NATIONAL INSTITUTE OF OCEANOGRAPHY

WORMLEY, GODALMING, SURREY

Special Purpose Utility Programs for use with the I. B. M. 1800 Computer

N.I.O. INTERNAL REPORT No. N. 25

JULY 1971

NATIONAL INSTITUTE OF OCEANOGRAPHY

WORMLEY, GODALMING, SURREY

SPECIAL PURPOSE UTILITY PROGRAMS
FOR USE WITH THE
I.B.M. 1800 COMPUTER

N.I.O. Internal Report No. N25

JULY 1971

N.I.O. PROGRAMS N.25

N.I.O. Program Directory Suite

127	Dump N.I.O. Program Directory	NIODP
128	Alter Entries in Program Directory	FILCH
129	Add Records to Program Directory	FIAT
143	Add Internal Report Numbers to Program Directory	AIRNO
149	Initialise Alphabetically sorted Program Directory File	IDAT5
217	Initialise Program Directory File 1	IDAT1
218	Initialise Program Directory File 2	IDAT2
219	Initialise Program Directory File 3	IDAT3
220	Initialise Program Directory File 4	IDAT4
	D.P.G./P/23	

Dumping Disks to Magnetic Tape

-83	Dump Disk to Magtape	DIKTR
-84	Dump Magtape to Disk	TDIKW
204	Book-keeping for DIKTR	DIMAG
205	Book-keeping for TDIKW	MAGDI
224	Dump DIREK Directory	DDUMP
225	Save DIREK Directory	DSAVE

Tape Security System

-92	T.S.S. Find a File and Record (Part of TASS)	FINDM
-93	T.S.S. Find Next available File and Record (Part of TASS)	NEXTM
-94	T.S.S. Store Current Tape Position (Part of TASS)	STORM
211	T.S.S. File Initialisation (Part of TASS)	INITM
212	T.S.S. Query File (Part of TASS)	QUERM
	D.P.G./P/22 (or see D.P.G./P/7)	

Various Special Purpose Utilities

-95	Determine First and Last Records in a Dynamic Data File	IFREG
-96	Find the Record Number of a Dynamic Data File given Day and Time	JFND
-99	Calculate Algebraic Differences between Two Times	DFMIN
-118	BCD Tape Handling Utility	SLSET
133	Tape Handling and Utility Management Program	THUMP
226	Change Drive 1 Disk to Drive 2	D1TO2
227	Change Drive 2 Disk to Drive 1	D2TO1
231	Relabel a Drive 1 or Drive 2 Disk	LABEL

Queries regarding the use or availability of any of the programs in this volume may be made to:-

The Program Librarian,
Data Processing Group,
National Institute of Oceanography,
Wormley, Godalming, Surrey.

from whom a comprehensive list of all current N.I.O.Programs is available.

All the programs in this volume have been compiled and executed on an I.B.M. 1800 Computer having the following configuration:-

1802 Processor-Controller with 24,576 words of core storage

3 1810 Disk Drives Model A

2 2401 Magnetic Tape Drives (30 Kc/s) (7 Track)

1442 Model 6 Card Read - Punch

1443 Printer, 240 lines/minute

1816 Keyboard-Printer

Facit Paper Tape Reader, 1000 Characters/second

Facit Paper Tape Punch, 150 Characters/second

The Operating System was T.S.X. Version 3

N.I.O. PROGRAM 127	CLASSIFICATION S
Title	Dump N.I.O. Program Directory
Name	NIODP
Language	1800 Fortran IV
Operating System	T.S.X. or M.P.X.
Machine	I.B.M. 1800
Purpose	This program is part of the N.I.O. program directory suite and is used to dump the N.I.O. Program Directory.
Input	Date entered via keyboard.
Output	The program directory. The programs are first listed in numerical order, then in order of classification letter, then in order of Internal Report Number and lastly alphabetically.
Use	The program is stored in the fixed area on a drive 1 disk.

To use:-

```
                      19
//bJOB                 X

//bXEQbNIODPbbbFX
```

Execution Time	Approximately 15 minutes per complete listing, 3 minutes per part listing (see under Options)
Options	If data switch zero is on the program will loop back to the beginning after producing a listing. If data switch 1 is on only the numerical and alphabetic listings will be produced.
Notes	For full details of the complete suite of programs and data files specifications see DPG/P/23.
Programmer	William Strudwick.

N.I.O. PROGRAM 128 CLASSIFICATION S

Title	Alter entries in Directory Files
Name	FILCH
Language	1800 Fortran IV
Machine	I.B.M. 1800
Purpose	This program is part of the N.I.O. program directory suite and is used to alter records in the data files DATA3 and DATA4 which contain information on programs and subprograms in the Program Library.

Input

Data cards are as follows:-

1) Pairs of the following for each program or subprogram.

First Card:-

```
cc
 1-4 Program Number, Integer, right justified.
 6-10 Name of program left justified.
13-16 Programmers Initials (e.g. WKS), left justified.
18    Classification Letter.
20-23 Internal Report Number, left justified (e.g. N18).
```

Second Card:-

```
cc
 1-80 Title of the program or subprogram.
```

ii) More of the above pairs if.required.

iii) 2 Blank cards.

Output

Updated disk files and message on lineprinter: 'JOB COMPLETE'.

Use

The program is stored in the fixed area on a drive 1 disk
To use:-

```
                          19
//bJOB                     X

                    16
//bXEQbFILCH        FX
```

Data cards (see under input)

Notes

This does not update the alphabetically sorted file (DATA5).
For full details of the complete suite of programs and data files specifications see DPG/P/23.

Programmer

William Strudwick.

N.I.O. PROGRAM 129 CLASSIFICATION S

Title Add records to Program Directory

Name FIAT

Language 1800 Fortran IV

Operating System T.S.X. or M.P.X.

Machine I.B.M. 1800

Purpose This program is part of the N.I.O. Program Directory Suite and it is used to add records to the data files DATA3 and DATA4 which contain information on programs and in the Program Library.

