

DISPLAY

COPY

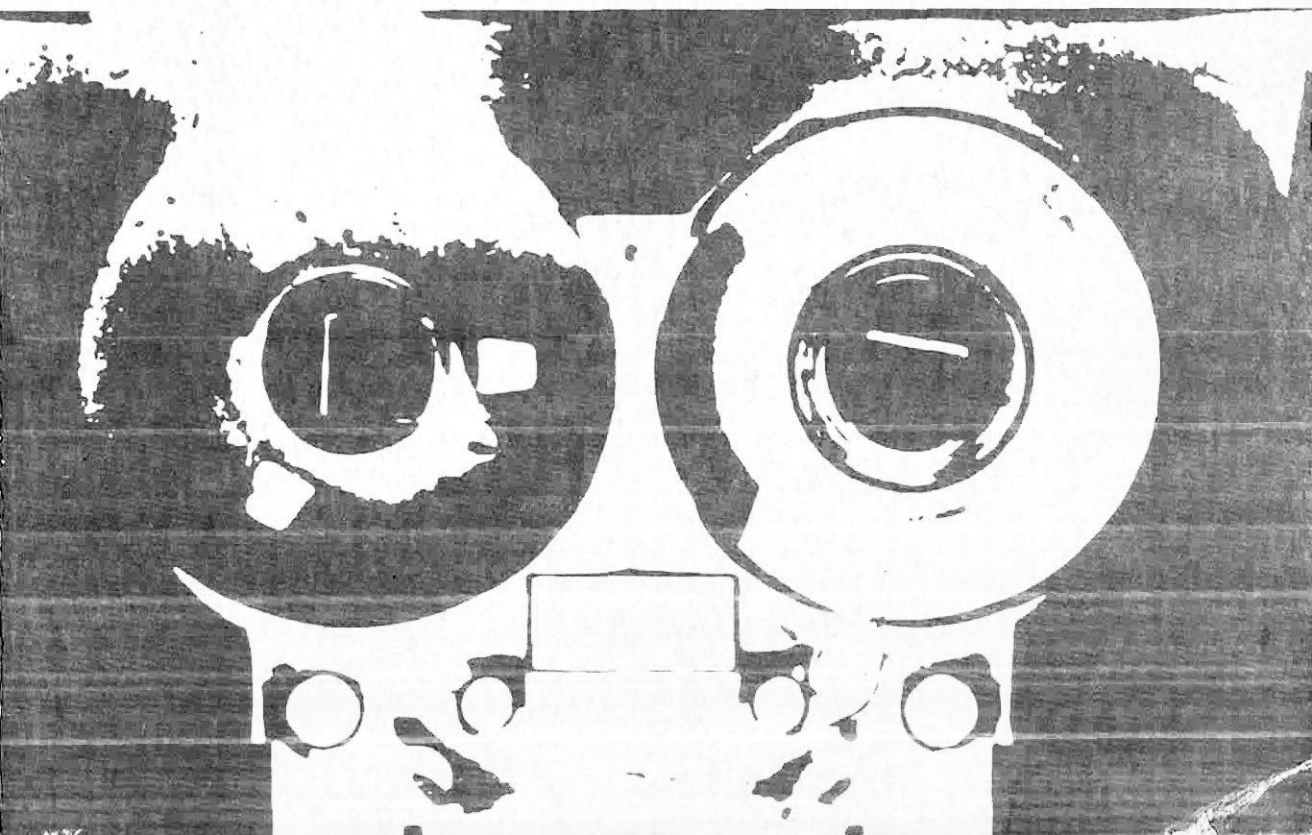
Please, Do Not Remove

REVISED EDITION

3238

**INTRODUCTION TO
THE IBM 360 COMPUTER
AND OS/JCL
(JOB CONTROL LANGUAGE)**

Judith Rattenbury




**INTRODUCTION TO
THE IBM 360 COMPUTER
AND OS/JCL
(JOB CONTROL LANGUAGE)**

Judith Rattenbury

Revised Edition

Survey Research Center • Institute for Social Research
The University of Michigan
Ann Arbor, Michigan



ISR Code No. 3238

Library of Congress Catalog Card No. 74-634773
ISBN 0-87944-010-4 paperbound
ISBN 0-87944-011-2 clothbound

The Institute for Social Research
The University of Michigan, Ann Arbor, Michigan 48106

© 1971 by The University of Michigan, All Rights Reserved
First Published 1971
Revised Edition 1974
Manufactured in the United States of America

PREFACE TO THE SECOND EDITION

Since the first edition of this monograph, several new releases of the OS operating system have been issued by IBM. In addition, the PCP version of OS, which the ISR was using, is now practically obsolete. Nearly all IBM/360 installations using OS use the MFT (multiprogramming with a fixed number of tasks) or MVT (multiprogramming with a variable number of tasks) versions. IBM has announced that no further changes will be made to OS which is "functionally stabilized" at release 21.6.

Some minor changes have been made in the text to reflect these system changes and a few points not covered previously have been added.

Ellen Bronson is responsible for the excellent typing of this edition.

Judith Rattenbury
December 1973

PREFACE

Computers are considered essential tools in many branches of research today. In particular, in the field of social research, the computer has come to be more and more heavily relied upon for the processing and analysing of data. To make use of this highly complex and flexible tool, the researcher has three main alternatives: firstly, he may devote 1-2 years to learning the detailed intricacies of computer programming and to becoming a computer specialist himself; secondly, he may hand over responsibility for all computer work he thinks he needs to a computer specialist; thirdly, he may learn just enough about the basic principles of computers to be able to make use himself of already written general purpose computer programs. The first alternative is time consuming and not readily available to many people; the second is not only costly but can lead to real communication problems, with the researcher missing possible different and valuable approaches to his data by not knowing the potential of the computer; the third, which assumes that use will be made of work already produced by computer specialists, and which lets the researcher make use of the computer in the way he wants with a minimum of assistance from others, seems the most reasonable approach.

The state of the art of program development has reached a point where there is now a wealth of already written general purpose programs available for a variety of purposes. This is especially true in the field of social research, where the majority of processes to which data are subject are similar in many research projects. Yet, to avail himself of these programs (which have been developed over the years by numbers of computer specialists) it is essential that the research scientist have some passing familiarity with the computer and know how to get it working on his data.

In the past, computer operators kept copies of programs in the computer room and knew a certain amount about and how to use each. The computer user had merely to hand the computer operator his data (usually on cards) with verbal instructions as to what he wanted done. With computers today, not only are the types of application more complex and the ways of storing data more variable, but also complex computer operating systems have replaced many of the functions previously performed by human operators. Thus, instead of giving verbal instructions to the operators, the computer user must provide instructions to the computer itself. These instructions--giving, for instance, the particular program and detailed descriptions of data files to be used--have to be prepared in a special language which the computer can understand.

The IBM 360 is a third generation computer which can run under the control of various different operating systems. Having found that a particular program

is available which he wants to use, the computer user must prepare instructions in "Job Control Language" informing the computer of his requirements. IBM manuals for the 360 describe in detail the vocabulary and syntax rules for the job control languages for the various operating systems. However, these are really reference manuals to the already initiated and are incomprehensible to others. It is in fact essential to have a good grasp of the basic principles of computer processing of data, the media on which data can be stored (e.g., cards, tape, disk) and to some extent how an operating system works before the job control language can be understood and used to full advantage.

In the following pages, an attempt is made not only to give details of the most used subset of the job control language for the IBM 360 computer OS operating system in clear and concise form, but also to relate the use of different features of the job control language to the computer in a meaningful way. Thus, Chapter I gives a brief introduction to the hardware and software of a computer and the purpose of an operating system and its job control language. Chapter II gives the basic rules which all job control language statements must follow and then goes into details on how to specify the first two types of job control statements (JOB and EXEC cards). The third type of information that has to be specified in job control language defines data files to be used. Before this is covered in detail, Chapters III, IV, and V are devoted to describing how data is stored on computer media and in particular the physical aspects of magnetic tapes and disks. With this knowledge, the data definition parameters described in Chapters IV and VII become reasonably logical and easy to understand. Chapter VIII describes how to set up procedures (standard sets of job control statements which can be used over and over again). Chapter IX contains a general description of how job control statements are handled by the computer, together with explanations of the output printed by the operating system's control program at the beginning and end of a job and also gives details of the different types of errors that can occur while a job is being run, their causes and their correction.

Numerous examples are given throughout the text, and copies of actual computer output are presented and explained. Several exercises are also included.

Since the text was prepared for use by researchers and their assistants at the Institute for Social Research, at the University of Michigan, examples in some cases are peculiar to use of the IBM 360/40 at the ISR. At the time of writing, the operating system being used at the ISR was "OS release 18" with the primary control program (PCP); some parameters relevant only to multiprogramming versions of OS (MFT and MVT) are therefore not discussed. It is felt, however, that the bulk of the text is applicable and relevant to anyone using an IBM 360 running under the OS operating system.

With no previous experience of computers (except perhaps a familiarity with the punched card), any interested person after reading this text should be capable of preparing job control statements, running his own jobs and detecting and correcting errors. Those with experience of other computers may also find it a useful introduction to the use of an IBM 360 computer.

I am grateful to the many people at the Institute for Social Research who have shown enthusiasm in learning about computers instead of relying on specialists. They provided me with the stimulus to produce this introduction which was put together from notes written for classes given to staff members to introduce them to the IBM 360 computer and to enable them to run their own jobs on the ISR's computer. Special thanks should go to Nancy Mayer, without whose patience the typing would never have been completed.

Judith Rattenbury
January, 1971

CONTENTS

	<u>Page</u>
Preface	iii
List of Handouts	ix
I. Introduction to a Computer and Its Job Control Language	1
1.1 Basic Components of a Computer	
1.2 Memory	
1.3 Input and Output	
1.4 Hardware at ISR	
1.5 Software	
1.6 Operating System	
1.7 Control Program	
1.8 Job Control Language	
1.9 Definition of a Job	
1.10 JCL Statements for the IBM 360 OS Control Program	
II. JOB and EXEC Statements	7
2.1 JOB Card	
2.2 Form of the JOB Card Used at the ISR	
2.3 Examples of Good Job Cards for Use at the ISR	
2.4 Possible Errors on Job Cards	
2.5 Explanation of Handout 2	
2.6 EXEC Card	
2.7 Examples of EXEC Cards	
2.8 Possible Errors in the EXEC Card	
2.9 Examples of Errors on EXEC Card	
Handout 1 (Job Card Examples)	77
Handout 2 (Computer Output from Using the Job Cards from Handout 1)	78
Handout 3a (HASP Log Output from an Abnormal Termination)	80
Handout 3b (System Completion Code at an Abnormal Termination)	81
Handout 4 (EXEC Card Examples)	82

Handout 5 (Computer Output from Using the Execute Cards from Handout 4)	83
III. Introduction to DD statements and Data Formats	15
3.1 DD Cards--Introduction	
3.2 Data	
3.3 Records	
3.4 Blocks	
3.5 Advantages of Blocking	
3.6 Limitations on Blocksize	
3.7 Summary of Record Types	
3.8 Block Length	
3.9 Buffers	
IV. Tapes	21
4.1 Physical Characteristics	
4.2 Tape Labels	
Handout 6 (Standard Labelled Tape Description)	86
V. Disks	27
5.1 Physical Characteristics	
5.2 Volume Table of Contents	
5.3 Writing Data Sets to Disk	
5.4 Deleting Data Sets to Disk	
5.5 Data Set Organization	
5.6 Disk Characteristics	
Handout 7a (VTOC Listing by Program IEHLIST)	87
Handout 7b (VTOC Listing by Program SPVTOC)	88
Handout 8 (Listing of Catalog)	89
VI. DD Statement Parameters	33
6.1 General DD Card Format	
6.2 DISP Parameters	
6.3 DSNNAME or DSN Parameter	
6.4 Possible Errors in DISP and DSN Parameters	
6.5 LABEL Parameter	
6.6 UNIT Parameter	
6.7 VOLUME or VOL Parameter	
6.8 DCB Parameter	
6.9 Exercises on DD Cards for Input Files	
6.10 SPACE Parameter	
6.11 Exercises on DD Cards for Output Files	
6.12 Which DD Parameters Are Needed	
Handout 9 (DD Card Error Examples (DISP and DSN))	90
Handout 10 (Exercises on DD Cards for Input Files)	91

	<u>Page</u>
Handout 11 (Exercises on DD Cards for Output Files)	92
VII. Special DD Statements	51
7.1 Specifying the Printer, Card Punch, and Card Reader	
7.2 Program Libraries--JOB LIB, STEPLIB	
VIII. Catalogued Procedures	55
8.1 Purpose	
8.2 Description	
8.3 Example of Procedure	
8.4 Overriding Cards in a Procedure	
8.5 Output Example of SCRAT Procedure	
8.6 Adding Extra DD Cards to Those in a Procedure	
8.7 Multi-Step Procedures	
8.8 Aliases	
8.9 Creating a Catalogued Procedure	
8.10 Symbolic Parameters in Catalogued Procedures	
Handout 12 (Output from Executing Procedure SCRAT)	93
Handout 13 (Listing of 4 Catalogued Procedures)	94
IX. Allocation, Deallocation, and Common Errors	63
9.1 OS Job Scheduler	
9.2 Allocation Messages	
9.3 Deallocation Messages	
9.4 Types of Errors	
9.5 JCL Errors	
9.6 Completion Codes	
9.7 Errors Generated by FORTRAN Written Programs	
9.8 Explanation of JCL Errors on Handout 15	
9.9 Explanation of Allocation Errors Caused by JCL Errors on Handout 16	
Handout 14a (Allocation Messages)	96
Handout 14b (Deallocation Messages)	97
Handout 15 (Examples of Good and Bad DD Cards)	98
Handout 16 (Examples of Allocation Errors Caused by JCL Errors)	99
References	73
Handouts	75
Index	101

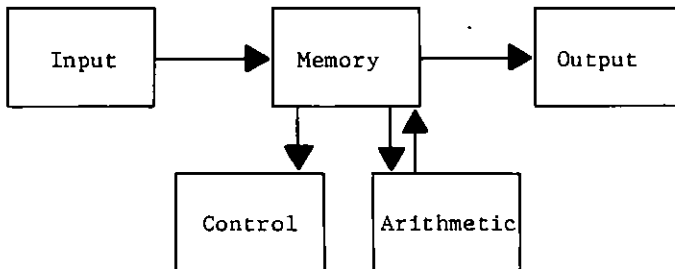
LIST OF HANDOUTS

	<u>Page</u>
1. JOB Card Examples	77
2. Computer Output from Using the Job Cards from Handout 1	78
3. a. HASP Log Output from an Abnormal Termination	80
b. System Completion Code at an Abnormal Termination	81
4. EXEC Card Examples	82
5. Computer Output from Using the Executed Cards from Handout 4	83
6. Standard Labelled Tape Description	85
7. a. VTOC Listing by Program IEHLIST	87
b. VTOC Listing by Program SPVTOC	88
8. Listing of Catalog	89
9. DD Card Error Examples (DISP and DSN)	90
10. Exercises on DD Cards for Input Files	91
11. Exercises on DD Cards for Output Files	92
12. Output from Executing Procedure SCRAT	93
13. Listing of 4 Catalogued Procedures	94
14. a. Allocation Messages	96
b. Deallocation Messages	97
15. Examples of Good and Bad DD Cards	98
16. Examples of Allocation Errors Caused by JCL Errors	99

Chapter 1

INTRODUCTION TO A COMPUTER AND ITS JOB CONTROL LANGUAGE

1.1 Basic Components of a Computer

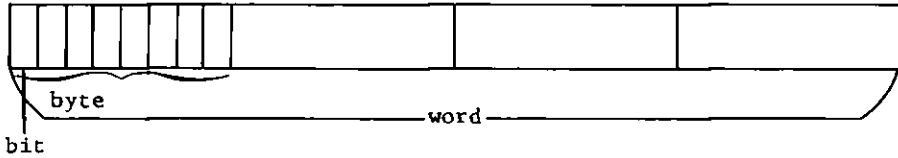


The memory is where all information, both data and instructions for performing the desired operations on the data, is stored. The memory however is empty until something is put into it and this is done by the input units which can read data in various different forms. If some instructions have been read into the memory via an input unit, then they will be taken one by one by the control unit which controls the execution. If arithmetic is to be performed, this is carried out by the arithmetic unit and the results of any computations are stored back in the memory. Finally, since results are of no use in the computer's memory, they are output through the output units onto printed paper or some other medium.

