly easy to use this design in measuring the educational outcomes in courses that teach other programming languages.

1. This type of study lends itself to additional quantification of results, even using the qualitative approach. Consideration should be given to pre-post measurements wherever possible to improve the power of the analysis of the results. This research design lends itself to any educational process that uses a combination of lectures, labs and prescribed educational materials. It also lends itself to evaluating the use of different concurrent instructional designs. Were this process to become the start of longitudinal studies, it could very well lead to innovations in computer programme education that could make a significant contribution to the body of knowledge that appears to be on the verge of an explosion due to advances in educational technology.

2. There are some unanswered questions that might be a source of future study.

1) Is the lack of prior programming experience a detriment or disadvantage in the course?
2) Is the classroom language a problem and what can be done about it?
3) Why do “reading text books” and “asking lecturers” rate lower than classmates, friends, group study and asking demonstrators as a source for solving problems?
4) What is the relative value of the “offline universe” versus “online universe” as a means of solving learning problems?

A series of issues about the learner’s readiness and characteristics to learn as well as in which way they are likely be addressed in the classroom are mentioned in these questions, which also confirm outside the classroom learning and resources must be within the scope of reaching while necessary. It seems to prefer the use of materials and online resources lecturers and students of social networks, the demonstrators when they need help to solve problems. In part, this can be incorporated into learning design because they may improve the quality of the learning process and results.

It is undoubted that early experience of programming can contribute previous learning models. However, it is predicted that there is an era when these concepts and topics are fresh for everybody. It is probably an excellent measure of this additional value as in what way students do at this time, for example, a previous programming experience whatever. It may be another measurement in later experiment.

According to the report, respondents did not believe classroom language to be a problem, but one respondent did identify reliance on textbook in his own language. This may be indicative of a problem, that is, those for whom English is not their first language may not have the English language capacity needed for active participation in the learning process without additional language support. An objective way to measure the language capacity before the training can be adopted as a standard of a sequence of other population variables available in evaluation in terms of their probable impact on classroom performance and post-lecture/lab interactions’ frequency and type with both demonstrators and lecturers. Language proficiency is also likely to influence the student’s willingness to make contact with demonstrators and lecturers.

Language barriers are also an explanation of why textbooks were not used by some EU country or International students, who did not make much contact with lecturers either. Although this was unlikely to be the student’s conscious decision, furthermore, the student would not know how to measure those decisions’ impact on the final outcome of learning. It was predicted that most online access, as well as consultations with friends and classmates were carried out in English, the capacity of which would be a variable or instrumental in the judgment of learning results regardless of how hard the student tried to compensate for the deficiencies that were already known. In the case of any courses where there were sufficient numbers of students who had language barriers it would justify modifying the instructional programme so as to address those particular needs

The issue of offline/online universe’s utility seems to be a matter of convenience as well as easy access. The Internet is usually available 24 hours and 7 days a week, allowing for flexible last minute scheduling. Although due to limited time and space, there are limited offline resources, it is undoubtedly that friends and classmates are more easily accessed than lecturers and demonstrators. A student’s access to the Internet determines online accessibility. Researchers may investigate the modification of lecturers and demonstrators’ availability and accessibility. Nevertheless, if language capacity is the essential issue, providing greater ease of access might not make any difference in student behaviours.

Taken together, the conclusions and future work of this study, suggest that there is useful and valuable information to be gained from a detailed observation of the ways in which students go about learning to program. There are already ways in which learning from the study have been incorporated into existing educational approaches at Southampton (particularly in the lab classes).For example, there is an online student wiki page for all our students to discuss their problems whilst learning to program. The university has set up a help desk in the computing lab where students could ask any questions about learning to program during weekdays. It is to be hoped that continued study of this area and wider discussion of the methods and outcomes will be sustained.

References

Adams, J.C., Ernst, D.J., Wisconsin, U., Claire, E., Murphy, T., Ortiz, A., Adams, J., College, C. & Ernst, P.D., 2010. Multicore Education : Pieces of the Parallel Puzzle. In *Proceedings of the 41st ACM technical symposium on Computer science education - SIGCSE '10*. pp. 194–195.

Ala-mutka, K., Uimonen, T. & Järvinen, H.-M., 2004. Supporting Students in C ++ Programming Courses with Automatic Program Style Assessment. *Journal of Information Technology Education*, 3, pp.245–262.

Ames, C., 1992. Classrooms: Goals, structures, and student motivation. *Journal of educational psychology*, 84(3), pp.261–271.

Barker, L., 2009. Student and Faculty Perceptions of Undergraduate Research Experiences in Computing. *Transactions on Computing Education*, 9(1), pp.1–28.

Ben-Ari, M., 1998. Constructivism in Computer Science Education. In *Proceedings of the 29th SIGCSE technical symposium on Computer science education*. New York, pp. 257–261.

Biggs, J., 1995. Assessing for learning: Some dimensions underlying new approaches to educational assessment. *Alberta journal of educational research*, 41(1), pp.1–17.

Biggs, J., 2003. *Teaching for Quality Learning at University* 2nd ed., Buckingham: Society for Research into Higher Education/Open University Press.

Biggs, J., Kember, D. & Leung, D.Y., 2001. The revised two-factor study process questionnaire: R-SPQ-2F. *The British journal of educational psychology*, 71(Pt 1), pp.133–49.

Boyer, K.E., Dwight, R.S., Miller, C.S., Raubenheimer, C.D., Stallmann, M.F. & Vouk, M.A., 2007. A case for smaller class size with integrated lab for introductory computer science. *Proceedinds of the 38th SIGCSE technical symposium on Computer science education - SIGCSE '07*, p.341.