Input Data Cards are as follows:-

1) Pairs of the following for each program or subprogram.
 First Card:-
```
cc
 1-4  Program Number, Integer, right justified
 6-10 Name of program, left justified
13-16 Programmers initials (e.g. WKS), left justified
   18 Classification letter
20-23 Internal Report Number, left justified
```
 Second Card:-
```
cc
 1-80  Title of the program or subprogram
```
2) More of the above if required
3) 2 blank cards

Output Updated disk files and message on lineprinter 'JOB COMPLETE'

Use The program is stored in the fixed area on a drive 1 disk.
To use:-

```
                          19
//bJOB                     X
                       16
//bXEQbFIAT             FX
```

Data cards (see under INPUT)

Notes This does not update the alphabetically sorted file (DATA5)
For full details of the complete suite of programs and data files specifications see DPG/P/23.

Programmer Willian Strudwick.

N.I.O. PROGRAM 143 CLASSIFICATION S

Title: Add Internal Report Numbers to Program Directory

Name: AIRNO

Language: 1800 Fortran IV

Machine: IBM 1800

Operating System: T.S.X. and M.P.X.

Purpose: This program is part of the N.I.O. program directory suite and is used to add Internal Report Numbers to programs already existing in the program directory files on disk.

Input: Data cards as follows:-

1) Internal Report Number. e.g. N18 in columns 1-4, left-justified.
2) cc 1-4 Number of program or subprogram to have internal report number added, integer, right justified.
3) More of type (2) if required.
4) Blank Card.

Output: Updated files on disk and message on lineprinter 'JOB COMPLETE'

Use: The program is stored in the fixed area on a drive 1 disk.

To use:-

```
                    19
//bJOB              X

//bXEQbAIRNObbbFX
```

Data cards (see under INPUT)

Notes: This does update the alphabetically sorted file. For full details of the complete suite of programs and data files specification see DPG/P/23

Programmer: William Strudwick.

N.I.O. PROGRAM 149 CLASSIFICATION S

Title	Initialise Program Directory File 5
Name	IDAT5
Language	1800 Fortran IV
Machine	IBM 1800
Operating System	T.S.X. and M.P.X.
Purpose	This program is part of the N.I.O. Program Directory suite and is used to initialise the data file DATA5 which contains information on programs in alphabetical order.

Input

Data cards as follows:-

1) Pairs of cards:-

First card:-

```
cc  1 - 4   Program number, integer, right justified
    6 -10   Name of program, left justified.
   13 -16   Programmer's Initials, left justified.
       18   Classification Letter.
   20 -23   Internal report Number, left justified (e.g. N.18).
       25   Is one for the last program otherwise zero.
```

Second card:-

```
cc  1 -80   Title of program or subprogram
```

2) More of the above if required.

Output

Initialised file on disk and message on lineprinter 'JOB COMPLETE'

Use

The program is store d in the fixed area on a drive 1 disk.

To use:-

```
                    19
//bJOB               X
                  16
//bXEQbIDAT5     FX
```

Note

For full details of the complete suite of programs and data files specifications see DPG/P/23

Programmer

William Strudwick.

N.I.O. PROGRAM 217 CLASSIFICATION S

Title	Initialise Program Directory File 1
Name	IDAT1
Language	1800 Fortran IV
Machine	IBM 1800
Operating System	T.S.X. and M.P.X.
Purpose	This program is part of the N.I.O. Program Directory Suite and is used to initialise the data file DATA1 which contains abbreviations for use with the suite.

Input

Data Cards are as follows:-

1) Pairs of cards:-

cc 1-60 Abbreviations to appear in the first 60 print positions on the lineprinter.

Second Card:

cc 1-60 Abbreviations to appear in the last 60 print positions on the lineprinter.

2) More of type one if required.

3) Two cards punched with slashes in column one of each card.

Output

Initialised file on disk and message on lineprinter 'JOB COMPLETE'

Use

The program is stored on a drive 1 disk

To use:-

```
                19
//bJOB          X

//bXEQbIDAT1

*FILES(1,DATA1,1)

*CCEND

Data cards (see under INPUT)
```

Notes

For full details of the complete suite of programs and data files specifications see DPG/P/23.

Programmer

William Strudwick.

N.I.O. PROGRAM 218 CLASSIFICATION S

Title Initialise Program Directory File 2

Name IDAT2

Language 1800 Fortran IV

Machine IBM 1800

Operating System T.S.X. and M.P.X.

Purpose This program is part of the N.I.O. Program Directory Suite and is used to initialise the data file DATA2 which contains the numbers of the internal reports currently in circulation.

Input Data cards as follows:-

1) cc 1-4, 5-8, 9-12 77-80

The numbers of the internal reports e.g. N.18, left justified. The last Internal Report Number should be followed by two asterisks in the next number field. e.g. If the last Internal Report is in cc 9-12, atars should appear in cc 13 and 14.

2) More of the above if two asterisks are not encountered by the end of the previous card.

Output Initialised file on disk and message on lineprinter 'JOB COMPLETE'

Use The program is stored on a drive 1 disk.

```
Use:-      //bJOB          19
                            X

           //bXEQbIDAT2

           *FILES(2,DATA2,1)

           *CCEND

           Data cards (see under INPUT)
```

Notes For full details of the complete suite of programs and data files specifications see DPG/P/23

Programmer W. Strudwick.

N.I.O. PROGRAM 219 CLASSIFICATION S

Title	Initialise Program Directory File 3
Name	IDAT3
Language	1800 Fortran IV
Machine	IBM 1800
Operating System	T.S.X. and M.P.X.
Purpose	This program is part of the N.I.O. Program Directory suite and is used to initialise the data file DATA3 which contains the information on the subprograms in the Program Library.

Input

Data cards are as follows:-

1) For each subprogram two cards.

First Card:-

cc 1-5	Name of subprograms, left justified.
8-11	Programmers Initials, left justified.
13	Classification Letter
15-18	Internal Report on which the write up of the subprogram appears, left justified.
20	Is one for the last set of cards otherwise zero.

Second Card:-

cc 1-80	Title of Subprogram.

2) More of the above pairs if required.

Output

Initialised file on disk and message on lineprinter 'JOB COMPLETE'

Use

The program is stored on a drive 1 disk.

To use:-

```
                    19
//bJOB               X

//bXEQbIDAT3

*FILES (3,DATA3,1)

*CCEND
```

Data cards (see under INPUT)

Notes

For full details of the complete suite of programs and data files specifications see DPG/P/23. The cards for each subprogram should be in numerical order. The maximum number of subprograms allowed is 499.

Programmer

William Strudwick.