1.2 Memory

The memory consists of many units called bytes, each with a unique address and capable of holding one character (number or letter). Each byte consists of sub units called bits, each of which is capable of holding a zero or a one. Each character stored in a byte is represented by a different combination of

zeros and ones. For many purposes, bytes are grouped to form a new unit called a word. In IBM 360/370 computers there are 8 bits in each byte and 4 bytes to a word.



The size of a computer's memory is normally measured in K bytes where K is equal to 1024 bytes. Thus a 128 K computer has $128 \times 1024 = 131,072$ bytes of memory (also known as core store).

1.3 Input and Output

There are several input and output media with which a computer can cope; with each of these is associated a device or unit which actually does the reading or writing.

1.3a Cards

The punched card is the basic medium on which information can be manually prepared for input into a computer. Each card has 80 columns and 12 rows in which holes can be punched. The holes in one column represent one character so that one card is capable of holding up to 80 characters of information. Information punched on cards is read into a computer's memory by an input unit called a card reader at speeds of up to 1000 cards a minute. Information from the memory can also be punched onto cards by a unit called the card punch at speeds of up to 500 cards a minute. Each character, represented by the holes punched in one column of a card, occupies one byte of computer memory.

1.3b Printed Output

The most basic form of output from a computer is printed output which is normally the only medium which can be read by people. Printed output is produced by an output unit called the line printer which can print at speeds of up to 1300 lines a minute. The number of characters per line varies on different line printers but is normally 132.

1.3c Magnetic Tape

This medium is used for storing large quantities of data compactly. Also, since a computer's memory is of limited capacity, it is sometimes used as a kind of auxiliary memory as work space. Information is magnetically recorded on tracks on the tape at densities ranging from 200 to 1600 characters (or bits) per inch (bpi). A reel of tape is up to 2400 feet long and, if written at a density of 800 bpi, can hold up to 20 million characters of information. There are some devices on the market today by means of which data can be manually recorded directly onto magnetic tape. However, normally it is a medium which only a computer reads or writes. Thus, if data are originally punched onto cards, and

are going to be used several times, they are normally first read into the computer and written to magnetic tape; thereafter the data can be used from the magnetic tape.

Some of the advantages of tape storage of data over cards are: (i) Information can be read into the memory up to 10 times faster, (ii) It is a much more compact form of storage, (iii) Tapes are less likely to get damaged or lost than cards which can be torn, dropped, or mislaid.

Magnetic tape is read and written by a device called a magnetic tape drive or unit. Different units write at different densities and use a different number of tracks. For instance, the ISR has one 7-track tape drive which can write at three different densities (200, 556, 800 bpi), three 8-track tape drives which always write at a density of 800 bpi, and two 9-track tape drives which can read and write either at 800 or 1600 bpi.

1.3d Disk

Disk is a medium read by "direct access" devices of several kinds. The difference between a direct access device and a tape unit can be compared to the difference between a record player and a tape recorder. In order to read data from the middle of a disk, one can either scan through the complete disk or move the read/write head directly to the position of the desired data. On disk, information is recorded magnetically on tracks on the surface which form a succession of concentric circles. The density of data on the inner tracks is greater than on the outer so that each track has the same capacity.

The current disk drives at the ISR are capable of reading and writing a disk pack which contains 8000 tracks of space. This capacity is about equivalent to two 2400 feet magnetic tapes written at 800 bpi density.

1.3e Operator's Log

A log of events on the computer is recorded by a unit called the consol or typewriter, which types information relevant to the computer operator. The operator may also enter commands to the computer through this device.

1.4 Hardware at ISR

The computer currently in the basement at the ISR consists of an IBM 360 Model 40 computer with 384 K bytes of memory, 1 card reader, 1 card punch, 1 line printer, 6 magnetic tape drives and 6 3101 disk drives. This comprises the hardware available.

1.5 Software

The physical aspects of a computer described in sections 1.1-1.4 above are known as the computer's hardware. A computer cannot function however unless there are instructions in its memory telling it what to do. A set of instructions to perform a particular task is called a program and the set of programs available for use on a computer is known as the computer's software.

At the ISR, software can be divided into three classes:

- (i) Programs obtained from IBM
- (ii) General purpose programs supplied by center computing support groups
- (iii) Programs written by users

General purpose programs can be defined as programs which perform one major task with various optional features, on many different data formats. A special purpose program is one which is written to perform one specific task on one specific kind of data.

1.6 Operating System

The main software that IBM provides with a 360 computer is known as an operating system. This consists of amongst other items, compilers (programs to translate instructions written by programmers into machine instructions), utility programs for performing common tasks such as listing cards, etc., and also a "control program."

There are several different operating systems available for the 360. The one used at the ISR is called OS (Operating System). DOS (Disk Operating System) is an example of another used in some installations. In the past, IBM has issued a new release of its OS operating system incorporating new features about every 9 months. However, no more modifications are being made to OS and the stabilized version is release 21.6.

1.7 Control Program

The control program, part of which is kept permanently in the computer memory, is responsible for controlling the execution of jobs given to the computer. In earlier computers, human operators had to be given a lot of verbal instructions in order for the correct user's program with the correct data to be run. On the 360, many of these tasks are now performed by the control program and instead of the user giving instructions to the operator, he must give them to the control program. The control program used at the ISR is called OS/MFT (Multiprogramming with a Fixed Number of Tasks). To assist with scheduling of jobs, reading input and printing of output, a system called spooling is often used. With spooling, all card input is stored temporarily on disk as soon as it is read, and later used from disk when the job is actually scheduled and run. Similarly, all printed output from a job is written temporarily to disk and printed later when the printer is free while another job is executing. Spooling means that the input and output resources of the computer are used with the maximum efficiency. OS/MFT has a spooling capability but many installations substitute a special spooling package called HASP.

1.8 Job Control Language

The instructions the user needs to give the control program in order to tell it what program is to be executed and what data is to be used, must be coded in a special language and punched onto cards to be understood by the control program. This special language is called Job Control Language (JCL). Just like any other language, JCL has rules and syntax which must be learned. In addition to learning, it should also be understood, since errors are then made less

frequently, and if they are still made, are detected much more easily.

1.9 Defintion of a Job

A job to be run on the 360 consists of a set of control cards needed to perform the required set of tasks. The execution of one program is called a job step. Besides the JCL cards needed by the control program, the program to be executed also may need control cards. The complete set of cards, JCL and control cards for the program is known as a setup.

1.10 JCL Statements for the IBM 360 OS Control Program

1.10a General Format of a JCL Statement

//name	operator	parameters	comments
(i)	// (slash,slash)		in columns 1 and 2 indicate a JCL statement and are recognized as such by the control program.
(ii)	Name		This consists of 1 to 8 alphameric characters starting with a letter. Its purpose depends on the type of operator.
(iii)	Operator		Three possibles: JOB, EXEC, DD
		JOB cards	give the name and account number of the user
		EXEC cards	give the names of the programs to be executed
		DD cards	define the type of data to be used
(iv)	Parameters		Depend on the type of operator--they are of two kinds:
		(i)	Keyword, written in the form: Keyword = value
		(ii)	Positional, written in the form: value
			Parameters are separated from each other by commas.
(v)	Comments		Any information may be given here. It is merely listed out by the control program.

There must be at least one blank between the name and the operator, the

operator and the first parameter, the last parameter and the comments and no blanks anywhere else. Blanks in fact act as delimiters between the four items: name, operator, parameters, comments.

1.10b Punching Conventions

Punching (of the //) starts in column 1 of a card and can continue up to column 71; if more room is required, a continuation card can be punched by interrupting punching at the comma after one parameter on the first card, before reaching column 71; punching // in the first 2 columns of a continuation card; leaving at least one but not more than 13 blanks and continuing punching.

1.10c Comments Cards

Special cards with /** (slash, slash, asterisk) in columns 1 to 3 may be punched. Such cards are not processed by the control program but are merely listed.

Chapter II

JOB AND EXEC STATEMENTS

2.1 JOB Card

2.1a Purpose

Every job submitted to the computer must start with a JOB card. This gives, amongst other things, the name of the user and the account number to which computer time used is to be charged.

2.1b Format

```
//jobname JOB account no.,user,optional system parameters
```

2.1c Description of Items

- | | | |
|-------|---------------------|---|
| (i) | jobname | mandatory - 1 to 8 alphanumeric characters starting with a letter |
| (ii) | JOB | operator |
| (iii) | account no. | consists of one or more accounting fields. If more than one, the values must be enclosed in parentheses, e.g., (field 1, field 2, field 3). The actual format depends on the installation's requirements. |
| (iv) | user | This is the user (or programmer's) name in up to 20 alphanumeric characters. |
| (v) | optional parameters | Various different system options are available through specification with keyword style parameters. e.g., MSGLEVEL=(m,n) specifies how much in the way of job control language and unit allocation and deallocation messages are printed. |
- m=0 all JCL (user and system supplied)
m=1 User supplied only

m=2 Job card only
 n=0 No allocation and deallocation messages
 n=1 All allocation and deallocation messages

Default values for m and n are set by the installation. At the ISR they are (1,1).

Other available parameters are CLASS, COND, MSGCLASS, PRY, RD, REGION, RESTART, ROLL, TIME, TYPRUN. See [1], [2] for further information.

2.2 Form of Job Card Used at the ISR

The JOB card for use on the 360 at the ISR has been split into two parts as follows:

```
//receipt no.   JOB   (,                               CSF supplied part
// accounting fields),user,system parameters       User supplied part
```

2.2a CSF Supplied Part of Job Card

The CSF supplied part is picked up by the computer room along with a receipt card. The jobname on the JOB card is a unique number prefixed by an M which is also on the receipt card. The receipt card is retained by the user for later retrieval of the output of the job.

2.2b User Supplied Part of Job Card

The user supplies a continuation card for the job card:

(i) // in columns 1 and 2 followed by 1 to 13 blanks

(ii) The accounting fields consist of 9 possible fields, only the first one of which is mandatory: (a,b,c,d,e,f,g,h,i),

where a = project number
 b = MTR no., 1-4 alphameric characters (a subdivision of the project)
 c = job no., 1-4 alphameric characters (within MTR)
 d = estimated CPU time (default 3 minutes)
 e = estimated number of printed lines in thousands (default=5)
 f = estimated number of punched cards (default=1000)
 g = number of copies of printed output (default=1)
 h = special forms indicator for printed output
 i = number of lines to be printed per page (default=61)

Items a-c are used to identify expenditures on different parts of a project. Billings are given by a total for a project; a subtotal for each MTR within that project; and a subtotal within each MTR for each job no.

- | | | | |
|-------|-----------------------|-----------|---|
| (iii) | user | mandatory | The user's name is up to 20 alpha-
meric characters. If special char-
acters or blanks are required, the
whole name must be enclosed in apos-
trophes, e.g.,

'JOHN B. SMITH' |
| (iv) | systems
parameters | optional | see 2.1c(v) above |

2.3 Examples of Good Job Cards for Use at the ISR

Both the receipt part and the user part (continuation) of the JOB card are shown. The receipt part is prepunched and picked up from the table by the computer room.

- (i) Project 499999, MTR 1000, job no. JBR, user Smith

```
//M012345 JOB (,
//      499999,1000,JBR),SMITH
```

- (ii) Project 7999, MTR 20, no job no., CPU estimate of 2 minutes, user Jones

```
//M111111 JOB (,
//      7999,20,,2),JONES,CLASS=B
```

- (iii) Project 498980, no MTR, job no. 2, CPU estimate of 6 minutes, 2 copies of printed output, user Brown

```
//M999999 JOB (,
//      498980,,2,6,,2),BROWN
```

- (iv) Project 40000, no MTR or job no., CPU estimate of 20 minutes, 30,000 lines of printout, user Green

```
//M987654 JOB (,
//      400000,,20,30),GREEN
```

2.4 Possible Errors on Job Cards

Fatal errors are:

```
jobname missing
no accounting fields
no user name
syntactical errors
invalid project number
```

Some examples of job cards with errors are given on handout 1. The output resulting from running these through the computer running directly under OS is given in handout 2. If the HASP spooling system is being used, JOB cards are checked as the setup cards for the job are initially read. If there is an error,

the job is never entered into the queue but rather deleted immediately with the message:

```
***** ILLEGAL JOB CARD *****
***** JOB DELETED BY HASP OR CANCELLED BY OPERATOR BEFORE EXECUTION *****
```

No indication is given in this case as to what the error was caused by.

2.5 Explanation of Handout 2

- (i) The blank after the project number means that 1236),BROWN are treated as comments and a continuation card is expected because of the comma after the project number.
- (ii) MGSLEVEL instead of MSGLEVEL
- (iii) No jobname after //
- (iv) Parenthesis before project number (499990) on continuation card is unnecessary since there is one on the first card.
- (v) No parenthesis after the project number to match the left parenthesis on the first card.
- (vi) Z class of jobs does not exist.
- (vii) Special characters are not allowed in the accounting fields.