Boyer, K.E., Phillips, R., Wallis, M.D., Vouk, M.A. & Lester, J.C., 2009. The Impact of Instructor Initiative on Student Learning : A Tutoring Study. In *40th ACM Technical Symposium on Computer Science Education*. pp. 14–18.

Braught, G., MacCormick, J. & Wahls, T., 2010. The benefits of pairing by ability. *Proceedings of the 41st ACM technical symposium on Computer science education - SIGCSE '10*, pp.249–253.

Buckingham, A. & Saunders, P., 2004. *The Survey Methods Workbook: From Design to Analysis*, John Wiley & Sons.

Carlisle, M.C. & Educator, D., 2010. Using YouTube to Enhance Student Class Preparation in an Introductory Java Course. In *Proceedings of the 41st ACM technical symposium on Computer science education - SIGCSE '10*. pp. 470–474.

Carr, L., Davis, H. & White, S., 2005. AnnAnn - a Tool to Scaffold Learning about Programs. In *6th Higher Education Academy Conference on Infromation and Computer Science*. Available at: http://eprints.soton.ac.uk/259233/1/LTSN-ICS2004.pdf.

Carter, J., Bouvier, D., Cardell-Oliver, R., Hamilton, M., Kurkovsky, S., Markham, S., McClung, O.W., McDermott, R., Riedesel, C., Shi, J. & White, S., 2011. ITiCSE 2011 Working Group

Report Motivating All our Students. In *16th Annual Conference on Innovations and Technology in Computer Science Education*. Darmstadt.

Carter, J., White, S., Fraser, K., Kurkovsky, S., McCreesh, C. & Wieck, M., 2010. ITiCSE 2010 Working Group Report Motivating our Top Students. In *15th Annual Conference on Innovations and Technology in Computer Science Education*. Bilkent, Ankara, Turkey.

Caruso, T., Hill, N., Vandegrift, T. & Simon, B., 2011. Experience Report : Getting Novice Programmers to THINK about Improving their Software Development Process. In *Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE '11*. pp. 493–498.

Chen, T.-Y., Lewandowski, G., McCartney, R., Sanders, K. & Simon, B., 2006. What do beginning students know, and what can they do? In *Proceedings of ITiCSE'06*. Bologna, p. 329.

Cutts, Q., Cutts, E., Draper, S., O'Donnell, P. & Saffrey, P., 2010. Manipulating Mindset to Positively Influence Introductory Programming Performance. In *Proceedings of the 41st ACM technical symposium on Computer science education - SIGCSE '10*. Milwaukee, pp. 431–435.

Davis, H.C., Carr, L., Cooke, E. & White, S., 2001. Managing Diversity: Experiences Teaching Programming Principles. In *2nd LTSN-ICS Annual Conference*. London.

Denny, P., Luxton-reilly, A., Tempero, E. & Hendrickx, J., 2011. CodeWrite : Supporting Student-Driven Practice of Java. In *Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE '11 ACM technical symposium on Computer science education - SIGCSE '11*. ACM Press, pp. 471–476.

Dewan, P. & Hill, C., 2012. How a Language-based GUI Generator Can Influence the Teaching of Object-Oriented Programming. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education - SIGCSE '12*. ACM Press, pp. 69–74.

Dyke, G., Saint-Etienne, E.M. De, Fauriel, C. & Cedex, S., 2011. Which Aspects of Novice Programmers' Usage of an IDE Predict Learning Outcomes ? In *Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE '11*. pp. 505–510.

Eidelman, L. & Hazzan, O., 2007. Eccles ' Model of Achievement-Related Choices : The Case of Computer Science Studies in Israeli High Schools. In *Proceedinds of the 38th SIGCSE technical symposium on Computer science education*. Covington, pp. 29–33.

Entwistle, N., 2000. Promoting deep learning through teaching and assessment: conceptual frameworks and educational contexts. *TLRP conference, Leicester*, pp.1–12.

Fincher, S., Cooper, S., Lafayette, W., Kölling, M. & Maloney, J., 2010. Comparing Alice , Greenfoot & Scratch. In *Proceedings of the 41st ACM technical symposium on Computer science education - SIGCSE '10*. pp. 192–193.

Fowler, F.J., 1995. Improving Survey Questions: Design and Evaluation.

Gehringer, E.F. & Miller, C.S., 2009. Student-generated active-learning exercises. In *40th ACM Technical Symposium on Computer Science Education*. Chattanooga, p. 81.

Hamer, J., 2004. An approach to teaching design patterns using musical composition. *ACM SIGCSE Bulletin*, 36(3), p.156.

Hanks, B., Murphy, L., Simon, B., McCauley, R. & Zander, C., 2009. CS1 Students Speak : Advice for Students by Students. In *40th ACM Technical Symposium on Computer Science Education*. Chsttanooga, pp. 19–23.

Haungs, M., Clark, C., Clements, J., Janzen, D., Ave, G. & Obispo, S.L., 2012. Improving First-year Success and Retention through Interest-Based CS0 Courses. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education - SIGCSE '12*. ACM Press, pp. 589–594.

Hoffman, D.M., Lu, M. & Pelton, T., 2011. A web-based generation and delivery system for active code reading. In *Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE '11 ACM technical symposium on Computer science education - SIGCSE '11*. New York, New York, USA: ACM Press, p. 483.

Hooper, C.J., Carr, L.A., Davis, H.C., Millard, D.E., White, S.A. & Wills, G.B., 2007. AnnAnn and AnnAnn.Net: Tools for teaching programming. *Journal of Computers*, 2(5), pp.9–16.