N.I.O. PROGRAM 220 CLASSIFICATION S

Title	Initialise Program Directory File 4
Name	IDAT4
Language	1800 Fortran IV
Machine	IBM 1800
Operating System	T.S.X. and M.P.X.
Purpose	This program is part of the N.I.O. Program Directory suite and is used to initialise the data file DATA4 which contains the information on the programs in the program library.

Input

Data cards as follows:-

1) cc 1-3 the number of the first program plus 1

2) For each program a pair of cards punched as follows:-

First card:-

cc 1 - 6	Name of program, left justified.
8 -11	Programmers initials, left justified.
13	Classification letter.
15 -18	Internal report in which the write-up of the program appears left-justified.
20	Is one for the last set of cards otherwise zero.

Second Card:-

cc 1 - 80 Title of program

3) More of the above pairs if required.

Output

Initialised file on disk and message on lineprinter 'JOB COMPLETE'

Use

The program is stored on a drive 1 disk.

To use:-

```
                  19
//bJOB             X

//bXEQbIDAT4

*FILES(4,DATA4,1)

*CCEND
```

Data cards (see under INPUT)

Notes

For full details of the complete suite of programs and data files specifications see DPG/P/23. The cards for each program should be in numerical order.

The maximum number of programs allowed is 499.

Programmer

William Strudwick.

N.I.O. PROGRAM DIRECTORY

This is a suite of programs which maintains information on the programs and subprograms in the Program Library and produces an up-to-date listing of this information.

The Program Directory is a printout of the information. Each page is numbered and has the date on it. The first page contains abbreviations used in the directory. The programs are then listed in numerical order (subprograms first), then in order of classification letter (subprograms first), then in order of the Internal Report in which the write-up appeared (subprograms first), and finally in alphabetical order (programs and subprograms mixed). There is a blank page between each section.

The program NIODP, which produces the Program Directory, has options in it which allow a shorter Directory to be produced. This Directory has the programs listed in numerical order and alphabetical order only. This is much faster to produce. For further details see the write-up of NIODP (N.I.O. program 127).

INFORMATION STORED

The information held about each program and subprogram is as follows.

(i) The number of the program. This is calculated from the position on disk where the information is stored, except for the alphabetically sorted file. Subprograms have negative numbers, programs have positive numbers.

(ii) The name of the program.

(iii) The programmer's initials.

(iv) The title of the program. This is a short sentence which describes, as far as possible, what the program or subprogram does.

(v) The classification letter. This is a single letter which decides into which group the program is to be placed.

The classifications are:-

B	Biology, Chemistry
G	Geophysics and Geology
M	Mathematics and Statistics
N	Navigation

P Physical Oceanography, Waves, Tides, and Currents

S Special Purpose Utilities

U General Purpose Utilities

(vi) The Internal Report number in which the write-up of the program or subprogram appears. This will normally be blank for a new program or subprogram.

DATA FILES

The information is held on a drive 1 disk, at present cartridge label 56, in 5 data files. These data files are described below. For compatibility throughout the suite of programs the same symbolic file number and associated variable are used for the same logical file.

DATA1 is symbolic file number 1 and has associated variable IREC. It is defined as having 100 records each of 30 words. The first word of the first record is a count of the total number of records used for data, the remaining words are not used.

This file contains the abbreviations which are printed out at the beginning of each Program Directory. They are arranged on disk so that each even record will contain the characters to be printed out in the first 60 print positions on the lineprinter and each odd record the characters to be printed out in the last 60 print positions. For best presentation, the abbreviations have been standardised to the following.

The first 60 print positions down the page will contain the programmer's initials and the programmer's name in the form:-

```
XXXb-bYYYYYYYYYYYYYYYYYYYY
```

where the Ys are the name and the XXX is the abbreviation for that name.

The last 60 print positions will contain any other abbreviation in the form

```
XXXXb-bYYYYYYYYYYYYYYYYYYYY
```

where the Ys are the explanation of the abbreviation XXXX.

This file is initialised by the program IDAT1 (N.I.O. Program 217), and no other program can alter it. All information is left justified and is A2 format. The maximum number of abbreviations is 99.

DATA2 is symbolic file number 2 and has associated variable JREC. It is defined as having 100 records each of 2 words. The first word of the first record is a count of the total number of records used for data, the other word is not used.

This file contains the Internal Report numbers which are currently in circulation. They are arranged on disk in numerical order. The Internal Report numbers include the prefix N, e.g. N18. The information is held in A2 format and is left justified. This file is initialised by the program IDAT2 (N.I.O. Program 218) program AIRNO (N.I.O. Program 143) can add records to it. It is used by program NIODP (N.I.O. Program 127).

DATA3 is symbolic file number 3 and has associated variable KREC. It is defined as having 500 records each of 48 words. The first word of the first record is a count of the number of records used for data; the remaining 47 words are not used.

This file contains the information on the subprograms. They are in numerical order. The layout is as follows:-

words:	1-3	Name of program (A2)
	4-5	Programmer's Initials (A2)
	6-45	Title (A2)
	46	Classification Letter (A1)
	47-48	Internal Report Number (A2)

The file is initialised by the program IDAT3 (N.I.O. Program 219). Programs AIRNO, FILCH and FIAT update it.

DATA4 is symbolic file number 4 and has associated variable LREC. It is defined as having 500 records each of 48 words. The layout is the same as for DATA3 with the following exception.

The first word of the first record is the record number at which data starts and the second word is the record number at which data ends, the remaining 46 words are not used. The file is initialised by the program IDAT4 (N.I.O. Program 220). Programs AIRNO, FILCH and FIAT update it.

DATA5 is symbolic file number 5 and has associated variable MREC. It is defined as having 1000 records each of 49 words. The first word of the first record contains a count of the number of records used for data, the remaining 48 words are not used.

This file contains information on programs and subprograms in the Program Library in alphabetical order. The layout of the file is as follows:-

words:	1	Program Number (Integer)
	2-4	Name of Program (A2)
	5-6	Programmer's Initials (A2)
	7-46	Title (A2)
	47	Classification Letter (A1)
	48-49	Internal Report Number (A2)

The file is initialised by the program IDAT5 (N.I.O. Program 149). Program AIRNO updates it.

PROGRAMS

The following are the programs in the suite.