2.6 EXEC Card

2.6a Purpose

The execute card tells the control program what problem program is to be executed. It can also be optionally used to supply parameters to the problem program. It always signals the start of a new job step. Normally, one EXEC card is needed for every program to be executed, although, it will be seen later that there are exceptions to this.

2.6b Format

```
//stepname EXEC PGM = program name } , system parameters,
                or procedure name }      symbolic parameters
```

2.6c Description of Items

- (i) Stepname Optional - If present, for user aid in identifying the step; it consists of 1 to 8 alphanumeric characters starting with a letter.

- (ii) EXEC operator
- (iii) PGM=program PGM is a keyword; program is 1 to 8 alphanumeric characters starting with a letter giving the name of the program to be executed. A program with that name must exist in a program library on disk.
- Procedure name One to 8 alphanumeric characters starting with a letter giving the name of the procedure to be executed.

Sometimes a lot of JCL cards for achieving a certain task are always the same whatever data is being used. In this case, these JCL statements can be stored in a special "procedure" library on disk and when they are required, called for by referring to the appropriate name. The first statement of a procedure is always // EXEC PGM=program name. It is possible to have more than one EXEC PGM statement in a procedure; hence it is possible for one user EXEC card to call for the execution of more than one program. Procedures are covered in detail in Chapter VIII.

- (iv) Optional system parameters Various different system options are available through the use of keyword style parameters. These include ACCT, COND, DPRTY, PARM, RD, REGION, ROLL, TIME. See [1], [2] for further information.

The COND parameter can be used to test various types of outcome of a previous step in order to determine whether the current step is to be executed or not. One kind of outcome of a previous step is whether the step ended normally or abnormally.

Abnormal termination means that the control program terminated a job for some reason. To the user it looks like handout 3b. The line immediately after all the 'allocated to' messages gives a 'completion code' defined as a system completion code or a user completion code. In the example there is a system condition completion code of 813. Some of the more commonly encountered system completion codes are given in Chapter IX.

When the COND parameter is not used, once a job step has been abnormally terminated, no other step in that job will be executed.

COND=EVEN punched on an EXEC card means that the step is to be executed even if previous ones have failed.

COND=ONLY punched on an EXEC card means that the step will be executed only if a previous

step has failed.

Suppose a job consists of 4 steps with the following EXEC cards:

```
//STEP1 EXEC PGM=A
//STEP2 EXEC PGM=B,COND=EVEN
//STEP3 EXEC PGM=C,COND=ONLY
//STEP4 EXEC PGM=D
```

If STEP 1 terminates abnormally, then STEP 2 and STEP 3 will be executed but STEP 4 will be bypassed.

If STEP 1 terminates normally, then STEP 2 and STEP 4 will be executed but STEP 3 will be bypassed.

The PARM parameter can be used by programs to pass values directly to the program rather than reading them from control cards. For example the IBM 360 Sort/Merge program allows one to pass values in this way, e.g.,

```
// EXEC SORT,PARM='MSG=AP'
```

instructs the sort program to write its messages on the printer.

- (v) **Symbolic parameters** The values for symbolic parameters on DD cards are also given in certain circumstances on the EXEC card. This feature is connected with procedures and is discussed in Chapter VIII.

2.7 Examples of EXEC Cards

- (i) To execute a program called ATEST

```
//ST1234 EXEC PGM=ATEST
```

- (ii) To execute a program called COPY passing it a parameter of NOLIST

```
// EXEC PGM=COPY,PARM='NOLIST'
```

- (iii) To execute a procedure SORT even if a previous step has terminated abnormally

```
//AB EXEC SORT,COND=EVEN
```

- (iv) To execute a procedure called SCRAT only if a previous step has terminated abnormally, and pass it a parameter of MAP

```
// EXEC SCRAT,COND=ONLY,PARM='MAP'
```

2.8 Possible Errors in the EXEC Card

Step name too long or not starting with a letter
 Program name specified not in the program library
 Procedure name specified not in the procedure library
 Keyword parameter spelled incorrectly
 Blanks in wrong places

2.9 Examples of Errors on EXEC Card

See handout 4 for examples of EXEC cards. Handout 5 gives the messages that the control program writes on encountering the errors.

- (1) - (4) OK
- (5) Double apostrophe in the PARM field
- (6) PARM misspelled
- (7) Stepname beginning with a number
- (8) Stepname of 9 characters
- (9) PGM misspelled
- (10) Program name beginning with a number
- (11) PROGRAM is an invalid keyword. Should be PGM
- (12) Procedure name has 9 characters
- (13) Invalid value (EVEM instead of EVEN) for the COND parameter
- (14) Blank after IEBGENER, means that COND=EVEN is treated as a comment and a continuation card is expected
- (15) Blank after IEFBR14 means that COND=ONLY is treated as a comment and ignored. Program would be executed
- (16) The procedure named CARDSORT is not in the library of procedures
- (17) COND misspelled
- (18) OK - An example of a value being given to a symbolic parameter 'TAPE'

Chapter III

INTRODUCTION TO DD STATEMENTS AND DATA FORMATS

3.1 DD Cards - Introduction

3.1a Purpose

DD stands for data definition. There must be a DD card to describe each data file being used in any job step. Programs can be written to input and output data without knowing on what medium the data resides. When the program is executed, this information is supplied on the relevant DD card, e.g., whether data to be read is from cards, tape or disk.

3.1b General Format

```
//ddname DD parameters
```

3.1c Description of Items

- (i) ddname 1-8 alphameric characters as given in the program write up. The ddname is the connection between the problem program and the data it is to use. That is, the ddname is what is referred to in the problem program. When the program is executed, it finds the particular data to be used by looking at the parameters on the DD card for the appropriate ddname.
- (ii) DD Operator standing for Data Definition
- (iii) Parameters. These all take the form:

Keyword=One or more values

If more than one value is to be specified, they are enclosed in parentheses and separated by commas.

The values themselves can have a keyword format or be positional, e.g.,

DISP = (OLD,PASS) (OLD & PASS are positional)
 DCB = (BLKSIZE=800) (BLKSIZE is a keyword)

3.1d DD Parameter Keywords

The DD parameter keywords to be covered in this text are given below with a brief description of the kind of information they give about the data.

DCB	to specify data characteristics
DISP	to indicate if data already exists or is being newly created
DSNAME	gives a name to the data file
LABEL	indicates which file on a tape is to be used; kind of labels the tape has
SPACE	used to allocate space for a data set being created on disk
UNIT	indicates the device type (e.g., tape or disk)
VOLUME	gives the tape or disk serial number on which the data resides

These parameters are mainly used for data on tape or disk. The card reader, card punch and printer have a different method of specification (see Chapter VII).

3.1e Understanding DD Parameters

A full comprehension of DD parameters presupposes a knowledge of tape and disk characteristics and of data types and formats. A full description of the DD parameters will therefore be left until these topics have been discussed.

3.2 Data

Suppose certain information has been gathered about people, e.g., sex, age, income, education, together with an ID for each person.

The above items are called variables--the values they take vary from person to person.

Whatever medium is used to record the data--paper, card or magnetic tape--it is necessary to map out the positions these different items occupy.

e.g.,

Sex	Age	Income	Ed.	I.D.
.
.
.

The values for one person occupy one line on this chart. The five items occupy five fields, each field containing a fixed number of characters known as the field width. The field width for a variable must be equal or greater than the maximum number of digits a variable can have, e.g., if it is possible to have ages greater than 99, even if rare, then the field width for age must be 3.

3.3 Records

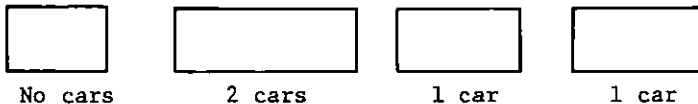
For each person we now have a record made up of five fields. The record length is the sum of the number of characters in the five fields. The records for all the people make up a file of data (known also as a dataset).

If records were on tape or disk, they can be thought of as looking something like this:



These records all containing the same variables, have the same length. They are called fixed length records.

If one were to add to the 5 variables, a 6th variable which was the number of cars owned and then succeeding variables giving the make, year and purchase price of each car, then there would be a variable amount of information collected for each person according to the number of cars he owned. One could then either allow for up to 4 cars for each person and pad out the inapplicable variables with some special code, thus keeping all records the same length; or one could theoretically have variable length records according to the number of cars owned:



3.4 Blocks

The record discussed above is sometimes known as a logical record; it is the logical unit of analysis and is the amount processed by a program at one time. Tape and disk drives actually read data in a unit of a block or physical

record at a time. One block can contain one or more logical records.

Between each block on a tape there is an interblock (interrecord) gap of up to 3/4 inch of tape. The drives recognize the end of a block by this gap.

When there is more than one record per block, the data are referred to as blocked.

3.5 Advantages of Blocking

There are two advantages:

- (i) To get more information on a tape--i.e., the fewer interblock gaps there are, the better the utilization of the tape, e.g., if 80 character records (card images) are written to a 2400 feet tape at a density of 800 bpi, then the tape can hold approximately 34,000 unblocked records or alternatively 169,000 records if they are written 10 to a block.
- (ii) The initiation of physical reading and writing of data is a relatively slow operation by computer standards. The fewer times, therefore, this is done, the better. Since data are read a block at a time, the larger the block, the fewer the initiations of reading/writing.

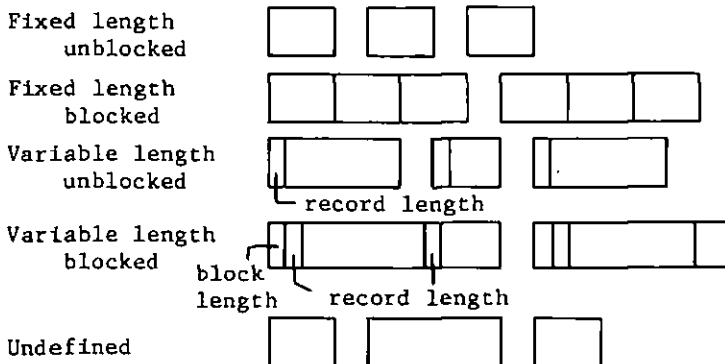
3.6 Limitations on Blocksize

The limitation on the block size is imposed because of core limitations. Since data is read into the computer's memory (or core) a block at a time, there must be an area in the memory set aside at least as big as the block size. If a very large block is used, then there is no room left for program instructions.

There is another limitation when data is on disk which is the capacity of one track (since unless a track overflow feature is available, one block cannot occupy more than one track).

The combination of the above two items is the reason why the usual maximum block size presently being used in the OSIRIS programs at the ISR is 3520 characters.

3.7 Summary of Record Types



NOTE: Variable length records contain a count of the number of characters in the record in the first two characters of the record. If, in addition, the variable length records are blocked, there is also a count of the number of characters in the entire block in the first two characters of the block.

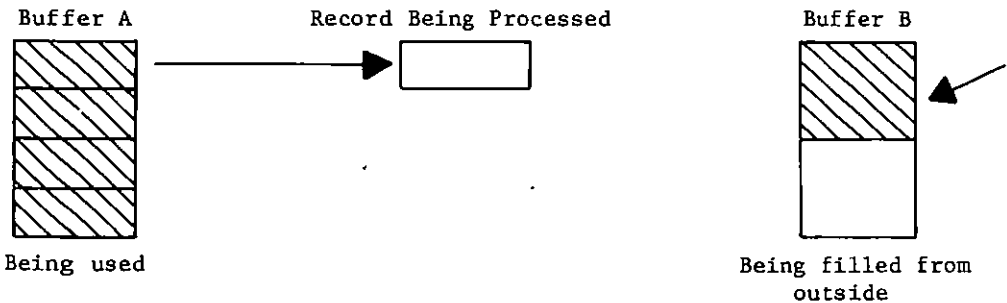
Undefined records are unblocked, variable length records with no count in the first two characters. They cannot be blocked.

3.8 Block Length

The block length or block size is the number of characters in the block. With fixed length records, this will always be a multiple of the record length, and will be fixed except for a possible short block at the end of the file. With variable length records, the block length will be variable and the block will contain a variable number of records such that the total length does not exceed some prespecified maximum block size.

3.9 Buffers

Blocks of data are actually read into a special area of memory called a buffer. From the buffer, records are taken one by one for processing until all records have been used at which point another block is read into the buffer. The actual transfer of data from an input medium into a buffer can take place concurrently with computations by the central processing unit (CPU). However, one cannot be using records from a given buffer until the transfer is completed. Therefore, as a time saving device, two buffers are often used for one data file. Records are used from buffer A while buffer B is being filled. When all records from buffer A have been processed, then they are taken from other buffer B and a new block is read into buffer A and so on. With this method of operation, known as double buffering, the CPU is not kept waiting for data.



If two buffers are assigned to a data file, then two areas of core, each at least as big as the blocksize of the data, have to be set aside. Since the core size is limited, with large programs it is sometimes impossible to spare this amount of core space, so in this case, time savings have to be waived and one buffer used.

Chapter IV

TAPES

4.1 Physical Characteristics

4.1a Tracks

Data are recorded magnetically on tracks on tape. The tracks run vertically down the length of the tape and one character is recorded by magnetising a certain combination of the tracks in a horizontal direction. Tape drives normally write on nine tracks on tape although the ISR also has one tape drive which writes with seven tracks. Note that the tape is originally the same and it is the tape drive which makes it into seven or nine track. However once written, it can only be read by the appropriate tape drive.

4.1b Parity

With 9-tracks, 8 are used for recording characters and the 9th is used as a parity check. This is magnetised or not so as to make the total number of magnetised tracks in one character for every character on a tape an odd or even number. Data can thus be recorded at either odd or even parity.

With 7-tracks, 6 tracks are used for recording characters and the 7th is the parity track. The default at the ISR is odd parity for all tapes.

4.1c Density

Density on a tape is defined as the number of characters written on one inch of tape. It is measured in bpi (bits or bytes per inch).

Different types of tape drives are capable of recording data at different densities. Some tape drives use only one specific density while others have a dual or triple capacity. Once data are recorded at a particular density, they can only be read from a tape drive capable of using that density. Perhaps the most commonly used density is 800 bpi. However, densities of 200, 556 and 1600 are also used and newer tape drives are capable of reading and writing at much higher densities. Normally, installations have a default density (at the ISR, this is 800 bpi on all the tape drives). Thus, for tapes being read and written exclusively at one installation, the user does not need to be concerned about speci-

fyng densities.

4.1d Interblock Gaps and Tape Capacity

Tapes are usually either 1200 or 2400 feet long. However, the number of characters that can be stored on a tape at a given density depends on the physical block size. This is because between each block there is an interblock gap (by which the tape drive recognizes the end of a block while reading) of nearly 3/4 inch.