Hundhausen, C., Agrawal, A. & Ryan, K., 2010. The Design of an Online Environment to Support Pedagogical Code Reviews. In *Proceedings of the 41st ACM technical symposium on Computer science education - SIGCSE '10*. pp. 182–186.

Jenkins, J., Brannock, E., Cooper, T. & Hall, M., 2012. Perspectives on Active Learning and Collaboration : JavaWIDE in the Classroom. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education - SIGCSE '12*. ACM Press, pp. 185–190.

Jenkins, T., 1998. A participative approach to teaching programming. In *Proceedings of the 6th annual conference on the teaching of computing and the 3rd annual conference on Integrating technology into computer science education: Changing the delivery of computer science education*. ACM, pp. 125–129.

Jenkins, T., 2005. How do they think they're doing? In *Proceedings of 6th Annual HEA-ICS Conference*. York, pp. 39–43.

Jenkins, T., 2001a. Teaching programming - A journey from teacher to motivator. In *Proceedings of 2nd Annual LTSN for Information and Computer Science Conference*. London, pp. 65–71.

Jenkins, T., 2001b. *The Motivation of Students of Programming*. University of Kent at Canterbury, UK.

Jenkins, T., 2001c. The motivation of students of programming. *ACM SIGCSE Bulletin*, 33(3), pp.53–56.

Kearse, I.B. & Hardnett, C.R., 2008. Computer Science Olympiad : Exploring Computer Science through Competition. In *39th SIGCSE Technical Symposium on Computer Science Education*. Portland, pp. 92–96.

Kolb, D.A., 1984. *Experiential Learning : Experience as the source of learning and development*, Prientice-Hall.

Kolb, D.A., 1985. *Learning Style Inventory*, McBer.

Lahtinen, E., Ala-Mutka, K. & Järvinen, H.-M., 2005. A study of the difficulties of novice programmers. In *Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education - ITiCSE '05*. New York, New York, USA: ACM Press, p. 14.

Laurillard, D., 2002. *Rethinking University Teaching*, London: Routledge. Available at: http://books.google.co.za/books?id=J-4b_vBQqAQC.

Laurillard, D., 1997. *The experience of learning : implications for teaching and studying in higher education* F. Marton, D. Hounsell, & N. Entwistle, eds., Edinburgh: Scottish Academic Press.

Lave, J. & Wenger, E., 1991. *Situated learning: Legitimate peripheral participation*, Cambridge: Cambridge University Press. Available at: http://books.google.com/books?id=CAVIOrW3vYAC&pgis=1.

Lawhead, P.B., Duncan, M.E., Bland, C.G., Goldweber, M., Schep, M., Barnes, D.J. & Hollingsworth, R.G., 2003. A road map for teaching introductory programming using LEGO© mindstorms robots. *ACM SIGCSE Bulletin*, 35(2), p.191.

Lefcourt, H.M., 1982. *Locus of Control: Current Trends in Theory and Research* 2nd ed., London: Lawrence Erlbaum Associates.

Lohmann, S., Ziegler, J. & Tetzlaff, L., 2009. Comparison of tag cloud layouts: Task-related performance and visual exploration. *Human-Computer Interaction–INTERACT …*, pp.392–404.

Marceau, G., Fisler, K. & Krishnamurthi, S., 2011. Measuring the effectiveness of error messages designed for novice programmers. In *Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE '11*. New York, New York, USA: ACM Press, p. 499.

Marton, F. & Säljö, R., 1976. On qualitative differences in learning I: Out-come and process. *British Journal of Educational Psychology*, 46(1), pp.4–11.

Nield, K., 2007. Understanding the Chinese learner: A case study exploration of the notion of the Hong Kong Chinese learner as a rote or strategic learner. *The Journal of Hospitality Leisure Sport and Tourism*, 6(1), pp.39–48.

Pask, G., 1975. *The Cybernetics of Human Learning and Performance*,

Petersen, C., Maier, S.F. & Seligman, M.E.P., 1993. *Learned Helplessness: A Theory for the Age of Personal Control*, Oxford: Ovford University Press.

Pintrich, P., 1999. The role of motivation in promoting and sustaining self-regulated learning. *International journal of educational research*, 31, pp.459–470.

Radermacher, A., 2011. Investigating the Effective Implementation of Pair Programming : An Empirical Investigation. In *Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE '11 ACM technical symposium on Computer science education - SIGCSE '11*. ACM Press, pp. 655–660.

Radermacher, A. & Rummelt, R., 2012. Assigning Student Programming Pairs Based on their Mental Model Consistency : An Initial Investigation. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education - SIGCSE '12*. ACM Press, pp. 325–330.

Ragonis, N. & Ben-Ari, M., 2005. On understanding the statics and dynamics of object-oriented programs. In *Proceedinds of the 36th SIGCSE technical symposium on Computer science education*. St Louis, p. 226.

Riordan, B. & Traxler, J., 2003. Supporting Computing Students at Risk Using Blended Technologies. In *Proceedings of 4th Annual HEA-ICS conference*. Galway, pp. 174–175.

Rogerson, C. & Scott, E., 2010. The Fear Factor : How It Affects Students Learning to Program in a Tertiary Environment. *Journal of Information Technology Education*, 9, pp.147–171.

Scaffidi, C., Dahotre, A. & Zhang, Y., 2012. How well do online forums facilitate discussion and collaboration among novice animation programmers? In *Proceedings of the 43rd ACM technical symposium on Computer Science Education - SIGCSE '12*. New York, New York, USA: ACM Press, p. 191.

Schmeck, R.R., 1988. *Learning strategies and learning styles* R. R. Schmeck, ed., Plenum Press.

Scribner, S. & Cole, M., 1973. Cognitive consequences of formal and informal education. *Science*, 182(4112), pp.553–559.