Name	Purpose	Number
IDAT1	Initialise File 1	217
IDAT2	Initialise File 2	218
IDAT3	Initialise File 3	219
IDAT4	Initialise File 4	220
IDAT5	Initialise File 5	149
FIAT	Add records to directory files	129
AIRNO	Add internal report numbers to directory files	143
FILCH	Change records in directory files	128
NIODP	Print Program Directory	127

SETTING UP THE SYSTEM

1. Compile programs IDAT1, IDAT2, IDAT3, IDAT4 and store in Working Storage.

2. Set up five data areas with a deck of cards as follows:-

```
                   19
//bJOB              X

//bDUP
                   13      17      21        29
*STOREDATAD         WS1     FX1     DATA1     ØØ1Ø

*STOREDATAD         WS1     FX1     DATA2     ØØØ1

*STOREDATAD         WS1     FX1     DATA3     ØØ75

*STOREDATAD         WS1     FX1     DATA4     ØØ75

*STOREDATAD         WS1     FX1     DATA5     Ø154
```

(Under MPX the *DFILE function should be used.)

Compile program IDAT5 and store in the fixed area using the following store cards:-

```
                   21      27
*STORECI            IDAT5   IDAT5

*FILES (5, DATA5, 1)

*CCEND
```

3. Initialise data files using programs

IDAT1, IDAT2, IDAT3, IDAT4, IDAT5

(See individual write-ups for format of cards.)

4. Compile and store in the fixed area the following programs (the *FILES card needed in each case is shown).

```
AIRNO     *FILES (3, DATA3, 1), (4, DATA4, 1), (5, DATA5, 1)

FIAT      *FILES (3, DATA3, 1), (4, DATA4, 1)

FILCH     *FILES (3, DATA3, 1), (4, DATA4, 1)

NIODP     *FILES (1, DATA1), (2, DATA2, 1),

                 (3, DATA3, 1), (4, DATA4, 1)

                 (5, DATA5, 1)
```

5. To check that everything is O.K., execute program NIODP.

NOTES

The drive 1 disk should have at least 5Ø cylinders of fixed area and 25 cylinders of working storage.

ALTERING FILES

DATA1	use	IDAT1
DATA2	use	IDAT2
DATA3	use	IDAT3, FIAT, FILCH or AIRNO
DATA4	use	IDAT4, FIAT, FILCH or AIRNO
DATA5	use	IDAT5 or AIRNO

Notes

(i) When using AIRNO, update deck of cards containing the data for DATA5, as IDAT5 will destroy what was previously on disk in this file when it is next run.

(ii) To add or change a record in DATA5 make the alteration on the card deck of the file and reload using IDAT5.

(iii) When adding a new program to the Library run program FIAT with appropriate data cards. Then insert the data cards used with FIAT in their right place in the card deck of DATA5 and reload using IDAT5.

(iv) When changing a record run program FILCH with appropriate data cards. Take out the cards referring to the program being changed and insert the cards used with FILCH in their right place in the card deck of DATA5 and reload using IDAT5.

William K. Strudwick

5.12.70

Title	Disk to magnetic tape, and magnetic tape to disk dumping subroutines.
Names	DIKTR and TDIKW respectively.
Machine	I.B.M. 1800.
Operating System	T.S.X.
Language	1800 Assembler.

Purpose

To dump the contents of an entire disk to a file on magnetic tape; and at a later date if required, to write that file back to disk.
These subroutines are used in conjunction with programs 204 and 205, which 'call' the subroutines and keep a directory of all the dumps which take place.
However, they may be used in other programs, when book-keeping is not required, so long as the calling sequence is correct.

To Use

The calling sequence is as follows:-
In Fortran:-

```
CALL DIKTR(IDISK,IDRIV,IDECK)
  or TDIKW
```

In Assembler:-

```
cc27                     cc35
  CALL                      DIKTR
                         or TDIKW
  DC                        IDISK
  DC                        IDRIV
  DC                        IDECK
```

where IDISK is the number of the disk from/to which the dump is to take place, and
IDRIV is the drive number of that disk.
IDECK is the magnetic tape deck to be used.

Method

Both subroutines check the first word of sector Ø of the disk on the drive IDRIV, to ensure that the correct disk has been loaded, then consecutive sectors from disk (or records from magnetic tape in the case of TDIKW), are read and written alternately. This takes place a total of 1600 times (the number of sectors on a disk).
The subroutine DIKTR checks to see whether each sector is 'file-protected' and if so, records that status also on magnetic tape (in the first word of the 321-word long record).
TDIKW does the reverse so that bit zero of the sector address word is set to 1 if there was a file-protect status for that sector.

Error Messages

If the wrong disk has been loaded, or there is an error in reading from or writing to magnetic tape, explanatory messages will be output to the typewriter.

Core Requirements	1294 and 1280 words respectively.
Programmer	Cathy Clayson.

Title	Disk to magnetic tape and magnetic tape to disk dumping programs.
Names	DIMAG and MAGDI
Machine	I.B.M. 1800
Operating System	T.S.X.
Language	1800 Fortran IV
Purpose	To keep a record of the dumps which take place from disk to magnetic tape and vice versa.

Control Cards

```
//bJOB          cc 19      N.B. No 'X' must be present
                  X             when executing MAGDI

//b* (Project/Name/Title)

//bXEQ  DIMAG
    or  MAGDI

*FILES (1∅, DIREK, ∅)

*CCEND
```

Before executing MAGDI, one must ensure that the file protect status on all sectors of the disk is turned off. To do this, precede the above control cards with the following:-

```
                cc19
//bJOB            X

//bDUP

                cc11     cc20      cc25      cc30
*DWRAD            1        000       63F        F
```

Input Data

The following two data cards come immediately after the *CCEND card and are:-

1) ITITL
   Format:80A1
   which is a brief description of the contents of the disk or magnetic tape file.
2) NAMT, IDISK, IDRIV, IDECK, IDATE
   Format: 10X,3A2,4X,13,2X,I1,2X,I1,2X,5A2 where:

   1) NAMT - is the name of the magnetic tape to or from which the dump is to take place.
   2) IDISK - is the number of the disk from or to which dumping takes place.
   3) IDRIV - is the drive number of the disk.
   4) IDECK - is the magnetic tape deck to be used (ie. 1 or 2).
   5) IDATE - is the date of the dump,

   and in the case of MAGDI, a 3rd data card follows and should contain
   IFILE
   Format: 15
   which is the number of the file on magnetic tape at which reading commences.