Thus, if cards are written to tape unblocked, at a density of 800 bpi, each card (80 characters) will take .1 inch for the data together with .75 inches for the interblock gap, i.e., 100 cards will take approximately 85 inches. If the cards were put 10 to a block, 10 cards would take 1 inch and .75 inches for the interblock gap. Now the 100 cards will only take up about 17 inches of tape.

4.1e Tape Marks

At the end of every file on a tape there is a special one character block called a tape mark. The tape drives recognize the tape mark as an end of file signal. In addition, at the end of the last file on a tape, there are two tape marks.

4.1f Write Rings

Tape drives will not permit writing on a tape unless a special write ring is inserted in the tape before mounting on the tape drive. This affords some protection against overwriting data by mistake. At most computer installations operators mount tapes on tape drives without this ring unless the user has specifically requested 'ring in'.

4.1g Speed

Access rates of data from tape vary from 7,500 characters of data per second to 320,000 characters per second, according to the type of tape drive.

4.2 Tape Labels

4.2a Unlabelled Tapes

Unlabelled tapes contain just data and tapemarks. If there is more than one file on the tape, each file is separated by a tape mark. There are two tape marks at the end of the last file on the tape.



4.2b Labelled Tapes

IBM/360 standard labelled tapes have the structure shown on handout 6a.

4.2c Why Use Labels?

(i) Safety: reading - The operating system will check that the correct tape is mounted and the correct file is being read.

writing - The operating system will check that the correct tape is mounted; also, if an expiration date has been used it will help protect files from being overwritten by mistake.

(ii) Convenience Data characteristics such as record format and length are stored in the label and do not need to be provided by the user when the data is read. Labels make the tape partially self described. If the contents of the tape are forgotten, it is possible to list the labels of the tape. If there were no labels, all the files would look much the same.

4.2d When Should Unlabelled Tapes Be Used?

The only two circumstances necessitating unlabelled tapes are:

- (i) A tape is received from an outside source and arrives without labels.
- (ii) A tape of data is to be sent to a non-360 computer installation where the labels would not be compatible.

4.2e Format of Labels

See handout 6b.

The first 80 character block is a volume label. This must be written on the tape once and for all before it is first used for recording data. After this, data files may be written and each will have a header label and a trailer (end of file) label in addition to the data.

4.2f Writing a Volume Label

A volume label may be written to a new tape, or an existing tape's volume label changed with the IBM 360 utility program IEHINITT. (See [3]).

4.2g Checks Made by the Operating System

- (i) When a tape is mounted after a request to the operator has been issued by the operating system, the tape number in the

volume label is checked against the tape number specified by the user and if not equal OS will ask the operator to mount the correct tape.

- (ii) If the tape number is correct, OS then checks that the file name in the appropriate header label is the same as the user requested. If not, OS issues an 813 system completion and terminates the job.
- (iii) If a file is being written over another file, OS checks that the expiration date of the file being overwritten has been reached. If not, OS issues a message to the operator asking whether it is OK to go ahead and write or not (the operator then replies 'U' if he has been instructed to override the expiration date by the user, otherwise, he cancels the job).

4.2h Listing Tape Labels

Tape labels may be listed using an installation utility program. These programs usually treat every tape mark as indicating the end of one file. Therefore, the header label for the first data file is file 1, the first data file is file 2, and the trailer label for the first data file is file 3 and so on. At the ISR, procedures LABPRT and LABELDOC are available for this purpose.

4.2i Multifile Tapes

As a general rule, it is better not to keep too many files on one tape unless the files are stable and not used too often. Reasons:

- (i) Searching for files at the end of a tape takes computer time (it takes about 5 minutes just to search through a complete tape).
- (ii) Once a file has been overwritten, no other files previously written after this one can be used, e.g.,

Old File 1 TM File 2 TM File 3 TM File 4 TMTM

If file 2 were overwritten, one might get something like this:

New File 1 TM File 2 TMTM File 3 TM File 4 TMTM

File 3 and File 4 are now inaccessible since the two tape marks at the end of File 2 indicate the end of all files (there is a last resort way of getting at File 4 if the worst happens). Therefore, if you have 25 files on a tape and the first 10 are of no use, it is not possible to reuse this part of the tape without first copying the other 15 files to a different tape.

4.2j Overwriting Files With Expiration Dates

In the example in 4.2i above, if File 2 is being overwritten, this is the only file whose expiration date will be checked. If part of File 3 happens to be overwritten in the process, any expiration date on File 3 will be totally ignored.

4.2k Runaway Tapes

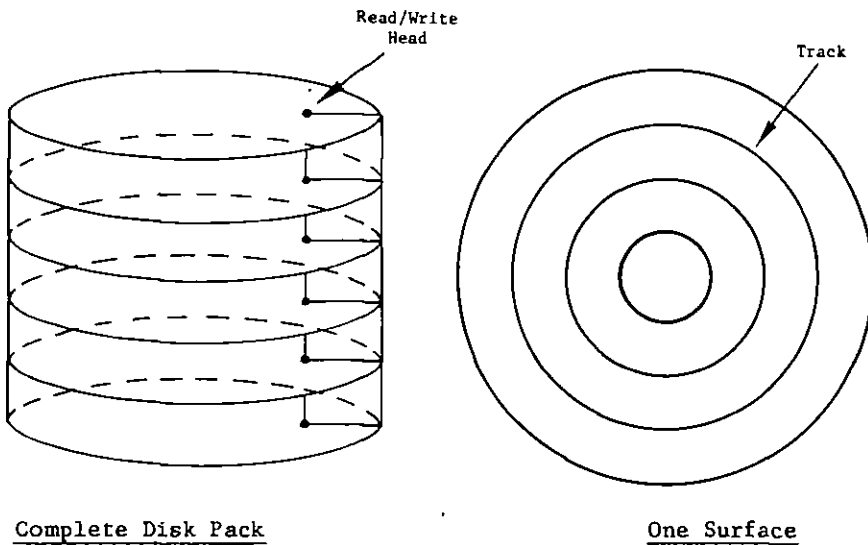
If a file is requested beyond the last file on the tape (i.e., the first double tape mark encountered) then the tape starts 'running away' until it reaches the physical end of the tape. This error normally manifests itself with a 613 system completion code.

Chapter V

DISKS

5.1 Physical Characteristics

More information can be found in [4].



5.1a Surfaces and Tracks

Characters are recorded magnetically on tracks on disk. The tracks are concentric circles on the available surfaces of the disk. Some types of disk are:

- 2314 On a 2314 disk pack there are 20 surfaces. Twenty read/write heads are fixed to horizontal arms connected by a vertical bar which can move backwards and forwards in a straight line across the surfaces while the surfaces revolve continuously under the heads. Thus for any one position of the 20 heads, 20 tracks of information can be

accessed. There are 200 different positions of the heads corresponding to 200 tracks on each surface.

2311 The 2311 disk pack is constructed in a similar fashion to the 3214 but there are only 10 surfaces instead of 20.

3330 This relatively new type of disk pack has 19 surfaces.

5.1b Cylinders

This is the unit of disk space accessible without moving the read/write heads. It consists of one track from each surface, e.g., 20 tracks on a 2314.

5.1c Capacity of One Disk Pack

- (i) 2314 has 200 cylinders or 4000 tracks per disk pack. Each track can hold a maximum of 7295 characters. One block of data may not normally overflow onto more than one track so that the physical blocksize of data on disk may not exceed 7295 characters. Between physical blocks on a disk track there are inter-block gaps taking up the space of about 100 characters. Therefore two 3600 character blocks of data would take over 7294 characters worth of space and will not fit on one track. If cards are written to disk, only 45 will fit on one track if they are unblocked. If they are blocked at 3520, however, 88 will fit on one track.
- (ii) 2311 has 200 cylinders or 2000 tracks per disk pack. The maximum capacity of one track is 3625 characters.
- (iii) 3330 has 404 cylinders or 7676 tracks per disk pack. Each track has a maximum capacity of 13030 characters.

NOTE: The capacity of one track is the same whether it is near the center of the surface or at the outside, the characters are merely packed more densely by the hardware near the center.

5.2 Volume Table of Contents

5.2a Function

The first 10 tracks or so of a disk are set aside to contain the volume table of contents (VTOC). The VTOC contains the header labels of all data sets residing on a disk (containing much the same information as a tape header label) together with the starting track number of the data set. In addition, a record of all tracks that have not been allocated to data sets (free space) is kept.

5.2b Listing the VTOC

There are a variety of different programs available for listing the contents of a disk (i.e., listing the VTOC). IEHLIST is an IBM OS utility program and is described in [3]. Different installations usually have other programs which format the information in different ways (e.g., SPVTOC, MAPDISK, SLIST).

5.2c Creating the VTOC and DASDI

When a new disk is acquired it must be 'DASDI-ed' before it can be used.

IEHDASDR, an IBM utility program which labels the disk, allocates specified tracks to contain the VTOC and checks the complete disk for bad tracks. See [3].

5.2d Examples of VTOC Listing

Handout 7 shows a VTOC listing a) using program IEHLIST, and b) using the program SPVTOC.

Looking at a):

Top line gives the disk serial number--ISR001. For each data set on the disk whose name appears in the leftmost column we have:

- | | | |
|-------|--------------------|--|
| (i) | Creation date | Date which the data set was written. |
| (ii) | Purge date | Will equal the creation date if no expiration date was defined; else the expiration date. |
| (iii) | File type | Data set organization. Here, partitioned or sequential. Not defined usually means that space was allocated but the job blew before anything was written. |
| (iv) | Extents | The number of different extents the data set is occupying. If all the original space request was allocated on contiguous tracks and the secondary allocation was not used the number of extents will be 1. |
| (v) | File serial number | The volume serial number of the disk. |
| (vi) | Volume Sequence | Will always be 1 unless a data set takes up more than one disk. |
| (vii) | Security | It is possible to give data sets a password so that they cannot be accessed without specifying the correct password. In this case security would be YES. |

Looking at b), more information is given.

Firstly, we have (i) data set name, (ii) disk serial number, (iii) volume sequence, (iv) creation date, (v) expiration dates; then

- | | | |
|-------|-------|---|
| (vi) | DSO | Data Set Organization
PO partitioned
PS sequential
IS indexed sequential |
| (vii) | RECFM | Record Format |

(viii)	BLKSZ	Blocksize
(ix)	LRECL	Record length
(x)	KEY	Applicable to indexed sequential only
(xi)	DP	Applicable to indexed sequential only
(xii)	TRKAL	The number of tracks allocated to the data set
(xiii)	TRKUS	The number of tracks actually being used by the data set
(xiv)	EX	The number of extents being used as for (iv) above
(xv)	SECQU	Secondary quantity. The number of tracks that were specified in the secondary allocation when space was originally allocated.
(xvi)	T	How space was originally allocated: T - tracks A - cylinder B - in blocks

5.3 Writing Data Sets to Disk

When a data set is to be written to disk, an estimate of the amount of space it will need has to be given by the user to the control program. The user allocates space by specifying the number of cylinders, tracks, or blocks of a given size that will be needed and the control program then looks in the VTOC records of free space to see if that space exists.

5.4 Deleting Data Sets From Disk

When a data set is deleted from a disk, what is actually done is to remove the reference to that data set from the VTOC. At the same time the record of free space is updated. Thus next time this disk is used, the space occupied by the deleted data set is considered free space, and will be written onto.

5.5 Data Set Organization

Data can be organized in various different ways on disk:

5.5a. Sequential Organization

If records of a file are written one after another consecutively and retrieved in the same way, the organization of the data set is said to be sequential. Tape files are always written sequentially and any program which is capable of reading a file from either a tape or disk will require the data to be

sequential whether on tape or disk.

5.5b Random

Data can also be written to or read from disk 'randomly.' Each record has a key of some kind assigned to it. When the file is written, the system keeps track of where the records are put by keeping an index of the keys. The records, when read, can be retrieved nonsequentially by referring to the key of the required record; the computer uses the key to calculate or find the track number of the record.

5.5c Partitioned

Another kind of organization is called partitioned. This type of data set can be thought of as a collection of small sequential data sets called members all belonging to the same family or library. The name of the data set (library name) is in the VTOC and points to the beginning of the data set. At the beginning of the data set there is a directory which in turn points to each of the members.

Program libraries are always partitioned data sets, e.g., OSIRPGM (the OSIRIS program library) is the name of a data set whose members are individual programs (e.g., DSLIST, TCOR), in the program library. Having all the programs in one data set makes updating and program retrieval much simpler.

5.6 Disk Characteristics

Disks can be 'public' or 'private.' Public disks are ones which can be used by anyone and are normally mounted on the system. Private disks are those mounted especially for one job.

Some disks, in particular those which hold system program libraries, are 'permanently resident' and can never be removed from the system. Others are removable and can be replaced by user packs. At all installations there will be a limit to the number of user packs that can be mounted at once according to the number of removable disks on the system.

5.7 The Catalog

The catalog is a special data set permanently residing on disk.

If a data set is catalogued, the following information about it is stored in the catalog:

- Data set name
- Device type on which data set is written
- Volume serial number
- File position (if device type is tape)

This information can be retrieved by the control program by referring to the data set name; i.e., if a user only provides a data set name for a file, the control program will provide the device type, (tape/disk) volume number, and file position (if device type is tape).

An example of a listing of part of catalog is shown in handout 8.

1. The data set names are given in the left most column.
2. What is being catalogued - here always a data set (DS).
3. Volume serial number on which the data set resides.
4. File number on the volume. For a disk data set this will always be zero, e.g., CSFRDATA. For a tape data set, it will be a positive number, e.g., CHINA is on File 2 of tape S031.
5. Device type - 30002001 means 2311 disk
 30008001 means tape
 30002008 means 2314 disk

Chapter VI

DD STATEMENT PARAMETERS

6.1 General DD Card Format

DD Cards have the general form:

```
//ddname DD parameters
```

6.1a ddname

A DD card is needed by the control program for every file that is being read or written by a particular problem program. Printed output, card input or punched output each count as a file for this purpose as well as data files being read from or written to tape or disk.

In a problem program, the instructions for reading and writing refer to a particular ddname, e.g., a hypothetical instruction in a program might be:

```
READ DICTIN
```

which would read one record from a file defined by the ddname DICTIN.

When the program is executed, the characteristics of the file for a particular run are defined on the DD card with the appropriate ddname. Thus for a particular program, the ddnames are constant; it is only the parameters given on the DD cards which may change from run to run if different files are being used.

6.1b DD

DD stands for Data Definition.

6.1c Parameters

DD parameters describe the characteristics of a particular data file and take the form:

```
keyword = (value 1, value 2.....) ,
```


e.g., DISP=(NEW,PASS)

or keyword = (keyword1=value1, keyword2=value2) ,

e.g., DCB=(RECFM=F,BLKSIZE=3000) ,

or a combination of the two is

VOL=(PRIVATE,RETAIN,SER=1000)

The following keywords will be discussed:

DCB, DISP, DSN, LABEL, SPACE, UNIT, VOL

Some of the values have defaults, that is, if they are not given on a DD card, the system supplies certain values. The order in which the keywords are given does not matter.