Shinners-Kennedy, D., Barnes, D.J. & Ct, K., 2011. The Novice Programmer ' s " Device to Think With " Categories and Subject Descriptors. In *Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE '11*. pp. 511–516.

Stamouli, I., Begum, M., Mancy, R. & Reid, N., 2005. ExploreCSEd: Exploring Skills and Difficulties in Programming Education. In *Proceedings of ITiCSE'05*. Lisbon, p. 371.

Stone, J. a. & Clark, T.K., 2011. The impact of problem-oriented animated learning modules in a CS1-style course. In *Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE '11 ACM technical symposium on Computer science education - SIGCSE '11*. New York, New York, USA: ACM Press, p. 51.

Stubbings, R., Franklin, G., Boden, D., Powis, C., Bent, M. & Woolley, M., 2009. SirLearnALot. In *Proceedings of 10th Annual HEA-ICS conference*. Canterbury, pp. 90–94.

Tyler, R.W., 1949. *Basic Principles of Curriculum and Instruction.*, Chicago: University of Chicago Press.

Vihavainen, A., Paksula, M. & Luukkainen, M., 2011. Extreme apprenticeship method in teaching programming for beginners. In *Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE '11*. New York, New York, USA: ACM Press, p. 93.

Vygosky, L.S., 1978. *Mind in Society: The Development of Higher Psychological Processes*, Cambridge, MA: Harvard University Press.

White, S., Carter, J., Jamieson, S., Efford, N. & Jenkins, T., 2007. TOPS - Collaboration and competition to stretch our most able programming novices. In *2007 37Th Annual Frontiers in Education Conference - Global Engineering: Knowledge Without Borders, Opportunities Without Passports*. pp. S2J–20–S2J–24. Available at: http://www.scopus.com/inward/record.url?eid=2-s2.0-50049123619&partnerID=tZOtx3y1.

Yap, H., Chong, N., Loh, P., Baba, R. & Yahaya, A., 1991. Survey of domiciliary cockroaches in Penang, Malaysia. *Journal of Bioscience*, 2, pp.71–75.

Zimmerman, B., 2000. Self-efficacy: An essential motive to learn. *Contemporary educational psychology*.

Bibliography

Additional references for Chapter 3 are listed below:

1. Observation Method

Kerig, P., & Baucom, D. (2004). *Couple observational coding systems*. Marwah: Lawrence Erlbaum Associates.

Shaughnessy, J. J., Zechmeister, E. B., & Zechmeister, J. S. (2009). *Research methods in psychology* (8th ed.). Boston: McGraw-Hill.

2. Questions & Questionnaire

Bradburn, N., Sudman, S., & Wansink, B. (2004). *Asking questions: the definitive guide to questionnaire design--for market research, political polls, and social and health questionnaires* (Rev.). San Francisco, CA: Jossey-Bass.

Sudman, S., & Bradburn, N. (1982). *Asking questions: a practical guide to questionnaire design.* San Francisco: Jossey-Bass.

3. Quantitative and Qualitative Research Method

Atkinson, P., Delamont, S., & Coffey, A. (2004). *Key themes in qualitative research: Continuities and changes*. Walnut Creek, Calif.: AltaMira Press.

Barbour, R. S. (2008). *Introducing Qualitative Research: A Student Guide to the Craft of Doing Qualitative Research*. *Qualitative Research Journal*. Los Angeles, CA; London: SAGE.

Berg, B., & Lune, H. (2012). *Qualitative research methods for the social sciences* (8th ed.). Boston; London: Pearson.

Boeije, H. (2009). *Analysis in qualitative research*. Los Angeles; London: SAGE.

Creswell, J. (2014). *Research design: Qualitative, quantitative, and mixed methods approaches* (4th ed.). Los Angeles, Calif.: SAGE.

Edmonds, W., & Kennedy, T. (2012). *An applied reference guide to research designs: Quantitative, qualitative, and mixed methods*. London: Sage.

Groves, R. M. (2009). *Survey methodology* (2nd ed.). Chichester: Wiley.

Kerig, P., & Baucom, D. (2004). *Couple observational coding systems*. Marwah: Lawrence Erlbaum Associates.

Appendix A. Questionnaire for the Pilot Study

Learning to Program: An Investigation

This is a short survey aiming to find out more about your perception and experience of learning to program. If you have any enquiries, or suggestions about the questionnaire, please do not hesitate to contact me via js9g09@ecs.soton.ac.uk.

Section 1. Personal Information

Please provide the following information about yourself

<u>Question 1.1</u>

Please select your gender

- ○ Female
- ○ Male

<u>Question 1.2</u>

Please enter your age

<u>Question 1.3</u>

Do you have any programming experience before taking the programme?

- ○ None
- ○ Yes, but only a little.
- ○ Yes, I am experienced.
- ○ Yes, I am very experienced.

<u>Question 1.4</u>

Please select your origin

- ○ UK
- ○ EU
- ○ International

Question 1.5

How will you grade your English language ability?

1 = Very poor; 2 = Poor; 3 = Good; 4 = Very good; 5 = Excellent; 6 = Fluent.

Very poor Fluent

1 2 3 4 5 6

Question 1.6

Did you start the introductory programming module on time?

- Yes.
- No I was late by less than two weeks.
- No I was late by three weeks or more.

Question 1.7

Please select your final grade for COMP1004 Programming Principles.

(A: ≥70; B: 60~69; C: 50~59; D: 40~49; E: <40)

- A
- B
- C
- D
- E
- I didn't take this module.