Method

In DIMAG, the directory file is searched for a previous entry for the magnetic tape, NAMT. If there is no previous entry, then a header file of 321 words containing the mag. tape label (e.g. M51) is written to the 1st file on the mag. tape 50 times, and then writing of the disk data starts at file 2.
If there is already an entry, the magnetic tape is wound to the next available file and writing commences there.
N.B. The file number should not exceed 9, which is 1+ the number of disks which can be written to one particular magnetic tape.
The program then calls DIKTR to do the dumping, after which an 'end-of-file' marker is written, the directory updated and a message output to the 1443 printer and typewriter.
In MAGDI, the directory is searched for an entry for the required file and mag. tape and the header record is checked to see if the correct tape has been mounted. The tape is then wound to the file IFILE ready for reading to commence. The TDIKW subroutine is called, to dump that file back to disk, after which the appropriate entry in the directory is deleted and a message is output to the 1443 printer and typewriter.
The DIREK directory file has initially been set up with 49 records each 12 words long. (Although it may need to be enlarged at a later date). The first record contains the current number of entries in the directory (N.B. Each mag. tape may have 8 entries allocated to it, since approx. 8 disk dumps can be stored on mag. tape). Subsequent records contain a directory of every dump which has taken place from disk to mag. tape. The format is as follows:-

```
Word 1)
     2)  Name of the mag. tape.
     3)
     4   File to which disk data was written.
     5   Number of the disk.
     6   Disk drive number.
     7   Magnetic tape deck used.
     8)
     9)
    10)  Date of the dump (e.g. 29/08/1970).
    11)
    12)
```

N.B. Word 1 contains 'Mb' in A2 format when records are not currently in use, and after a dump from mag. tape to disk takes place, in which case the entry is deleted.
The directory file may be examined using the program DDUMP, which dumps the file to paper tape. To re-dump the paper tape to the disk file use DSAVE.

Output

Self-explanatory messages are printed out both on the 1443 printer and typewriter when dumping is complete.

Error Messages

Errors may occur if a magnetic tape with the wrong mode is mounted (i.e. with a mode other than Ø or 1), or when the DIREK directory must be enlarged.
(The mode is Ø for the 1st and 1 for the 2nd tape drive)

Restrictions

When dumping from magnetic tape back to disk it is advisable to use a disk containing no defective cylinders, since it may be a disk other than the one from which dumping took place.

If it is different, then *DLABL cards must follow the entire job, to just re-label the disk. Please see the Disk Librarian before doing this, as otherwise chaos may ensue!

<u>Execution Time</u>	Approx. 5 minutes.
<u>Programmer</u>	Cathy Clayson.

N.I.O. PROGRAMS 224 and 225 CLASSIFICATION S

Title	To dump and save the directory file DIREK
Names	DDUMP and DSAVE
Operating System	T.S.X.
Machine	I.B.M. 1800
Language	1800 Fortran IV
Purpose	To dump out the contents of DIREK (see programs 204 and 205) to paper tape and to re-constitute the data file by dumping the paper tape data (with or without alterations) back to disk.

Control Cards

```
 //bJOB
//b*(Project/Name/Title)
//bXEQ DDUMP)
       DSAVE)
*FILES (1Ø, DIREK, Ø)
*CCEND
```

Output	In the case of DDUMP a listing of the paper tape produced will show the current status of the directory file dumps. There is no output from DSAVE; the directory file is overwritten with the contents of the paper tape only.
Programmer	Cathy Clayson.

N.I.O.SUBPROGRAM -92 CLASSIFICATION U

Title	Tape Security System (TASS):Find a File and Record
Name	Subroutine FINDM
Machine	I.B.M.1800
Language	1800 Fortran IV

Purpose

An execution of

```
CALL FINDM(IDECK,NFIL,MREC)
```

will access record MREC in file NFIL of the mag. tape in use on tape deck IDECK.
IDECK should contain the number 5 or 10, depending on whether deck 1(LUN5) or 2(LUN10) is to be used.

Restrictions

$NFIL \leq 999$
$MREC \leq 32767$

The subroutine will work only with Fortran formatted I/O tapes written on an IBM 1800.

Notes

See DPG/P/7 and DPG/P/22 for notes on the TASS system

Programmer

Cathy Clayson.

N.I.O. SUBPROGRAM -93   CLASSIFICATION U

Title	Tape Security System (TASS): Find next available file and record.
Name	Subroutine NEXTM
Machine	I.B.M.1800
Language	1800 Fortran IV

Purpose

An execution of

```
CALL NEXTM(IDECK,NMAG,NFIL,MREC)
```

will access the next available record MREC in file NFIL (as found by searching disk file MFILE) of magnetic tape NMAG.
NMAG is the magnetic tape in use on the tape deck IDECK; where NMAG is a 3 word integer array and should contain the name of the magnetic tape in 3A2 format (left-justified), and IDECK should contain the number 5 or 10 depending on whether deck 1(LUN5) or 2(LUN10) is to be used.

The mainline program must contain

```
DEFINE FILE 201(301,5,U,NREC)
```

and after the // XEQ card

```
*FILES(201,MFILE,0)
```

an *IOCS(MAGNETIC TAPE,DISK,1443 PRINTER) card should also be present.

Output

The message

'TAPE NAMT POSITIONED AT RECORD MREC ON FILE NFIL'

is output to the 1443 printer.

Restrictions	The subroutine will work only with Fortran formatted I/O tapes written on an IBM 1800.
Notes	See DPG/P/7 and DPG/P/22 for notes on the TASS System.
Programmer	Cathy Clayson.

N.I.O.SUBPROGRAM -94 CLASSIFICATION U

Title	Tape Security System (TASS):Store current tape position
Name	Subroutine STORM
Machine	I.B.M.1800
Language	1800 Fortran IV

Purpose

An execution of

```
CALL STORM(NMAG,NFIL,MREC)
```

will search disk file MFILE for the record containing an entry for the mag. tape NMAG. Then it will store the next available record number MREC and file NFIL in that record.

NMAG is a 3 word integer array and should contain the name of the magnetic tape in 3A2 format (left-justified).

The mainline program must contain

```
   DEFINE FILE 201(301,5,U,NREC)
```

and after the //XEQ card

```
   *FILES(201,MFILE,0)
```

an *IOCS(MAGNETIC TAPE, DISK, 1443 PRINTER) card must also be present.

Output

The message

```
'NEXT AVAILABLE RECORD ON TAPE NMAG IS MREC ON FILE NFIL'
```

is output to the 1443 printer.