Files are usually called data sets in IBM 360 terminology.

6.2 DISP Parameters

Examples: DISP=(NEW,CATLG)
DISP=(OLD,PASS)

DISP stands for disposition and tells the control program whether a data set exist: (i) before it was used (ii) after it has been used by a program (iii) after it has been used by a program which terminated abnormally.

General form:

DISP=(before,after,abend)

'Before' can take one of the values:

NEW	meaning the data set does not exist and is going to be created
OLD	Data set already exists
SHR	Data set already exists and several jobs may access it at once
MOD	Data set already exists but more data is to be added to the end of it

'After' can take one of the values:

KEEP	Meaning the data set is to be kept at the end of the job step. On a tape this implies rewinding and unloading of the tape.
PASS	The data set is to be passed to a subsequent job step in the same job. On a tape, this means that the tape will not rewind at the end of this job step. If a subsequent job step uses data from the same tape, the computer will search backwards or forwards (as appropriate) for the file re-

quested with no intervention from the operator.

- DELETE The data set is to be deleted at the end of the job step. On a tape, this has the same effect as KEEP, i.e., the tape is rewound and unloaded.
On a disk, the pointer to the data set in the VTOC is removed and the file will no longer be accessible.
DELETE also deletes a catalog entry for the data set if there is one.
- CATLG An entry for the data set is to be made in the catalog at the end of the job step. CATLG implies KEEP.
- UNCATLG An entry for the data set in the catalog is to be removed. UNCATLG implies KEEP.

'After abend' can take the same values as described for 'after' above. It is used at the end of the job step if the program terminates abnormally (i.e., a system or user completion code is given), instead of the 'after' value. If it is not supplied, which is the normal case, the value given for 'after' is used at the end of the job step whether there was a normal or abnormal termination.

Defaults

- (i) If no DISP parameter is given on a DD card, the control program supplies: (NEW,DELETE). These are the values used for a temporary data set which is created and used during one job step but not needed at the end.
- (ii) If only a value for 'before' or 'after' is given:
- DISP=OLD - a value of KEEP is supplied for 'after'
- DISP=NEW - a value of DELETE is supplied for 'after'
- i.e., 'after' defaults to the condition before the job step was started
- DISP=(,after) - the 'before' value always defaults to NEW
- (iii) What happens to 'passed' data sets at the end of a job.

An 'after' disposition of PASS means pass this data set to a subsequent job step. If this disposition is given on a DD card in the last step of a job, or if the data set is never actually referred to in a subsequent step, then the control program has to take some action at the end of the job. It works the same way as (ii) above, i.e., a 'passed' data set is kept at the end of the job if the original disposition was OLD and deleted if the original disposition was NEW.

6.3 DSNAME or DSN Parameter

Examples	DSNAME=STUDY1
	DSN=X499999.DATA26
	DSN=&A

The DSNAME (or shortened DSN) parameter is used to define the name of the data set.

Rules

It can consist of up to 44 alphanumeric characters starting with a letter. If more than 8 characters are required, then each group of 8 (or less) must be separated by a period and the first character after a period must be a letter.

On a tape label, there is only room for 17 characters altogether. If more than 17 characters are specified on a DD card which is defining a new data set, only the last 17 will be written into the header label.

Data Set Names With Periods

Data set names containing periods cannot be catalogued unless an 'index' has previously been built by a programmer.

Disk Data Set Names at the ISR

Disk data sets being kept on ISR disks must have data set names starting with an X and followed by a project number. The project is charged by the day for the amount of disk space used, e.g.,

DSN=X499999D

Temporary Data Sets

Temporary data sets are those which are used only within a job and not needed afterwards. A temporary data set can be defined by specifying a name with an ampersand as the first character, e.g.,

DSN=&A

When such a name is given, the system translates it into the form:

SYS69349.TO84006.RP001.E016708.A

Such a data set cannot be kept; if a disposition (NEW,KEEP) is given, the system will substitute PASS for KEEP.

Default

If no DSN parameter is given on a DD card, the system assigns a temporary one of the form:

SYS69349.TO84006.RP001.E016708.R000D016

Label Checking on Tape by the Control Program

When a data set name is given on a DD card for an OLD data set on a

standard labelled tape, the name on the DD card is checked against the name in the header label of the data set. If the two are not the same, a system completion code 813 results.

The data set name for a NEW data set is written into the header label for the data set. If a temporary data set name was given, then the last 17 characters of something like the long names described above would be written in the label.

VTOC Checking on Disk

When a data set name is given on a DD card for an old data set on disk, the volume table of contents of the disk is searched for that name. If the name is not found, a system completion code 213 results.

When a data set name is given on a DD card for a new data set to be created on disk, the VTOC is searched to see if that name already exists. If it does, a message:

'DUPLICATE NAME ON VOLUME'

is given and the run terminated.

6.4 Possible Errors in DISP and DSN Parameters

Handout 9 shows some DD cards with only DSN and DISP parameters defined.

```
//A      OK

//B      No parentheses round the two DISP values. Parentheses
         may only be omitted if only the first value is being
         given, e.g., DISP=OLD.

//C      OK, providing data set is catalogued. Parentheses round
         OLD are unnecessary but OK.

//D      DSNAME value must start with a letter.

//E      CATALOG is an invalid value for the second DISP field.
         It should be spelled CATLG.

//F      DSNAME value can only have up to 8 characters between periods.

//G      DSNAME cannot contain special characters--here a colon
         (: ) unless the whole name is enclosed by apostrophes.

//H      The keyword DISS is not recognized by the control program.
         Should have been DISP.
```

6.5 LABEL Parameter

```
e.g., LABEL = (3,NL)
        LABEL = EXPDT=71300
        LABEL = (2,SL,RETPD=10)
```

6.5a Tape

On a tape the LABEL parameters can give the file number on the tape, the type of label and, if a new data set is being created, an expiration date or retention period.

```
LABEL=(file number, label type, retention period)
```

```
file number    1 or 2 digit number
```

```
label type     SL - standard label
               NL - no labels
               BLP - Bypass label processing
```

Retention period Either EXPDT=yyddd where yy and ddd give the year and day until which the data is to be protected.

 or RETPD=ddd where ddd gives the number of days for which the data is to be protected.

It is recommended that whenever possible, standard labels should always be used. This means that a volume label must be written on the tape (IBM utility IEHINITT or procedure TAPLAB at the ISR) and thereafter SL is always used in the LABEL parameter.

Checks made by control program are:

With an NL or SL label type, the system reads the first block on the tape.

If NL was specified and the tape has a volume label, an error condition results.

If SL was specified and the tape has no volume label, an error condition results.

With BLP, the system does not read the first block on the tape and absolutely no checks are made. BLP should be used therefore with great caution and never when writing unless it is necessary for some reason to write an unlabelled tape on a tape that had had standard labels.

6.5b AVR and Label Types

AVR stands for Automatic Volume Recognition. With this feature, tape volumes can be mounted before the job which needs them is executed. The control program reads the label of any tape premounted in this way and stores the volume number in the memory. When a DD card is read specifying a tape volume, the control program checks to see if the volume is already mounted by comparing the number with those in its memory. If the volume is not already mounted, a mount request message is then issued to the operator; otherwise the volume is used directly.

Tapes which are premounted but which do not have standard labels are rejected.

BLP is sometimes specified on a DD card for a standard labelled tape in order that the label can be treated as a data file and printed. If such a tape is premounted, the AVR feature will mean that the label is read and stored prior to the start of the job. It is then not accessible as data to a listing program. In such circumstances, tapes should not be premounted.

6.5c File Numbers and Label Types

The file number specified on the DD card gives the data file number. The control program actually recognizes a tape mark as being the end of file. However, if a label type SL is specified, then the header, data, and trailer are treated as one file even though there are actually three tape marks involved. If NL or BLP is specified, then each file is separated by one tape mark.

i.e., to access the second file of a standard labelled tape one

would normally specify:

LABEL=(2,SL)

But one could also say

LABEL=(5,BLP)

6.5d Disk

There is no such thing as a file number on disk and all data sets have data set header labels (the equivalent to standard tape labels). When reading a disk data set, therefore, no LABEL parameter is given.

When creating a new disk data set, the LABEL parameter is used only to set an expiration date or retention period, e.g.,

LABEL=EXPDT=70365 or LABEL=RETPD=90

With this expiration date specified, the data set will not be scratched or overwritten without special operator action until the 365th day of 1970, or, in the second example, until 90 days have elapsed.

6.6 UNIT Parameter

Takes the form UNIT = device name, e.g.,

UNIT=DISK
UNIT=TAPE

Each input or output device has a unique 3 hexadecimal digit unit number, e.g., 280. Each type of I/O device has a 4 digit generic name, e.g., 2400.

In addition, when an operating system is 'generated', other generic names can be added to the system but they will vary from installation to installation.

At the ISR, the following devices are currently available:

<u>Device Type</u>	<u>Unit Code</u>	<u>Device Code</u>	<u>Other Names</u>
Card reader	00A	2501	-
Printer	00E	1403	-
Card punch	015	2520	-
Disk	130	3101	DISK, D, PD
	131		DISK, D, PD
	132		SYSDA, DISK, D
	133		SYSDA, SYSWRK
	134		SYSDA, SYSWRK
	135		SYSDA, SYSWRK
7-track tape	280	2400,2403	TAPE7, T7

9-track tape	281	2400,2401	TAPE, T, T800
(800 bpi)	282	2400,2402	TAPE, T, T800
	283	2400,2402	TAPE, T, T800
9-track tape	290	3420	T1600, T16
(800/1600 bpi)	291	3420	T1600, T16

When a UNIT value is being specified on a DD card for a disk or tape, any of the above names can be used. The card reader, punch and printer must be specified by the user in a different way which will be covered later.

If the unit code is used, e.g.,

```
UNIT=281
```

then that actual unit is assigned.

If a generic name is used, e.g.,

```
UNIT=TAPE
```

then the control program chooses a suitable device from the available ones with that name. Names in fact should always be used rather than specific units since then it does not matter if one of the units is not functioning at a particular time.

More Volumes Than Devices - AFF Parameter

If more volumes are needed in a job step than there are devices to mount them on, but if not all volumes are required at the same time, then a request can be made to have more than one volume allocated to the same device as shown in the example below which requires five tapes.

```
// EXEC PGM=X
//DDA DD UNIT=TAPE,DSN=DAT1,VOL=SER=90,DISP=OLD
//DDB DD UNIT=TAPE,DSN=DAT2,VOL=SER=91,DISP=OLD
//DDC DD UNIT=TAPE,DSN=DAT9,VOL=SER=92,DISP=(NEW,KEEP)
//DDE DD UNIT=AFF=DDA,DSN=DAT3,VOL=SER=93,DISP=OLD
//DDF DD UNIT=AFF=DDB,DSN=DAT4,VOL=SER=94,DISP=OLD
```

The two files defined by DD cards with ddnames DDE and DDF have been assigned the same unit as the files defined by ddnames DDA and DDB by making use of the AFF (standing for affinity) parameter: UNIT=AFF=DDA says unit is the same as used for file defined by ddname DDA.

6.7 VOLUME or VOL Parameter

6.7a Function

This parameter specifies the tape/disk serial number(s) for a data set and also gives information on the type of volume for a disk, e.g.,


```
VOL=SER=1000
VOL=(PRIVATE,RETAIN,SER=D30)
VOL=SER=(1000,1001,1002)
```

6.7b Tape

The SER subparameter gives the volume serial number of the tape. If a data set resides on more than one tape, then all serial numbers are given separated by commas and inside parentheses.

6.7c Disk

If the disk is a public disk normally mounted on the system, then merely the serial number need be specified as for tape, e.g.,

```
VOL=SER=ISRA
```

If a private disk is being used, then two extra positional parameters should be specified in order to ensure that the disk is released from the system at the end of the job:

PRIVATE means that the disk is dismounted at the end of the job step

PRIVATE, RETAIN means that the disk is dismounted at the end of the job

e.g., VOL=(PRIVATE,RETAIN,SER=300)

Note that the positional parameters PRIVATE and RETAIN come before the keyword SER.

6.7d Catalogued Data Sets

If a data set is catalogued, the serial number does not have to be specified on the DD card. However, if it is catalogued and on a private disk, the private parameter must still be given, e.g.,

```
VOL=(PRIVATE,RETAIN)
```

6.7e If No Serial Number is Specified

Output (i.e., DISP=NEW)

Disk: System assigns any available disk which is currently mounted.

Tape: A request to mount a scratch tape is given to the operators who will then usually cancel the job.

Input (i.e., DISP=OLD)

Disk or tape: If the dataset is not catalogued a message 'LEVEL OF INDEX MISSING'

will be given when the system is allocating devices. This message means that the catalog was searched in order to find a serial number, but the data set name was not found.

6.7f Referring Back

If the same volume is being used in different steps of a job, the serial number need only be specified once and thereafter can be referred to as follows:

VOL=REF=*.stepname.ddname

where stepname is the name of the step and ddname the name of the DD card on which the volume serial number was given, e.g.,

VOL=REF=*.GOA.DICTIN

6.7g Scratch Tapes at the ISR

Four scratch tapes are available at the ISR for within job use and are labelled as follows:

SCR1	}	for use on 9-track drives
SCR2		
SCR3		
SCR7	}	for use on 7-track drives

6.7h Public Disks

If a temporary disk data set is required within a job, no disk serial number need be specified (the data set name should be absent or assigned a temporary name).

If it is required to store information for some time on disk, a specific disk number should be given. At the ISR disks ISRA and ISRB are available for storing data sets.

6.8 DCB Parameter

6.8a Function

DCB stands for Data Control Block. The parameters provide data characteristics not given elsewhere, e.g.,

DCB=(RECFM=FB,LRECL=80,BLKSIZE=3520,BUFNO=1)

The Data Control Block is actually an area of core set aside for keeping information about a data set while a program is running.

It is filled from three different sources in the order:

- i. Problem program
- ii. DD card
- iii. Label

6.8b Most Commonly Used Subparameters

RECFM	for specifying record format
F	Fixed length, unblocked
FB	Fixed length, blocked
V	Variable length, unblocked
VB	Variable length, blocked
U	Undefined
LRECL	for specifying the logical record length
BLKSIZE	for specifying the blocksize
BUFNO	for specifying the number of input/output buffers to be used while reading/writing the data
DEN	for specifying the density at which data is to be recorded on tape. Also used when an unlabelled tape of non-standard density is read.
0	200 bpi
1	556 bpi
2	800 bpi
3	1600 bpi
TRTCH	for specifying non-standard parity and character conversions
E	Even parity (odd parity is assumed if this is not given)
T	Translate from 8 bit EBCDIC code to 6 bit BCD code. Used when writing data to 7-track tape but can only be used for standard alphanumeric and special characters.
ET	Both E and T
C	Conversion feature, usable with some 7-track tape drives. When writing, 3 (8 bit) bytes of information from memory are written into 4 (6 bit) characters on tape. When reading, 4 characters from tape are stored in 3 bytes of memory. This feature is necessary when writing binary information to 7-track tapes so as not to lose any information.