Question 1.8

Did you have any prior knowledge, experience or formal skills relevant to programming before you started the module? Please identify any items from the list below:

- [ ] ICT

- ☐ Computing
- ☐ Math
- ☐ Pascal
- ☐ VB
- ☐ Java/C#
- ☐ C/C++
- ☐ Python
- ☐ HTML/CSS
- ☐ Others -> Please specify.

Section 2. All about Our Teaching

In this section, we are trying to find out how well-structured teaching contributes to your mastery of programming at first year undergraduate level.

Question 2.1

How many introductory programming lectures did you attend?

- ○ 100%
- ○ Around 75%
- ○ Around 50%
- ○ Around 25%
- ○ 0%

Question 2.2

How did this compare to your attendance for other modules you take?

- This attendance is much more than other modules.
- This attendance is a little higher than others.
- Same with the others.
- Lower than others.

<u>Question 2.3</u>

How do you feel about the first few lectures?

- I really enjoyed them and I understood them all.
- A little challenging for me to follow but mostly good.
- A huge challenge for me and I there was a lot I didn't follow.
- I didn't really understand anything.
- I did NOT attend these lectures. (Go to #2.3b, others go to #2.4)

    Question 2.3b

    Why did you miss these lectures?

    - They are too easy for me.
    - I failed to arrive at the university on time due to visa, illness or other problems.
    - I cannot understand the lectures so I'd rather learn by myself.
    - Other -> Please specify.

<u>Question 2.4</u>

How do you rate your understanding of the basic programming concepts? Please single-click a value.

1 = Very poor; 2 = Poor; 3 = OK; 4 = Good; 5 = Very good; 6 = Excellent.

Very poor Excellent

1 2 3 4 5 6

Question 2.5

We use lectures to explain basic programming concepts to you. How did your existing level of understanding affect your approach to learning after the lectures?

◉ I followed the lectures ok and felt well prepared for to the labs and coursework.

○ I followed the lectures ok but I realised needed to do additional private study to understand the concepts.

○ I didn't really understand the lectures but thought I would cope with the labs and coursework.

○ I found understanding programming so difficult I really wanted to avoid it.

○ Other (#2.5b)

Question 2.5b

Please specify.

Question 2.6

When did you begin to feel, that learning to program was a challenging task? Please select the most three challenging theories for you.

	Variables & Assignment	If-statement	Methods & Constructors	Loops	Array & Arraylist	Testing & Debugging	Inheritence	Polymorphism	Abstract classes & Interfaces	Exceptions
No.1	○	○	○	○	○	○	○	○	○	○
No.2	○	○	○	○	○	○	○	○	○	○
No.3	○	○	○	○	○	○	○	○	○	○

Question 2.7

What did you do to cope with these challenges?

If you there were no challenging tasks for you, just state in here. Also, if there is some other challenging for you, which is NOT listed above, please specify in here.

Question 2.8

Were you successful?

- There were no challenges for me.
- Yes. (#2.8b)
- Mostly. (#2.8b)
- Hardly at all. (#2.8c)
- No. (#2.8c)

Question 2.8b

How did you feel after that?

- Very pleased and highly motivated.
- Happy about my progress.
- Nothing. Just wanna continue my study.
- Other -> Please specify.

Question 2.8c

How did you feel after that?

- Nothing. I just ignore these problems and continue my study.
- I felt disappointed and demotivated about the following study.
- I felt really disappointed. I thought I just cannot do this programming thing. I don't wanna touch anything about programming by then.
- Other -> Please specify.

<u>Question 2.9</u>

Was there any live programming demonstrating in the lectures?

- Yes, and I do like that.

- Yes but I'd rather preferred online demonstration after the lectures.
- I don't care about the live demonstrations in lectures because I'll forget it when I practice anyway. But I would like some online video kind of demonstrations after lectures.
- Not necessary for me.
- Other (#2.9b)

Question 2.9b

Please specify.

<u>Question 2.10</u>

Which book did you use as your core textbook? What do you think about it?

<u>Question 2.11</u>

How do you rate about other learning material we provide?

1 = Strongly disagree; 2 = Disagree; 3 = Agree; 4 = Strongly agree

	Strongly disagree	1	2	3	4	Strongly agree
Good quality of printed or online notes		○	○	○	○	
Sufficient computing facilities		○	○	○	○	
Sufficient library materials		○	○	○	○	
Appropriate specialist resources (e.g. laboratory facilities)		○	○	○	○	

<u>Question 2.12</u>

Does class-size matter for your programming learning?

Yes. (Go to #2.12b & c)

No. (Go to Section 3)

Question 2.12b

How do you think about the class-size you have? Please single-click in the following scale.

Too small Too big

1 2 3 4 5 6

Question 2.12c

Please specify how class-size affected your learning.

Section 3. Practical Work

<u>Question 3.1</u>

What do you think about the practical work you have done?

1 = Strongly disagree; 2 = Disagree; 3 = Agree; 4 = Strongly agree.

	Strong disagree	1	2	3	4	Strongly agree
The labs were difficult for me to finish.						
The labs were useful for my learning.						
There were too many labs.						
The coursework was difficult.						
The coursework was useful for my learning.						
There was too much coursework.						

Question 3.2

How do you think the LABS you have done affected your learning to program? Please choose a value on the following scale:

Negative Positive

1 2 3 4 5 6

Question 3.3

How do you think the COURSEWORK you have done affected your learning to program? Please choose a value on the following scale:

Negative Positive

1 2 3 4 5 6

Question 3.4

Which is your preferred way to cope with your practical work? Please select one from the list.

- ○ Individually
- ○ Work in pairs
- ○ Teamwork with two or more partners

Question 3.5

When do you start to do your coursework?