Restrictions

$NFIL \leq 999$
$NREC \leq 32767$

Notes

See DPG/P/7 and DPG/P/22 for notes on the TASS system.

Programmer

Cathy Clayson.

N.I.O.PROGRAM 211 CLASSIFICATION U

Title Tape Security System (TASS):Initialisation of MFILE records.

Name INITM

Machine IBM 1800

Language 1800 Fortran IV

Purpose To initialise or amend an existing record of the disk file MFILE with the next available file number and record number of the requested magnetic tape (See DPG/P/7 and DPG/P/22)

Input (e.g.)

```
                //JOB
                //*Project/Name/Title
                //XEQ INITM
                *FILES(201,MFILE,0)
                *CCEND

cc 5     cc  17      cc  24
  M5          2          1

  M6          1        100
cc1
  *END
```

The above job will initialise the record for M5 with file 2, record 1, and for M6 with file 1, record 100.

The data ends with *END

Output The output consists of one line for each mag. tape as follows (e.g.):-

```
TAPE SECURITY RECORD FOR M6 INITIALISED WITH FILE
NUMBER 1 AND RECORD NUMBER 100
```

Execution Time Approximately 1 sec. per input data card

Programmer Cathy Clayson

N.I.O.PROGRAM 212	CLASSIFICATION U
Title	Tape Security System (TASS): Query File
Name	QUERM
Machine	IBM 1800
Language	1800 Fortran IV
Purpose	To examine the current status of the disk file MFILE, giving details of the next available files and records of all the N.I.O. magnetic tapes in use under the TASS system. (see DPG/P/7 and DPG/P/22).

Input

```
//JOB
//*Project/Name/Title
                  cc 16
//XEQ QUERM          FX
```

Output For every magnetic tape in use:- (e.g.)

```
TAPE SECURITY SYSTEM - NEXT AVAILABLE RECORD
ON TAPE M6 IS 532 ON FILE 3
```

Execution Time Approximately 30 secs. per 100 tapes in use.

Programmer Cathy Clayson

DATA PROCESSING GROUP

N.I.O. TApe Security System (see DPG/P/7)

The following programs belonging to the TASS system have been updated and given completely new names and numbers. This is so that the second magnetic tape unit may be incorporated into this system.

They are as follows:-

N.I.O.115	INITO	replaced by	N.I.O.211	INITM
N.I.O.116	QUERO	replaced by	N.I.O.212	QUERM
N.I.O.117	PACKO	deleted from system		
N.I.O. -6	FILEO	replaced by	N.I.O.-6/A	FILE2
N.I.O.-15	FINDO	replaced by	N.I.O.-92	FINDM
N.I.O.-16	NEXTO	replaced by	N.I.O.-93	NEXTM
N.I.O.-17	STORO	replaced by	N.I.O.-94	STORM

These new programs will be implemented as from 16th. November 1970.

Notes

1) The original programs will still exist for a while but if they are inadvertently used after the implementation date, an explanatory message will be output, telling the user to use the updated program.

2) A brief description of the new programs follows:-

INITM:	Initialisation of records in the disk file MFILE (see note 3)
QUERM:	Print out contents of MFILE
FILE2:	Find a file; now also prints out which mag. tape deck was used when error occurred (can also be used on unformatted and BCD tapes)
FINDM:	Find a file and record; calling sequence is now

```
CALL FINDM(IDECK,NFIL,MREC)
```

(IDECK contains the LUN of the mag. tape deck to be used, i.e. 5 or 10 - exit otherwise).

NEXTM: Find next available file and record; calling sequence is now

```
CALL NEXTM(IDECK,NMAG,NFIL,MREC)
```

(IDECK contains the LUN of the mag.tape deck to be used (see FINDM); and NMAG is a 3 word integer array containing the mag.tape name in 3A2 format).

STORM: Store current tape position; calling sequence is now

```
CALL STORM(NMAG,NFIL,MREC)
```

(NMAG contains the name of the mag.tape (See NEXTM)

3) The individual M files now situated on disk are being merged into one single file called MFILE consisting of records of 5 words as follows:-

Record 1	Word 1	-	Number of entries
	Words 2-5	-	Not used
Records 2 onwards	Word 1	-	Name of magnetic tape
	Word 2		
	Word 3	-	
	Word 4	-	Next available file
	Word 5	-	Next available record

4) The program descriptions for the original programs should now be amended to read:-

<u>Purpose</u> To inform the user that this program is now out of date and that he must use its equivalent - (fill in name)

<u>Output</u> A message is printed out on the 1443 printer telling the user of this change.

The error messages in the existing description of FILE2 (N.I.O.-6A) should be amended to read:-

'UNCORRECTABLE TAPE ERROR IN FILE NUMBER n ON DECK m'

and

'READ CHECKS HAVE OCCURRED IN FILE NUMBER n ON DECK m'

DPG/P/22

Cathy Clayson

29.9.70

N.I.O.SUBPROGRAM -95 CLASSIFICATION S

Title	Determine first and last record in a dynamic file
Name	Subroutine IFREC
Machine	1800
Language	Fortran IV
Calling Sequence	CALL IFREC (IFIL, IFST, LAST)

where IFIL is the logical file number of the data file
IFST will be set to the first record of real data
LAST will be set to the next available record on the file

Method, The program uses the dynamic data file header. The word NEXT in this header is not updated by this subroutine.

Programmer M. Fasham.

N.I.O.SUBPROGRAM -96 CLASSIFICATION S

Title	Finds the record number of a dynamic data file entry having a given day and time.
Name	Function JFND
Machine	1800
Language	Fortran IV

Calling Sequence

J = JFND(IFIL,ID,IT,JST)

where IFIL is the logical file number.
ID is the day number of the required record.
IT is the time in tenths of a minute of the required record.
JST is the record number at which the search is to start.

If JST is put equal to zero the whole file will be searched.

After the call J will be equal to the record number of the file entry whose day and time EXACTLY equals ID and IT. If no record can be found J will be put to zero.

Subroutine called

IFREC

Programmer M. Fasham

N.I.O.SUBPROGRAM -99 CLASSIFICATION S

Title Calculate algebraic difference between two times

Name Function DFMIN

Machine 1800

Language Fortran IV

Calling Sequence J = DFMIN (LDY1, LT1, LDY2, LT2)

where LDY1 is day number of first time
LT1 is time in 1/10 minute of first time
LDY2 is day number of second time
LT2 is time in 1/10 minute of second time
J will be equal to the algebraic difference between the two times in 1/10 minute. That is if the second time is temporally before the first time J will be positive.