6.8c When DCB Must Be Specified

Under most circumstances, all DCB information is supplied through the problem program and the label of a data set.

Outlined below are some of the circumstances under which one or more DCB parameters must be given on the DD card.

- (i) When running a program which uses a lot of core, it may be necessary to use "single buffering" on the data files (the normal default is two buffers). In this case, specify

DCB=BUFNO=1 on the appropriate DD cards.

- (ii) When reading an unlabelled tape, i.e., if NL is specified in the LABEL parameter. The DCB information normally obtained from the label must be specified on the DD card, i.e., record format, length and blocksize, e.g.,

DCB=(RECFM=FB,LRECL=120,BLKSIZE=1200)

- (iii) When reading a "dummy" data set--To test a setup, it is possible to define a data set as 'DUMMY'. In this case, since a tape or disk is not actually read, DCB information normally supplied through the label must be given. The mandatory parameter is the blocksize, since the system needs to know this when allocating buffers, e.g.,

DCB=BLKSIZE=3600

- (iv) Other circumstances--Unless otherwise specified in the write-up for a particular program or procedure, DCB information need not normally be given in any other circumstance.

6.8d Opening Data Files

Before a data set is first read or written it is first "opened." At this stage the specified number of buffers (default=2) are assigned each with a size equal to the specified blocksize.

Incompatibilities or missing DCB information result in errors of different kinds, e.g.,

- | | | | |
|-------|--|---|--|
| (i) | Blocksize not defined | - | 013 system completion code |
| (ii) | Blocksize not a multiple of the record length for fixed length records | - | 013 system completion code |
| (iii) | Actual physical blocksize greater than specified blocksize | - | 001 system completion code |
| (iv) | No record format defined | - | Default to U and complete physical block will be read as one logical record. |

6.8e Overriding Information in a Data Set Label

If a data set was written with one logical record length and it is required to read it with a different one, then the required record length may be specified on the DD card and the label value overridden. The actual blocksize, however, must be a multiple of the new record length, e.g.,

If card images with record length of 80 were written to tape and blocked at 3200, then to treat two card images as one logical record, LRECL=160 may be specified on the appropriate DD card. However, if the blocksize was 3600, a 013 completion code would result since 3600 is not a multiple of 160.

6.9 Exercises on DD cards for Input Files

Handout 10 gives some exercises in preparation of DD cards to describe input files. Solutions are:

- (i) `//INDATA DD DISP=(OLD,PASS),DSN=ABCD,
// LABEL=3,UNIT=TAPE,VOL=SER=1000`
- (ii) `//INDATA DD DISP=(OLD,PASS),DSN=DATA.A1234,
// UNIT=TAPE,VOL=SER=9999`
- (iii) `//INDATA DD DISP=(OLD,PASS),LABEL=(2,NL),
// UNIT=TAPE,VOL=SER=12345`
- (iv) `//INDATA DD DISP=(OLD,PASS),DSN=STUDYCEN`

(The file number, unit and volume serial number are retrieved from the catalog.)
- (v) `//INDATA DD DISP=(OLD,PASS),DSN=X40000A,
// UNIT=DISK,VOL=SER=ISRPO3`
- (vi) `//INDATA DD DISP=(OLD,CATLG),DSN=STUDY6,
// UNIT=DISK,VOL=(PRIVATE,RETAIN,SER=USER6)`
- (vii) `//INDATA DD DISP=(OLD,PASS),DSN=STUDY6`

6.10 SPACE Parameter

6.10a Function

To allocate an estimated amount of disk space for a new disk data set. General form:

`SPACE=(unit,(qty,2nd qty,directory),RLSE,CONTIG)`

e.g., `SPACE=(TRK,(100,10))`

6.10b Parameters

- (i) Unit - CYL Cylinder
 TRK Track
 1-4 digit number - Blocks

When a blocksize is specified, the system calculates equivalent tracks taking interblock gaps into account.

- (ii) Qty - A number giving the number of units to be allocated.
- (iii) 2nd Qty

If, while writing the data, it is found that the primary space allocation specified by the first quantity is insufficient, then the

second quantity is used to allocate more of the same units. This is done up to 15 times.

(iv) Directory

Only used for partitioned data sets, e.g., program libraries, and gives the number of directory blocks to be assigned. One directory block can hold names of five members.

(v) RLSE

This positional parameter tells the system to "release" unused space at the end of the job step. If this parameter is not used, then all space originally allocated to the data set by the primary allocation remains assigned to that data set according to the volume table of contents even if it is not being used by the data.

(vi) CONTIG

Specifies that contiguous units are to be allocated. The system always assigns contiguous units if it can, even without this parameter. If used, a job is in danger of being terminated if contiguous units are not available, i.e., its use is not really recommended. The time savings gained from reading a sequential file from contiguous tracks are minimal.

6.10c Examples

With a blocksize of 3520, two blocks will fit in one track. If a data set with not more than 100 blocks is to be written, then use

```
SPACE=(3520,(100),RLSE)
or  SPACE=(TRK,(50),RLSE)
```

If there may be slightly more than 100 blocks:

```
SPACE=(3520,(100,10),RLSE)
```

6.10d Errors Encountered

- (i) If a particular disk is specified and there is less space available than specified, the message:

```
DIRECT ACCESS SPACE NOT AVAILABLE
```

is given.

Note that this is on the basis of the primary allocation. Secondary units are not allocated until they are actually required.

- (ii) If, during the writing of a new data set, a secondary allocation of space is required and there are physically not enough available tracks left on the disk, a system completion code D37 results.
- (iii) If, during the writing of a new disk data set, the secondary quanti-

ty of space is allocated 15 times and more space is still needed, then a system completion code of B37 results.

- (iv) If no SPACE parameter is given for a new disk data set, the message:

'ABSOLUTE TRACKS NOT AVAILABLE'

is given during data set allocation.

6.11 Exercises on DD Cards for Output Files

Handout 11 gives some exercises in the preparation of DD cards for output files. Solutions are:

(i) //DATAOUT DD DISP=(NEW,CATLG),DSN=CARDATA,
// LABEL=2,UNIT=TAPE,VOL=SER=900,
// DCB=(LRECL=80,BLKSIZE=240,RECFM=FB)

(ii) //DATAOUT DD DISP=(NEW,PASS),LABEL=(1,NL),
// UNIT=TAPE,VOL=SER=90000,
// DCB=(LRECL=80,BLKSIZE=80,RECFM=F)

(iii) //DATAOUT DD DISP=(NEW,PASS),DSN=RAWS60,
// LABEL=8,UNIT=TAPE,VOL=SER=900,
// DCB=(RECFM=F,BLKSIZE=80)

(Note that for unblocked data, it is not necessary, although not wrong, to specify a value for LRECL.)

(iv) //DATAOUT DD DISP=(NEW,KEEP),DSN=X499999D,
// LABEL=RETPD=7,SPACE=(800,(500,10),RLSE),
// UNIT=DISK,VOL=SER=ISRA,
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=800)

(v) //DATAOUT DD DISP=(NEW,KEEP),DSN=S60,
// SPACE=(7280,(60,1),RLSE),UNIT=DISK,
// VOL=(PRIVATE,RETAIN,SER=USER20),
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=7280)

(Note that on a 2314 disk, the most efficient usage of the disk is to block the data at 7280.)

(vi) a. //DATAOUT DD DISP=(NEW,PASS),DSN=&A,
// SPACE=(TRK,(70,5)),UNIT=DISK,
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=3520)

b. //DATAOUT DD DISP=(NEW,PASS),DSN=&A,
// UNIT=TAPE,VOL=SER=SCRL,
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=3600)

6.12 Which DD Parameters Are Needed When:

Input (Uncatalogued)		Input (Catalogued)		Output		
Tape	Disk	Tape	Disk	Tape	Disk (permanent)	Disk (temporary)
a) for IBM sort/merge	for IBM sort/merge	a) for IBM sort/merge	for IBM sort/merge	As program write-up indicates	As program write-up indicates	As program writeup indicates
b) if unlabelled tape		b) if unlabelled tape				
Yes	Yes	Yes	Yes	Yes	Yes	No, unless file needed in more than one step
Yes	Yes	Yes	Yes	Yes	Yes	No
Yes unless file 1 of labelled tape	No	No	No	Yes	Yes if expiration date required	No
No	No	No	No	No	Yes	Yes
Yes	Yes	No	No	Yes	Yes	Yes
Yes	Yes	No	No	Yes	Yes	No

The above table gives some guide as to when the various DD parameters are required. More information about these DD parameters and others not discussed at all in this chapter can be found in [1], [2].

Chapter VII

SPECIAL DD STATEMENTS

7.1 Specifying the Printer, Card Punch, and Card Reader

If any of these three units is required, the DD card parameters are not of the same form as has been discussed for tape and disk.

7.1a System Output

In the 360 operating system, there are different classes of 'system output' available: class A, class B, etc. A particular data set may be defined as belonging to one of these classes by specifying SYSOUT=letter, e.g.,

```

        SYSOUT=A
or     SYSOUT=B

```

on the appropriate DD card.

The operator can specify, at any time, to which unit these output classes are to refer. Normally, however, they are set when the operating system is loaded into the computer's memory (i.e., at 'IPL' time) to be the printer and card punch respectively.

7.1b Examples

To output a data set which is defined by ddname OUTPUT, to the printer:

```
//OUTPUT DD SYSOUT=A
```

To output a data set which is defined by ddname DATAOUT, to the punch:

```
//DATAOUT DD SYSOUT=B
```

Sometimes DCB information is required with such data sets, e.g.,

```
//DATAOUT DD SYSOUT=B,DCB=BLKSIZE=80
```

7.1c Using a Special Print Chain - UCS Parameter

For a special print chain, e.g., for printing upper and lower case letters, the printer has to have a different set of characters stored in its buffer. This is done by using a parameter on the DD card, which specifies the type of chain:

```
//OUTPUT DD SYSOUT=A,UCS=(SN,,VERIFY)
```

UCS stands for Universal Character Set. There are various codes for the different sets. The chain normally mounted at the ISR has the code QN.

VERIFY makes the computer wait while the operator verifies that the correct character set has been loaded.

If the HASP spooling system is being used in conjunction with OS, then special print chains are usually requested in a different way. At the ISR, the special forms accounting field (see page 8) is used instead of the UCS parameter.

7.1d System Input

All JCL statements are known as 'system input' and must be input via the device specified by the operator when the operating system is loaded into the computer memory. Other kinds of input may also be entered via the system input device by specifying a DD card with an * on it, e.g.,

```
//INPUT DD *
```

This type of DD card tells the control program that there is an input file following this card.

The system input device is normally the card reader although it is possible for the operator to redefine it as a different device, e.g., a tape unit, at any time.

7.1e Card Input

From what was said in 7.1d, any input data set on cards must be preceded by a DD card with the appropriate ddname specifying the system input device (the card reader). In order that the control program may recognize the end of the card data set and not confuse subsequent JCL statements as part of the file, the data set must be followed by a card punched with /* in the first two columns. This performs the same function as a tape mark on a tape.

e.g., to input a data set with ddname IN:

```
//IN DD *
      Card data
/*
```

7.2 Program Libraries--JOB LIB, STEPLIB

7.2a Function

Program libraries are special partitioned data sets on disk where the data set name is the library name and the member names within the partitioned data set

are the program names. Programs must be stored in program libraries before they can be accessed.

7.2b Retrieving a Program from a Library

When a program is asked for on an EXEC card, the control program, unless directed otherwise, looks for the program in the IBM supplied library (SYS1.LINK-LIB). If the program is not in this library, an 806 system completion code is given.

7.2c Specifying Special Libraries

If a program is not in the IBM supplied library, the control program must be told which library it is in.

This is done with one of the special DD cards: JOBLIB or STEPLIB card.

7.2d JOBLIB

e.g.,

```
//JOBLIB DD DSN=ISRLIB,DISP=OLD
```

This card is placed immediately after the JOB card in a setup and will apply to all steps in the job. That is, each program called for in the job will be searched for in the specified library.

7.2e STEPLIB

e.g.,

```
//STEPLIB DD DSN=MYLIB,DISP=OLD
```

This card is placed immediately after the EXEC card for one step and applies only to that step. The step library takes precedence over the job library, i.e., a program is searched in the STEPLIB and only if not found there will the JOBLIB also be searched.

7.2f Concatenation

When two data sets are required to be considered as one, that is, on finishing reading one, one wants to just continue and read the other, they can be 'concatenated'. DD cards for the two data sets are put next to each other, the second one having the ddname left off; e.g.,

```
//JOBLIB DD DSN=ISRLIB,DISP=OLD
// DD DSN=MYLIB,DISP=OLD
```

In the above example, ISRLIB and MYLIB have been concatenated together and will be treated as one library.

7.2g STEPLIB v. JOBLIB

STEPLIBS were a new feature which came with release 18 of the Operating System. They really make JOBLIBS redundant.

Supposing it is required to execute three programs in one job: IEBGENER from the IBM library, PF51OP from ISRLIB and MINE1 from MYLIB.

Using JOBLIB cards:

```
//Jobname JOB . . .
//JOBLIB DD DSN=ISRLIB,DISP=OLD
// DD DSN=MYLIB,DISP=OLD
//STEP1 EXEC PGM=IEBGENER
.
.
//STEP2 EXEC PGM=PF51OP
.
.
//STEP3 EXEC PGM=MINE1
```

Using STEPLIB cards:

```
//Jobname JOB . . .
//STEP1 EXEC PGM=IEBGENER
.
.
//STEP2 EXEC PGM=PF51OP
//STEPLIB DD DSN=ISRLIB,DISP=OLD
.
.
//STEP3 EXEC PGM=MINE1
//STEPLIB DD DSN=MYLIB,DISP=OLD
```

The above two examples are equivalent.

7.2h Steplibs and Procedures

Steplib cards (unlike joblib cards) may be part of a catalogued procedure so that in a large number of cases the user who makes use of procedures need not be concerned with steplib cards. (See Chapter VIII.)

Chapter VIII

CATALOGUED PROCEDURES

8.1 Purpose

Some JCL statements needed to run a particular program are always the same. To save repunching these cards every time the program is run, a facility exists where part of the JCL needed for a program is stored in a special library on disk. This library is called a catalogued procedure library and the members of this library are individual procedures.