As soon as I get it Just before deadline

1 2 3 4 5 6 7 8 9 10

Section 4. Feedback

While you were learning to program, especially for COMP1004, there are many different ways we can get feedback when learning to program. Some feedback comes from using the environment; you can get verbal feedback from demonstrators and discussion with friends. Sometimes you get feedback from personal reflection after you have completed a task, and then again after you have received your marks or commented submission. As you answer the following questions think about the feedback you have received, generated or used.

Question 4.1

How did you use the feedback you received?

Question 4.2

Please rate the value of the following forms of feedback you use/experience.

1 = Never relevant/useful; 2 = Not that relevant/useful;

3 = Valuable/useful; 4 = Very valuable/useful.

	Never relevant/useful	1	2	3	4	Very valuable/useful
From the programming environment		○	○	○	○	
Personal reflection		○	○	○	○	
Discussion with friends		○	○	○	○	
Discussion with academics/lecturers		○	○	○	○	
Discussion with demonstrators		○	○	○	○	
Marks from my practical work		○	○	○	○	
Comments on my practical work		○	○	○	○	
Comments on other students' practical work		○	○	○	○	
Marks on other students' practical work		○	○	○	○	

Question 4.3

How do you rate the feedback you received?

Very bad Very good

1 2 3 4 5 6

<u>Question 4.4</u>

The feedback you have is:

1 = Strongly disagree; 2 = Disagree; 3 = Agree; 4 = Strongly agree.

	Strongly disagree 1	2	3	4	Strongly agree
Fair					
Timely					
Useful					

<u>Question 4.5</u>

What kind of feedback do you think is most helpful to help you learn?

<u>Question 4.6</u>

What kind of feedback do you find least useful?

<u>Question 4.7</u>

Is there any kind of feedback you want to receive but never got? Please specify.

<u>Question 4.8</u>

Normally, what do you do immediately after you get the feedback? And how do you react to it?

Section 5. Others

<u>Question 5.1</u>

Did you need extra help while learning to program?

- Yes.
- No.

Question 5.1b

What do you usually do when you need extra help when learning to program?

e.g. You just finished writing a piece of code but it always failed to compile.

Please tick at least one option from the following:

- ☐ Search from the Internet
- ☐ Go to textbooks or lecture slides.
- ☐ Ask friends.
- ☐ Ask lecturers or demonstrators. -> *Which do you NOT prefer to ask, lecturers, demonstrators, or maybe both of them? Please specify the reason.*
- ☐ Do nothing. -> *Please specify the reason*.
- ☐ Not listed above. -> *Please speficy.*

<u>Question 5.2</u>

Do you have any suggestions about additional ways in which you might be provided with help?

<u>Question 5.3</u>

Usually, how much time do you spend learning to program in your spare time per week, excluding the coursework & labs?

- ○ Not at all.
- ○ < 30 min per week.
- ○ 30 - 60 min per week.
- ○ 1 or 2 hours per week.
- ○ 3 or 4 hours per week.
- ○ More than above. (#5.3b)
- ○ Hard to answer.(#5.3c)

Question 5.3b

Please specify.

Question 5.3c

Please describe the reason.

Question 5.4

Imagine you have had just had a lecture, such as theory about loops, when will you practice this new skill:

- Wait until labs & coursework.
- Start on my own right away.

Question 5.5

Have you ever thought about leaving the course?

- Yes. (Go to #5.5b)
- No. (Go to #5.6)

Question 5.5b

Please specify the reason and your current attitude about programming.

Question 5.6

Now you have learned some other programming languages (C/C++), do you consider Java is a good gateway language in terms of learning to program?

- Yes Java is helpful.
- Mostly helpful. I know better about programming.
- Just a little bit. Programming is challenging for me.
- Totally unhelpful. Programming is still hard.

Question 5.7

Are you considering programming as your future career?

- Yes, definitely.
- Yes, probably.

- No, but I would be happy for my job to include some programming.
- No, not at all.

Thank you for taking this questionnaire.

Appendix B. Questionnaire for the Initial Survey

Understanding Your Thinking Towards Learning to Program

This short questionnaire aims to understand your background and current attitudes towards learning to program.

First of all, please write down your **age**: ___________.

Then, please provide the following information about yourself by simply tick in the right answer for you:

Gender: ☐ Female ☐ Male

Origin: ☐ UK ☐ EU ☐ International

Your Course: ☐ Computer Science ☐ Software Engineering ☐ Electrical Engineering

☐ Electromechanical Engineering ☐ Electronic Engineering

☐ Information Technology in Organisations

Do you have any programming experience? ☐ None ☐ A little ☐ A lot ☐ Extensive

Your current attitude towards learning to program:

1 = Strongly disagree; 2 = Disagree; 3 = Agree; 4 = Strongly agree

	1	2	3	4
I feel motivated to learn to program.	☐	☐	☐	☐
I think learning to program will be easy for me.	☐	☐	☐	☐
I think I will have to work hard to learn to program.	☐	☐	☐	☐
I think learning to program will be difficult for me.	☐	☐	☐	☐
I'm sure I will have mastered programming by the exam.	☐	☐	☐	☐

Appendix C. Questionnaire for the Weekly Survey

Understanding your Learning Experience

The following optional questions aim to understand your programming learning progress and expectation. Your responses will help us to enhance our teaching strategies.