Programmer M. Fasham.

N.I.O. SUBPROGRAM -118 CLASSIFICATION S

Title	BCD tape handling utility subroutine
Name	SLSET
Machine	IBM 1800
Operating System	TASK or TSX
Language	1800 Assembler

Purpose

SLSET is a relocatable call subroutine containing ten entry points which allow the user to perform various functions on 7-track BCD (2bytes/word) magnetic tape. The subroutine was written to be used by THUMP (N.I.O. Program 133), but may also be used by other FORTRAN or ASSEMBLER programs.
The magnetic tape format that can be handled by SLSET is defined in the program description for THUMP and will not be re-iterated here. This program description contains a summary of the calling sequences to SLSET as it could be used by programs proeessing 7-track BCD 2-byte per word magnetic tape.

Use

SLSET is a call subroutine and contains ten entry points. Before any of the other nine calls are made, a

```
CALL SLSET(MBKSZ,MAGCW)        - Fortran
or
CALL      SLSET
DC        ADMBK
DC        ADMAG
.
.
```

```
ADMBK        DC        MBKSZ
ADMAG        DC        MAGCW           - Assembler
```

must be made. This sets the tape block size (MBKSZ) and the tape density, parity and tape unit (MAGCW) within SLSET.

MBKSZ must be even and is equal to one-half the number of characters per tape block.

MAGCW is in the form of a hexadecimal number and corresponds to hexadecimal digits 2, 3 and 4 of the control word used by MAGT: MAGCW would normally be defined in a DATA statement as:

```
DATA MAGCW/ZOABC/
```

where A = 0 for odd bit parity

A = 1 for even bit parity

B = 4 for 800 bits/inch density

B = 5 for 200 " " "

B = 6 for 556 " " "

C = 0 for the first tape unit

C = 1 for the second tape unit

Alternatively MAGCW may be generated by

MAGCW = 256*A + 16*B + C, assuming that the variables corresponding to A, B and C are integers. The nine operational entry points will now be discussed.

1. <u>SERCH</u> A call to SERCH of the form CALL SERCH(NAMEF,MFILE,MERR) where NAMEF is a one-dimensional array of length 7 containing the name of the file to be found in 5A1 format, and the version number and modification level as the sixth and seventh members of the array, will position the first block of the file NAMEF under the read head and return the file number in MFILE.

   If the name file requested is not on the tape, MFILE will be set to zero.

2. <u>SFILE</u> A call to SFILE is for positioning the tape to the first block of a numbered file.

CALL SFILE(MFILE,MERR) where MFILE is the file number requested. If MFILE is greater than the number of files on the tape, then MERR will be set to non-zero; if the file is on tape MERR, is set to zero.

3. SLIST A CALL SLIST will cause the tape file currently under the read head (e.g. as positioned in a preceeding CALL SERCH) to be listed on the 1443 printer.

4. SPNCH A CALL SPNCH will cause the tape file currently under the read head to be punched onto cards.

5. STTLE A CALL STTLE will cause the headers of all files on the tape to be listed on the 1443 printer. The header of a file is defined as the first 80 characters of the first block of a file.

6. SPZER A CALL SPZER will write the end-of-all-data file on the tape, starting at the point on the tape currently under the write head.

7. SWIPE A CALL SWIPE will cause the name 2DUMY to replace the name in the header of the current file, the original name being placed in character positions 9 to 13 of the header. This function may cause read checks to occur when the tape is read subsequently. It is included for completeness, but not for general use.

8. SPINT A CALL SPINT will initialise a magnetic tape ready for use by the N.I.O. 2 byte BCD tape handling programs. All previous data on the tape is lost.

9. SPLOD This is the routine to load information from cards to tape in 2-byte BCD format. The card deck to be loaded must be followed by a card with a £ or ¢ punched in column 1.
CALL SPLOD (NAMEF, MSEQ, MCARD)
where NAMEF is a one dimensional array of length 7 containing the file name in 5A1 format and the version and modification level to be assigned to this file as the sixth and seventh members of the array. The version and modification will be unsigned integers (of 5 BCD characters each) in character positions 21 to 25 and 27 to 31 respectively of the file header.
They may be zero. NCARD will be returned as the number of cards converted and loaded to tape. If MSEQ is non-zero then the deck will not be re-sequenced; if zero then columns 73-80 of the card images on tape will be automatically sequenced in the form NAMXXXXO where NAM is the first three letters of the file name and XXXX increases (linearly) from 1 to 9999.
Further details concerning tape format are described in the program description 133 (THUMP).

Functions SLIST and SPNCH leave the tape at the first block of the next file when control is returned to the calling program.
Functions STTLE and SWIPE rewind the tape before returning control.
Functions SPZER, SPINT and SPLOD leave the tape at the next available point on the tape for loading information (i.e. at the beginning of the end-of-all-data) file.

Subroutine Used	MBUFF, DICEB, DICHO, HOLDI, ZIPCO.
Core requirements	822 words.
Programmer	David Brown.

N.I.O. PROGRAM 133

Title	BCD Tape Handling and Utility Management Program
Name	THUMP
Machine	I.B.M. 1800
Operating System	TASK or TSX
Language	FORTRAN

Purpose

THUMP interfaces with the tape handling subroutine SLSET to provide facilities for:

1. Loading information on cards onto magnetic tape.
2. Printing or punching information from magnetic tape onto the lineprinter or cards respectively.
3. Initialising a tape ready for Functions 1 and 2.
4. Deleting the reference to a file from the tape.

These functions are called in the subroutine SLSET depending on the THUMP control cards which follow the //XEQ THUMP, *CCEND cards. The program supports two 7-track tape drives, 200, 556 and 800 b.p.i. density, and even or odd parity. Only 72 card columns of real data are allowed, columns 73-80 are automatically written on tape as sequence numbers of the form NAMXXXX0, where NAM are the first three letters of the file name and XXXX0 is the sequence number which is incremented by 10 for each card loaded to the current file.

The first card-image of 80 characters of each file contains the header information for that file of the form:

```
cc                  cc           cc                          cc
1                   21           27                          80
NAMEF                VVVVV        MMMMMM                     *
```

NAMEF is the name of the file, VVVVV is a version number, and MMMMM is the modification level (these last two are used primarily if the file is a source program).