8.2 Description

In every catalogued procedure there is at least one EXEC card of the form:

```
//stepname EXEC PGM=program
```

This is followed by DD cards, usually those that need never concern the user but are used by the program for printing or for temporary work data sets, e.g., a procedure called PROCA might consist of just two statements:

```
//GO EXEC PGM=PF51OP
//SYSPRINT DD SYSOUT=A
```

When the control program reads an EXEC card through the card reader which contains no PGM keyword, e.g., // EXEC PROCA, it assumes that PROCA is a catalogued procedure. Before reading any further cards through the card reader, it will first search the procedure library on disk for a member called PROCA and when found, will replace the // EXEC PROCA statement with the two statements which belong to PROCA.

8.3 Example of Procedure

Handout 13 shows a listing of five members from a catalogued procedure library with member names SCRAT, COPY, LABPRT, FORTGCL, COSOME.

SCRAT is a procedure to make scratching of unwanted disk data sets easier.

Ignoring the first statement for the time being, there is then an EXEC card for program IEHPROGM which is an IBM utility program for scratching, uncataloguing, etc. (See [3].) The JCL requirements of this program are:

- (i) A DD card with ddname SYSPRINT specifying the printer
- (ii) A DD card with optional ddnames for any disk from which something is to be scratched
- (iii) A DD card with ddname SYSIN defining the file which contains information about what is to be scratched

In the SCRAT procedure, there are DD cards for all the disks that are normally mounted on the ISR system, so that the user of this procedure need not provide a DD card for the particular disk in which he is interested unless it is a private one. The SYSIN DD card specifies a file which actually contains control cards to scratch all data sets whose expiration dates have been reached. That is, if the procedure SCRAT is used by itself with no alterations by submitting a job with just the card:

```
// EXEC SCRAT
```

then all expired data sets from the public disks will be scratched.

8.4 Overriding Cards In a Procedure

If the user requires most of the JCL statements from a procedure but wants to make slight changes to one or more, he submits 'override' cards in his setup as well as the EXEC card.

The override cards must specify the ddname of the DD card to be overridden together with the stepname in which it appears (it will be shown later that a procedure can have more than one step in which case one has to distinguish the step with the override card), e.g., a card in the setup of the form:

```
//GO.SYSIN DD *
```

will override information specified for a SYSIN DD card in the step in a procedure with stepname GO.

If it is required to scratch and uncatalog a data set on disk ISRA with data set name A, then one must override the SYSIN specification in the procedure with a DD card which specifies a card file. In the card file, there will be the appropriate control cards for scratching, i.e., the setup will look like this:

```
// EXEC SCRAT
//GO.SYSIN DD *
SCRATCH DSNAME=A,VOL=2314=ISRA,PURGE
UNCATLG DSNAME=A
/*
```

NOTE: The PURGE keyword will ensure that scratching takes place even if the expiration date has not been reached. The format of the control cards as shown above is described in greater detail in [3].

8.5 Output Example of SCRAT Procedure

Handout 12 shows what appears on the printer when the SCRAT procedure is used.

Statements which were supplied by the user on cards are printed with a // in the first two positions.

Statements from the procedure and being used have XX in the first two print positions.

Statements from the procedure which have been overridden by a user supplied statement have X/ in the first two positions. (Note user supplied //GO.SYSIN DD * and overridden X/SYSIN DD.)

8.6 Adding Extra DD Cards to Those in a Procedure

DD cards which are required but whose ddnames do not appear in the procedure may also be supplied at run time.

These cards are punched like override cards but must come in the setup after cards overriding actual DD cards.

Also, if more than one DD statement is being overridden, the override cards must come in the setup in the same order as the DD statements in the procedure. If these rules are not followed, the DD statements do not get overridden correctly. This shows up when the job is run since the procedure statements will be preceded by XX instead of X/.

8.7 Multi Step Procedures

A catalogued procedure may contain JCL for more than one step, i.e., it may contain more than one EXEC card. Each EXEC card in a procedure specifies a program name and thus represents one job step. Since an EXEC card supplied by the user may specify a procedure name this type of EXEC card represents as many job steps as there are EXEC cards in the procedure.

The procedure FORTGCL, a listing of which is shown on handout 13, is an example of a two step procedure.

The first step which has an EXEC card:

```
//FORT EXEC PGM=IEYFORT
```

is for compiling a program written in the FORTRAN language into machine language, storing the machine language code in a temporary disk data set &LOADSET.

The second step, which has an EXEC card:

```
//LKED EXEC PGM=IEWL
```

is for linking the compiled code from step 1 with already compiled subroutines from other sources, storing the complete program in another temporary disk data set &GOSET.

The step names of the two steps are FORT and LKED respectively.

To override or add DD cards to the first step, the ddnames must be prefixed by FORT and to the second step by LKED.

An actual setup making use of this procedure might look something like this:

```

// EXEC FORTGCL
//FORT.SYSIN DD *
FORTRAN statements
/*
//LKED.SYSLMOD DD DSN=MYLIB(PROGA),DISP=SHR,VOL=SER=ISRB,UNIT=DISK
//LKED.SYSIN DD *
Control statements for the linkage editor program
/*

```

Using the procedure, the only DD cards that need to be provided are:

- (i) In the first step a DD card to specify that the FORTRAN statements to be compiled are on cards.
- (ii) In the second step, a DD card specifying where the linked, ready to execute program is to be stored (here in MYLIB) and a DD card specifying that control statements for the linkage editor program are on cards.

The ddname SYSIN is the one required by program IEYFORT for specifying the input file.

The ddname SYSLMOD is the one required by program IEWL for specifying the output file. SYSIN specifies the file containing the linkage editor control statements.

8.8 Aliases

Any member of a partitioned data set may have aliases, as well as its actual name. The alias names appear in the partitioned data set directory and point to the same place on the disk as the real name. Thus if the alias name is specified, the same member is found as if the real name was specified.

8.9 Creating a Catalogued Procedure

Suppose that a particular sequence of steps:

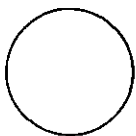
```

Copying cards to tape (program IEBGENER)
Sorting them (program IERRC000)
Checking for duplicate/bad/missing decks (program ZMERCH)

```

is often required.

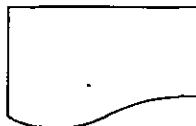
This process can be represented diagrammatically in the form of a 'system flowchart' using the following symbols:



Tape



Card



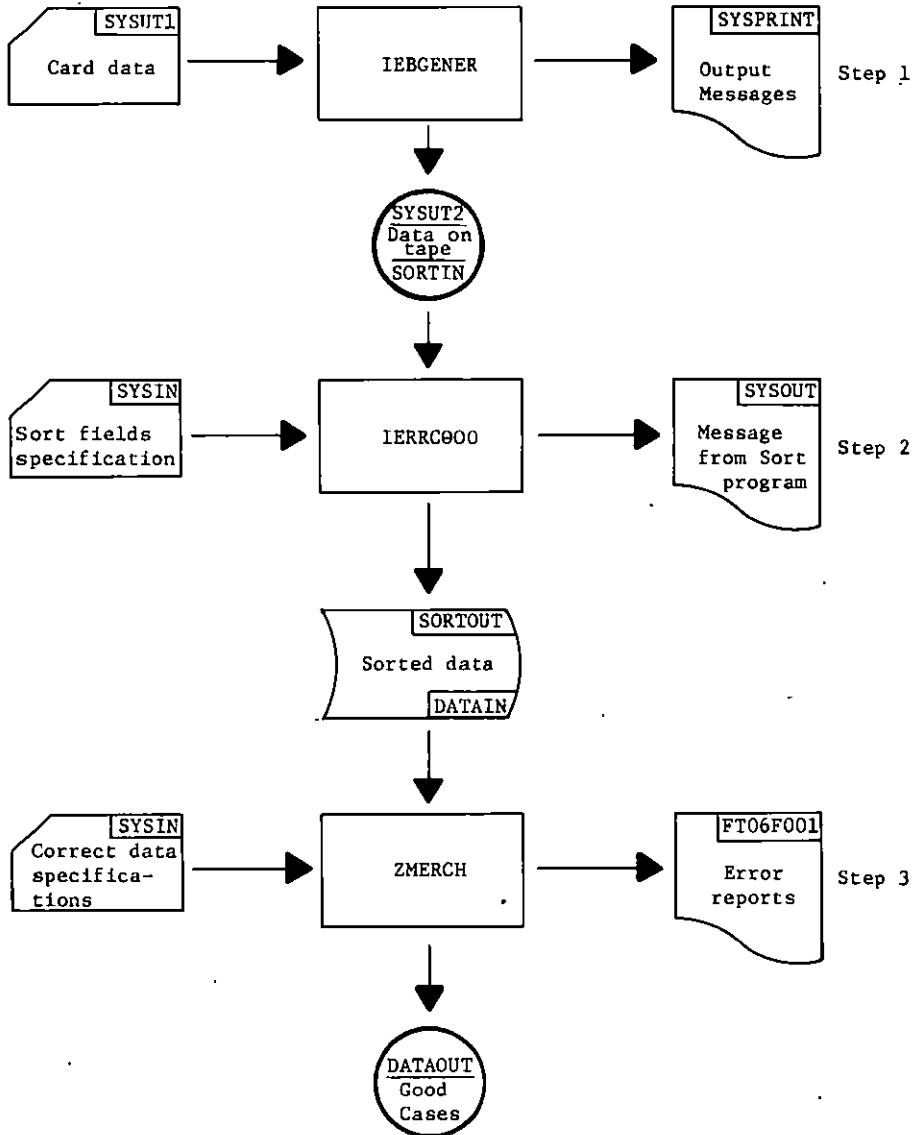
Printed Output



Disk



Processing



Assuming that the output from the sort program is not required and can be merely temporarily stored on disk, the only things that can change from run to run in this series of steps are the input data; the data set name, file number and tape number on which the input data is to be written; the sort fields specification; the correct deck specifications; the data set name, file number and tape number on which the good cases are to be written after checking the decks.

A catalogued procedure can be constructed containing all necessary JCL statements and information except the items mentioned above.

The ddnames for the various files are shown on the chart. Handout 13 shows what the procedure (named COSOME) might look like.

The three steps have been given stepnames GOA, GOB, GOC. In order to use

the procedure, one first needs an EXEC card to call the procedure.

Then one needs to supply a tape number and data set name for the output and a DD card defining the card reader as the input device for step A; a DD card defining the card reader for step B; and a DD card specifying the tape number and data set name for the output, and specifying the card reader for input of the deck specifications for step C, e.g.,

```
// EXEC COSOME
//GOA.SYSUT2 DD VOL=SER=9000,DSN=CUDATA1
//GOA.SYSUT1 DD *
    card data
/*
//GOB.SYSIN DD *
    sort fields control statement
/*
//GOC.DATAOUT DD VOL=SER=9001,DSN=CUDATA2
//GOC.SYSIN DD *
    control cards
/*
```

8.10 Symbolic Parameters in Catalogued Procedures

8.10a Function

Parameters on JCL statements can be given symbolic names in catalogued procedures and resolved when the procedure is called for by setting the symbolic name equal to a real value on the EXEC card calling for the procedure. Symbolic names are indicated by a name preceded by an &, e.g., if a DD card in a procedure had as one of its parameters:

```
----DSN=&NAME----
```

where & indicates a symbolic name which here is NAME; then this could be set equal to the correct dataset name when the procedure is called for as follows:

```
// EXEC procedure name,NAME=dataset name
```

8.10b Example

A program to print tape labels called LABPRT exists in the library OSIRPGM. If there were no catalogued procedure set up for this program, the JCL the user would have to supply would be as follows:

```
1 //GO EXEC PGM=LABPRT
2 //STEPLIB DD DSN=OSIRPGM,DISP=SHR
3 //FT06F001 DD SYSOUT=A
4 //FT08F001 DD UNIT=TAPE,DISP=(OLD,PASS),
5 // LABEL=(file no,BLP,,IN),VOL=SER=tape number
```

where the appropriate file number and tape number would be punched on card 5.

Supposing a procedure called LABPRT existed in the catalogued procedure library which contained the statements which never changed, i.e., 1-3.

Then to execute the program, the user would have to punch two cards only:

```
// EXEC LABPRT
//GO.FT08F001 DD UNIT=TAPE,DISP=(OLD,PASS),
// LABEL=(file no,BLP,,IN),VOL=SER=tape number
```

When the control program reads the EXEC card it will substitute for this all the JCL statements corresponding to the procedure (as stated, cards 1-3 above). It will then add to these the DD card //GO.FT08F001... as supplied by

the user specifying the appropriate file and tape numbers.

Supposing a fourth card exists in the procedure:

```
4. //FT08F001 DD UNIT=TAPE,DISP=(OLD,PASS)
```

Then the user only has to supply an EXEC card together with an override DD card for //FT08F001 giving the LABEL and VOL parameters. Any parameter on a DD card not specifically overridden remains as it exists in the procedure, i.e.,

```
// EXEC LABPRT
//GO.FT08F001 DD LABEL=(file no, BLP,,IN),VOL=SER=tape number
```

The only things that vary from run to run are the file number and tape number. Supposing these are assigned symbolic names of FILE and TAPE respectively. Now a fifth card may be introduced into the procedure:

```
5. // LABEL=(&FILE, BLP,,IN),VOL=SER=&TAPE
```

which will be a continuation for the fourth card.

All the user now needs to punch to print, for example, file 1 of tape 9000 is:

```
// EXEC LABPRT,TAPE=9000,FILE=1
```

The actual values for the symbolic parameters are substituted in the appropriate places in card 5 in the procedure.

A listing of the procedure LABPRT with all the above features is given in handout 13.

8.10c Default Symbolic Name Values

A further simplification in the user setup may be made by setting the most commonly used value of a symbolic name to a default value. This is done by means of a 'PROC' statement which can optionally appear as the first statement (i.e., before the EXEC card) in a catalogued procedure, e.g.,

```
// PROC FILE=1
```

This statement sets a default value of 1 for the symbolic name FILE. That is, if the user does not punch a value for the symbolic name FILE on his EXEC card, then the control program will use the value given on the PROC statement in the catalogued procedure.