Lab # ______

1. What is your feeling about today's lab? (Tick any that apply)

☐Hard to begin with

☐ Easy

☐Mixed easy & difficult

☐Difficult

☐Easier than I expected

☐Harder than I expected

☐Interesting

☐Challenging

☐Other: ___________________________

2. Until now, how do you feel about Java programming?

☐ Love it! :-D

☐Like it! :-)

☐Ho Hum... :-(

☐Don't like it. :-((

3. Based on your progress so far, what grade do you think you can get in the final examination?

☐First Class (⩾70%)

☐2.1 (60%-69%)

☐2.2 (50%-59%)

☐ Pass (40%-49%)

☐Fail (<40%)

Appendix D. Questionnaire for End of Term Survey

Is the Internet Helpful?

1. Do you have any programming experience before you started this degree?

☐ None ☐ Some ☐ A lot

2. When do you use the Internet when learning to program?

☐ None

☐ Before lectures/labs

☐ During lectures/labs

☐ After lectures/labs

3. What online platform(s) do you use?

☐ None

☐ Formally provided ones, e.g. ECS intranet, Edshare, student wiki

☐ Public searching engine, e.g. Google, Bing.

☐ Online live programming videos, e.g. watching tutorial videos in YouTube

☐ Social networking, e.g. Facebook.

☐ Forum/BBS

☐ Online courses

☐ Others, please specify: _________________________

4. How do you use the Internet?

☐ None

☐ Learning a new theory involved in the lectures, e.g. inheritance, debug, array.

☐ Learning extra theories, e.g. Enum.

☐ Review

☐ Getting some help when struggling for formal provided labs/coursework

☐ Others, please specify:_________________________

5. Why do you choose the Internet to help you learn to program?

☐ I never use the Internet. Please specify the reason.

☐ I don't have other choices.

☐ Convenient.

☐ I don't have to rely on other resources, e.g. textbook.

- □ Try to avoid asking "silly" questions to other people, especially the lecturers, because it might make them consider me to be "stupid" and not hard-working.
- □ Others, please specify: ____________________________.

6. Do you have any suggestions on how this module could be enhanced by adopting the Internet (probably in terms of improving the teaching strategies, providing more online resources or platforms to learn and giving feedback, etc)?