A file is created as tape blocks of size determined by the first THUMP control card, and is written in BCD as two characters per computer word. A standard block size (as used for EPL source program tapes) of 2,000 characters is recommended, this is also the size used for the N.I.O. Program Library on magnetic tape, at even parity.

Magnetic Tape Format

After careful consideration, the format of information on tape was standardised almost to that used by IBM-EPL for 7 track BCD tapes.

The tape data consists of files, separated by a tape mark. Each file consists of blocks of information, typically 2000 characters/block, and this figure must be divisible by 80 to allow complete card-images to be stored. The first file (file 1) consists of one block and is not used. The last file is followed by a tape mark, then one block containing 80 characters of BCD '9' (i.e. hexadecimal),
/09
- and another file mark. The tape data is undefined past the end-data file.

Each file created by THUMP has a header as the first 80 characters of the file, consisting of:

Char	Char	Char	Char
1	21	27	80
NAMEF	VVVVV	MMMMM	*

The name (NAMEF) and version and mod. level (VVVVV and MMMMM) are supplied by the load function. EPL source program tapes have the name as characters 1-5, but not necessarily the version and mod. level.

Although EPL use odd bit parity, this is not recommended for user created tape files, and even bit parity should be used.
Density of 800 bits per inch is recommended.

)peration A mixture of control cards, information to be loaded cards and blank cards may follow the //XEQ THUMP, *CCEND cards. The first card read by THUMP must define the block size in characters, the tape parity, tape recording density, and the tape unit to be used. The format is:

```
cc         cc    cc    cc
1          6     8     12
BBBB       P     DDD    U
```

where BBBB is the number of characters/block (right justified). This must be even and is typically 2000.

P is the parity, 0 = odd, 1 = even. Even parity is usual for BCD tapes.

DDD is the tape density and can be 200, 556 or 800 bits per inch.

U is the tape unit selected, 1 or 2.

The next card read will determine the first function to be carried out by THUMP. These cards must all have the card code 11-8-3 in column 1; this is the $ code, as printed, but on English 029 card punches must be punched as £ sign.

The format of the THUMP control cards is

```
cc      cc              cc     cc     cc
1       10              21     27     33
$FFFFF  NNNNN            NAMEF VVVVV MMMMM
```

where FFFFF is the function to be performed and may be

1. $LOAD - to create the next sequential file on the tape. The data to be loaded must immediately follow this card.

2. $END STACK - signifies the end of a card deck used in 1.

3a. $ZERO - to clear the tape of all files subsequent to and including NAMEF.

```
               cc
               21
3b.$ZERO     - $$$$$ to initialise a magnetic tape ready to be
```

used for loading files. All previous data on the tape is lost. This must be the first function performed on a new tape.

4. ØWIPE - to delete the header of the file named NAMEF. This function is not recommended, as read checks may occur whenever the tape is read subsequently.

5. ØLIST - to list the file on the line printer.

6. ØPUNCH - to punch the file onto cards.

7. ØNAMES - to print the headers of all files on the tape.

8. ØPAUSE - to cause THUMP to pause until START is pressed.

9. ØEND THUMP - to cause THUMP to call exit.

Functions 5 and 6 may be used to either list/punch a numbered or named file. If card columns 10 to 14 are blank, then the headers of files are checked for the name NAMEF. If card columns 10-14 contain a positive integer, then that file number is dumped.
The version number and modification level to be written in the file header, if the data is source coding, are VVVVV and MMMMM. These columns may be blank if VOMOO is assumed, i.e. there will not be duplicate names on any one magnetic tape.
After completing the function specified by the last control card, THUMP reads another card and carries out the function specified on that card. The last card of the stack must be a ØEND THUMP card.

Subroutines Used SLSET (and other 9 entry points) ZIPCO, DICHO, HOLDI, STACK, MBUFF.

Timings Double buffering is used for ØLOAD function, and the card read is at 260 cards/minute. Printing ØLIST is about 120 lines/minute. The search time to locate a file on magnetic tape is about 0.5 second/block.

Restrictions

1. The program functions LIST,PUNCH, WIPE, and ZERO ignore the VVVVV and MMMMM fields, so only one file of any one name may exist on one tape.
2, The LIST function prints certain special characters consistently incorrectly as no conversion occurs.

Programmers David Brown

N.I.O. PROGRAMS 226 and 227 CLASSIFICATION S

Title	Change drive 1 data disk to a drive 2 data disk or vice versa
Names	D1TO2 and D2TO1
Machine	I.B.M. 1800
Language	1800 Assembler
Purpose	To alter the starting sector address word of all FLET entries on the disk in question according to whether a drive 2 or a drive 1 data disk is required.

Control Cards

```
                  cc19                            cc19 cc24
//bJOB             X          or      //bJOB       X    X

//b*(Project/Name/Title)              //b*(Project/Name/Title)

//bXEQ D1TO2                          //bXEQ D2TO1

*CCEND                                *CCEND
```

Input Data	None
Output	None
Method	The program reads the 1st word of sector 1 (logical) for the number of the sector (N) of the start of FLET for the disk in question and then reads that sector N into core. FLET should always start and end with a .E entry and so the last entry in FLET (words 317-320) is checked to ensure that it is a .E entry (exit if not) and then the program works backwards through the sector, changing bit 3 of every 4th word (the starting sector address word) until the first .E entry is encountered. The sector is then written back to disk in its new state. If the end of the sector is reached before the first .E the sector is still written back to disk but then the (N-1)th sector is read into core and the process repeated
Execution Time	Approx. 2 mins. depending on size of FLET.
Programmer	Cathy Clayson.

N.I.O. PROGRAM 231	CLASSIFICATION S
Title	Re-label a drive 1 or drive 2 disk
Name	LABEL
Language	1800 Assembler
Machine	I.B.M. 1800
Operating System	T.S.X.
Purpose	To change the cartridge ID number on a drive 1 or drive 2 disk.

Input

One data card containing the drive number and the new cartridge label in the form

Card column

```
1 3   7
DbLLLLL
```

where D is the drive number and LLLLL is the label, integer right justified.

Output

The new and old label are written on the console typewriter, in the form

```
NNNNN+OOOOO
```

where NNNNN is the new label and OOOOO was the old label

Use

```
//bJOB
//b*PROJECT No./NAME/TITLE
//bXEQbLABEL
*CCEND
```

Data card (see under Input)

Programmer William Strudwick.