____

Appendix E. Big Table of Observation Data

	Lab 1	Lab 2	Lab 3	Lab 4	Lab 5	Lab 6	Lab 7	Lab 8	Lab 9	Lab 10
Time	**2012/10/5**	**2012/10/12**	**2012/10/19**	**2012/10/26**	**2012/11/2**	**2012/11/9**	**2012/11/16**	**2012/11/23**	**2012/11/30**	**2012/12/7**
Content	Intro, compiling, variables; Object comparison & autoboxing	Define class; Conditionals & control statement	Methods, object interaction & testing	Loops, ArrayList, Manipulating ArrayList	Array, HashMaps & APIs	Scope; Test harness; Scope debug with IDEs	Inheritance; Method overwriting; Polymorphism & Exceptions	Method overloading; Interfaces; Understanding class hierarchy	Reading from a file; Modelling loaded data; writing files	Revision
Attendence	≈100%	≈100%	≈100%	≈100%	≈100%	≈100%	≈100%	≈95%	≈95%	≈95%
Main Confusion	Many students (over 40 attendees) have some small problems	Many students (over 40 attendees) have some small problems	A few of the less confident students plus some less attentive students (about 30 attendees) had problems		A few of the less confident students plus some less attentive students (about 30 attendees) had problems	Many students (about 40 attendees) have some problems		Many less confident students and some less attentive students (over 40 attendees) had problems	A few of the less confident students plus some less attentive students (about 25 attendees) had	
	1. Over 80% questions were language specific problems: e.g. naming principles	1. About half of questions asked were detailed problems: e.g. naming principles, case sensitive, etc.	Over 80% questions related to two areas of confusion:		1. About half the questions asked were based on lab 4.	Over 80% questions asked were about:		About 1/3 of questions asked were about in-depth object-oriented theories.	About 70% questions were about how to read and write external files.	
	2. For some novices; structure of the code	2. About 1/3: logic (and, or, not)	1. Regarding: how to create/call methods		2. Some students confused about the spec, which was less clear than previous	1. How to write test script				
			2. Regarding concepts e.g. Parameters, constructors			2. How to debug in Eclipse.				
Reasons for confusion	Not familiar with how Java works, mainly some fundamental issues, e.g.unfamiliar with the write, compile, run process.	1. Not familiar with how Java works, mainly some fundamental issues, e.g.unfamiliar with the write, compile, run process and the main function.	Not familiar with how Java works, mainly some fundamental issues, e.g.unfamiliar with the write, compile, run process and the need to instantiate		1. Specification was not as clear as previous examples, i.e. lab task intentionally more challenging.	1. Specification was not as clear as previous examples, i.e. lab task intentionally more challenging.		Too clear spec in Lab 7 → surface learning for basic OOP theories.	1. New topics. Many students were still confused about how to use API (lab 5).	
		2. Weakness in logic (and, or, not)			2. PGTA had been busy with marking lab from last week, no one had	2. Lack of testing experience and computing thinking ability.			2. Lack of problem solving ability.	
Learning behaviours for novices	About 30% of attendees:	About 30% of attendees:	About 30% of attendees:		About 30% attendees (most of whom seemed to be novices) started to understand deeply about how Java works rather than simply following the spec only. Some of them stated that the feedback they received last week	Over half of students:		1. About 40% attendees appeared to start to understand deeply about how OOP theories work rather than simply following the spec only. (deep learning)	1. About 50% attendees appeared to understand deeply about how OOP theories work rather than simply following the spec only. (deep learning)	
	1. Reading set book/notes/slides for reference;	1. Reading set book/notes/slides for reference;	1. Reading set book/notes/slides for reference;			1. Review notes/slides/books		2. Some of them stated that the feedback they received last week suggested they follow learning approaches suggested by written and verbal feedback from	2. Some of them stated that the feedback they received last week suggested they follow learning approaches suggested by written and verbal feedback from	
	2. “Copying from examples” e.g. follow the spec closely	2. “Copying from examples” e.g. follow the spec closely	2. “Copying from examples” e.g. follow the spec closely			2. Less Web than before - debugging is a new topic, they appeared to be unsure about what to search for.				
Learning behaviours for pre-experienced students	About 10 students left early. I talked to them, they thought this lab was easy and finished in a short time.	About 20% of the attendees:	About 15% of attendees:		About 15% of attendees:	About 10% attendees:		About 15% of attendees:	About 20% of attendees:	
		1. Finish the task in a short time	1. Finish the tasked in a short time		1. Finish the tasked in a short time	1. Learn online		1. Finish the tasked in a short time	1. Finish the tasked in a short time	
		2. Answering questions to strugglers	2. Experimenting some extra functions beyond spec		2. Experimenting some extra functions beyond spec	2. Discuss with experts (i.e. PGTA, pre-experienced students)		2. Experimenting some extra functions beyond spec	2. Experimenting some extra functions beyond spec	
					3. Answering questions for strugglers			3. Answering questions to strugglers	3. Answering questions to strugglers	
How novices sought help	1. Prefer: mixture of printed/online slides, printed/personal notes, textbooks	1. Prefer: mixture of printed/online slides, printed/personal notes, textbooks	1. Prefer: mixture of printed/online slides, printed/personal notes, textbooks		1. Novices started to come to lab without textbooks - relying upon online searches and resources	1. Ask experts, i.e. PGTA, pre-experienced students.		1. Many novices now coming to lab without textbooks	1. Ask experts, i.e. PGTA, pre-experienced students.	
	2. Online tutorials that they found themselves	2. Online tutorials that they found themselves	2. Online tutorials that they found themselves		2. Reviewing slides and notes from the lectures	2. Little Google. Some explained they don't know		2. Reviewing slides and notes	2. More on Google: they know what to search now.	
	3. Ask experts (e.g. PGTA, pre-experienced students)	3. Ask experts (i.e. PGTA, pre-experienced students)	3. Ask experts (e.g. PGTA, pre-experienced students)		3. More online searching than before			3. More online searching than before		
	4. relatively little use of Google (some explained they don't know what to search for)	4. relatively little use of Google (some explained they don't know what to search for)	4. relatively little use of Google (some explained they don't know what to search for)							
How pre-experienced students sought help (about 20 students were asked each week)	1. Easy lab, very few questions.	1. Easy lab, no questions.	1. Easy lab, no questions.		1. Only a few pre-experienced students observed have questions, they mainly used Google.	1. Only a few pre-experienced students observed have questions, they mainly used Google.		1. Only a few pre-experienced students observed have questions, they mainly used Google.	1. Only a few pre-experienced students observed have questions, they mainly used Google.	
	2. Only a few pre-experienced students observed have questions, who used Google mainly.	2. Only a few pre-experienced students observed have questions, who used Google mainly.	2. Only a few pre-experienced students observed have questions, who used Google mainly.		2. Discuss with experts (i.e. PGTA, pre-experienced students)	2. Discuss with experts (i.e. PGTA, pre-experienced students)		2. Discuss with experts (i.e. PGTA, pre-experienced students), mainly for deeper understanding and exploring extra theories.	2. Discuss with experts (i.e. PGTA, pre-experienced students), mainly for deeper understanding and exploring extra theories.	
Comments & Analysis	1. “Copying from examples” e.g. follow the spec closely	1. “Copying from examples” e.g. follow the spec closely	1. BlueJ is introduced, but few questions were asked about this IDE.		1. Eclipse was introduced but few questions were asked.	1. Deadlines from other modules delayed some students' learning progress.		1. Busy lab due to the marking lab last week	1. Due to the marking lab in week 7, some students were still confused with lab 7 & 8	
	2. Surface learning (for novices). Little evidence of deep learning approaches.	2. Surface learning (for novices). No/Less deep learning.	2. The spec gives clear guidance		2. Feedback affected learning approaches, especially for novices/little skilled students.	2. Many students said this lab was hard to begin.		2. Feedback appeared to influence learning approaches, especially for novices/little skilled	2. Feedback appeared to influence learning approaches, especially for novices/little skilled	
	3. No one gave up. Syntax does not appear a problem	3. No one gave up. Syntax is not a problem	3. Strugglers wouldn't ask lecturers unless the lecturer talked to them first (one explained she was shy)			3. Debugging is different from coding. Some students reflected that they had no problem with Java but confused about debugging in Eclipse.		3. A few strugglers gave up. Some of them were still confused with tasks fromprevious labs, so this lab was too difficult for them.	3. A few strugglers gave up. Some of them were still confused with previous labs, so this lab was too difficult for them.	
	4. Not so much Web because they have no idea about what to search for	4. Not so much Web because they have no idea about what to search								
	5. Highly motivated when problem solved	5. Highly motivated when problem solved								

Lab 4 was a busy marking lab. I was busy marking lab 1-3 for students and had no time to make detailed observation. When students need help, they might have to ask their friends or search online.
Lab 7 was a busy marking lab. I was busy marking lab 4-6 for students and had no time to make detailed observation. When students need help, they might have to ask their friends or search online. But many students reflected that the lab spec was really easy to follow.
Lab 10 is a marking and revision lab. All the PGTAs were busy marking lab 7 – 9. Over a half students left as soon as they got marked. The students remained just asked questions for their coursework mainly. Talking to some strugglers, they gave up lab 8 & 9 because they believed that if they can do lab 1 – 7, they could at least