8.10d Program Library Names in the OSIRIS Catalogued Procedure Used at the ISR

The STEPLIB card in the OSIRIS procedure at the ISR specifies four different program libraries, each with a different symbolic name:

```
//STEPLIB DD DSN=&LIB,DISP=SHR
// DD DSN=&LIB1,DISP=SHR
// DD DSN=&LIB2,DISP=SHR
// DD DSN=&LIB3,DISP=SHR
```

In addition, on the PROC statement, default data set names are supplied for the four libraries:

```
// PROC LIB=OSIRPGM,LIB1=SRCLIB,LIB2=CPSLIB,LIB3=ISRLIB
```

If none of the LIB symbolic names (LIB, LIB1, LIB2, LIB3) are given an actual value on the EXEC card when the procedure is used, then the control program will substitute the value set in the PROC statement, i.e., the actual STEPLIB statement used will be:

```
//STEPLIB DD DSN=OSIRPGM,DISP=SHR
//          DD DSN=SRCLIB,DISP=SHR
//          DD DSN=CPSLIB,DISP=SHR
//          DD DSN=ISRLIB,DISP=SHR
```

If it is required to use a program out of a different library, e.g., MYLIB, then this library must be specified on the EXEC card. However, if the OSIRIS monitor is being used to load the program, then the library OSIRPGM which contains the monitor program (ISRSYS) is still required. One of the symbolic names LIB1, LIB2 or LIB3 should be used, e.g.,

```
// EXEC OSIRIS,LIB1=MYLIB
```

The control program searches down the set of concatenated libraries until it finds the program it requires. If the same program name exists in OSIRPGM and MYLIB, then the one from OSIRPGM will be used. If one particularly wants the one from MYLIB but still requires the monitor program, then the EXEC should be punched:

```
// EXEC OSIRIS,LIB=MYLIB,LIB1=OSIRPGM
```

Note that if more than one symbolic name is being given a value on the EXEC card, the order is immaterial. The last EXEC card could equally well be punched:

```
// EXEC OSIRIS,LIB1=OSIRPGM,LIB=MYLIB
```

Chapter IX

ALLOCATION, DEALLOCATION, AND COMMON ERRORS

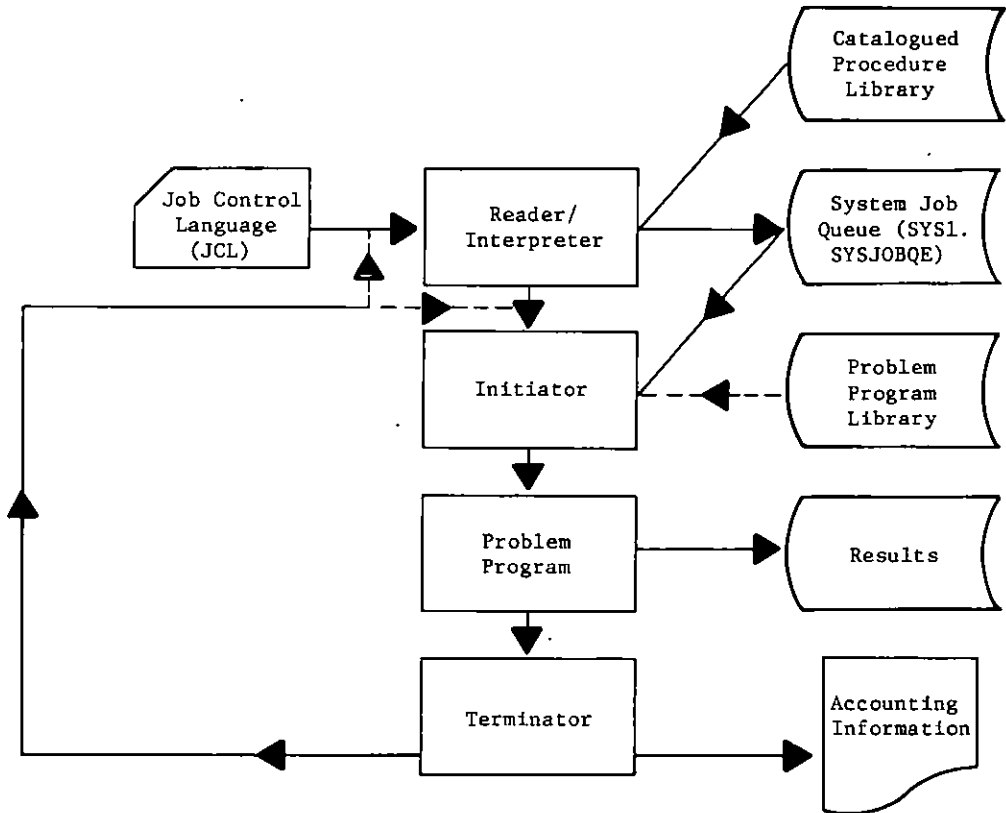
9.1 OS Job Scheduler

This is the IBM supplied software which checks JCL, allocates units and gets required problem programs into the computer memory. It is divided into three main parts:

- (i) Reader interpreter which reads JCL statements and other cards for a job from the reader and from catalogued procedures; performs syntax checks and writes out into a system job queue on disk. Continues reading JCL cards until a new JOB card is encountered, i.e., more than one step's worth of JCL may be read and checked at one time.
- (ii) Initiator which allocates devices for one job step and gets the required problem program into the computer memory.
- (iii) Terminator which deallocates all devices; prints out accounting information.

This process is shown in very rough outline in the following chart:

FLOW OF JOBS THROUGH THE COMPUTER



9.2 Allocation Messages

Allocation messages are printed on the output immediately below the listing of the JCL. They show which physical device has been allocated to which data set. Each device has a three digit code. For example, at the ISR, these are:

130-135	6 disk units
280	7-track tape drive
281-283	9-track tape drives (800 bpi)
290-291	9-track tape drives (800/1600 bpi)
00A	card reader
015	card punch
00E	printer
01F	consol/typewriter

The allocation messages (indicated by the code IEF2371) give one of the above device codes together with the ddname of the DD card defining the relevant data set. Thus, in handout 14A, it can be seen, that unit 130 has been allocated for five data sets defined by DD cards with ddnames FT47F001, FT48F001, FT01F001,

FT50F001, and DICTIN. Unit 281 (a tape drive) has been allocated for data set defined by the DATAIN DD card.

9.3 Deallocation Messages

Deallocation messages, indicated by the code IEF2851, are printed at the end of the output from each step giving the current status of all data sets used in that step. At the end of the job, deallocation messages are given for any data sets that were 'passed' in the individual steps. If the computer is being run with the HASP system, all allocation and deallocation messages for all the steps in the job are printed together before the output from the application programs being used.

Each deallocation message occupies two lines. The first line gives the data set name together with its status. The disposition is derived from the second DISP parameter on the appropriate DD card, e.g., if (NEW,DELETE) had been specified, then the status would be given as 'DELETED' at deallocation time. If the action specified in the DISP parameter is not possible for some reason, e.g., DELETE requested for a data set whose expiration date has not expired, the deallocation message indicates this, e.g., in the handout 14B, a message NOT DELETED 8 is output for data set ISRDICT meaning that the system tried to delete ISRDICT as requested on the DISP parameter but was unable to do so as the expiration date had not been reached.

NOT CATLCD 2 means that a request was made to catalog the data set but there was already an entry in the catalog for it. The various different codes are given in [5].

The second line of the deallocation message gives the volume serial number of the volume containing the data set. If this is irrelevant, e.g., for the printer, it is left blank.

Looking at handout 14B again, it can be seen that a data set with name X48070.NEWS on volume ISRES5 was kept, whereas a temporary data set on volume ISRES4 was deleted.

After the status of each data set has been given, messages indicated by the code IEF280E which relate to the status of devices are issued. In particular, if a device has been unloaded a message, IEF280E K (standing for keep) with the unit number and volume number of the volume on that unit, is issued. On a tape this means that the tape has rewound and unloaded itself. If it is to be used again it has to be physically remounted by the operator. This type of message is usually issued if a data set on tape has a KEEP disposition on its DD card.

9.4 Types of Errors

More details on errors and error codes can be found in [5].

System errors reported by the IBM/360 computer fall into three major categories: (i) JCL errors, (ii) Completion codes, (iii) Errors detected during the execution of application programs written in higher level languages such as FORTRAN or COBOL. Application programs themselves also issue error messages but these are the concern of the program, not the operating system.

9.5 JCL Errors

The majority of JCL errors are self explanatory. They occur at two stages: syntax errors detected while checking JCL, and errors occurring during the allocation or deallocation of devices to data sets.

9.5a Syntax Errors

When these are detected a message starting with an error code of the form IEFnnnn is printed on the line immediately below the offending statement.

The following points should be noted:

- (i) Blanks should only appear on either side of an operator (i.e., JOB, EXEC, DD) or at the beginning of a continuation card. Blanks anywhere else are errors. Blanks actually act as delimiters between the four main fields on a JCL card:

name	operator	parameter	comments
------	----------	-----------	----------

Therefore, if a blank appears in the middle of the parameters, everything following the blank will be treated as a comment.

- (ii) The first two or three print positions of the listing of JCL statements on the output are one of the following:

//	indicating a statement supplied by the user
X/	indicating a DD statement from a catalogued procedure which is being overridden by a user supplied statement
XX	indicating a statement from a procedure not being overridden by the user
***	indicating a comment card (starting with /**) supplied by the user
/**	indicating comments generated by having a blank after a parameter of a DD card, and continuing on subsequent // type cards

- (iii) Once a JCL syntax error is encountered no other steps in the job are processed. However, all the JCL is checked.
- (iv) If continuation cards are being used for DD statements, make sure there is a comma after the last item before column 71 of the preceding card. If the comma is missing, the continuation card is treated as a new JCL statement but since it contains no operator, the message 'UNIDENTIFIED OPERATION FIELD' is given.
- (v) Do not continue punching after column 71 of any card.

9.5b Allocation/Deallocation Errors

Even if the JCL is syntactically correct (i.e., parameters spelled right, etc.) some of the requisite parameters may be found to be missing when devices

are actually allocated to data sets.

- (i) A data set being written to disk must have a space allocation. A DD card defining a data set with DISP=NEW and UNIT=DISK but with no SPACE parameters results at allocation time in the message: 'ABSOLUTE TRACKS NOT AVAILABLE.'
- (ii) A new data set may not be defined on a disk which already contains one of the same name. The message 'DUPLICATE NAME ON VOLUME' results.
This sometimes happens when a job creating a disk data set terminated abnormally, and the job is being rerun.
- (iii) If a volume serial number is not given for an old data set, the catalog is searched for that name. If the name is not found in the catalog, the message 'LEVEL OF INDEX MISSING' results.
- (iv) 'INSUFFICIENT DISK SPACE' means more disk space has been asked for on a particular disk than is available. Either specify a different disk, or if possible, reduce the space allocation.
- (v) NOT CATLGD n where n is a number. Occurs at deallocation time when a data set is not catalogued even though its DISP parameter specified CATLG. In particular, NOT CATLGD 2 means that there was already an entry in the catalog for the data set.
- (vi) NOT DELETED n occurs at deallocation time if a data set which according to its DISP parameters should have been deleted was not for some reason, e.g., if the expiration date had not been reached; or if the data set was never actually used and there was therefore nothing to delete.
- (vii) DELETED at deallocation time means deleted. If this was not intended, beware--the DISP parameter was probably specified incorrectly.

9.6 Completion Codes

If a job step terminates abnormally then a completion code appears at the end of the device allocation messages as shown in handout 3b. If the HASP spooling system is being used, the completion code also appears in the HASP log at the start of the output. The abnormal termination message takes the form:

```
COMPLETION CODE    SYSTEM.....    USER.....
```

9.6a System Completion Codes

These are given by IBM software and can be user, program or OS faults. Some of the most common ones are listed below:

```
001                Input/output error.  a) In the case of tapes, there
                    could be a scratch or other physical fault on the tape
                    or else the density specification is incorrect.
```

- b) Block or logical record lengths are incompatible.
 c) A hardware error and the computer room should be notified.
 d) Reading and writing into the same data set simultaneously.
- OC2,OC1 If 5000 or 50 appears as the contents of register 15, then there is a DD card missing, or an incorrect DD name in the setup.
- OC4,OC5,OC6 Probably stray programming errors. Get advice from a programmer.
- 013 No DCB information given when needed, or DCB on DD card incompatible with that in the standard label.
- 137 Something wrong at the end of a magnetic tape (computer was unable to process trailer label). Rerun job which created the file in question.
- 213 A disk data set with DISP=OLD and specific volume serial number does not exist on the named disk pack.
- 222 The operator cancelled the job for some reason.
- 237 As for 137.
- 322 CPU time limit on job card exceeded.
- 613 Magnetic tape volume or header label problem. Could be trying to access a file not on the tape. Get a listing of the trailer label of the preceding file and the header label of the unaccessible file. Remember that there are actually three files for every data file on a standard labelled tape--one for the header label, one for the actual data and one for the trailer label. If the 613 error occurred while trying to access the third data file on a tape, then a listing of the 6th and 7th files (trailer label of 2nd data file, header label of 3rd) should be requested.
- It may be necessary to relabel the tape and recreate the files.
- 80A Run out of core space. Try setting DCB=(BUFNO=1) on DD cards.
- 813 No DSNAME specified for a data set or else the DSNAME does not agree with that in the label on the tape.
- B37 Not enough physical space on the disk to allocate further extents for a new data set. Either allocate more space in the primary allocation (this makes sure that the data set is allocated initially to a disk with plenty of space) or specify a different disk pack.
- D37 Not enough space allocated for a new disk data set. Increase space allocation on DD card and rerun job.

E37 No more room for another member in a partitioned data set. (The data set should be compressed, or copied somewhere and space reallocated.)

9.6b User Completion Codes

These are given by the program being executed and should be in the program documentation.

9.7 Errors Generated by Fortran Written Programs

IHC215I	Reading non-numeric characters when numeric are expected.
IHC217I	Reached the end of data when not expecting it. Do you have all the requisite control cards, and end of data cards?
IHC218I	Sometimes an I/O error which disappears when you re-run. Otherwise, see a programmer.
IHC219I	Missing DD card or incorrect ddname on a DD card.
IHC230I	Fortran source program error. See a programmer.
IHC210I	Usually indicates an attempt to divide by zero. Program continues to execute but results will be of doubtful validity. Usually indicates a data error.
IHC251I	Trying to take square root of a negative number. Check data.
IHC253I	Trying to take log of a negative number. Check data.

In the ISR's OSIRIS programs, most reading and writing of data and control cards is not performed with Fortran instructions. Matrices are however usually read in Fortran. The errors IHC215I and IHC217I are nearly always caused by errors in matrix input.

9.8 Explanation of JCL Errors on Handout 15

//DD01	UNIT parameter missing. A message 'UNIT SPECIFIES INCORRECT NAME' will be given at allocation time.
//DD02	If B was catalogued, the volume serial number would be retrieved from the catalog and this would be OK; otherwise, 'LEVEL OF INDEX MISSING' would be gotten at allocation time.
//DD03	dsname has 9 characters (8 is maximum between periods).
//DD04	The first field of a LABEL parameter is the file number which must be numeric. This should have read LABEL=(,NL)

//DD05 Assumes D to be catalogued. However, since the block-size is not a multiple of the record length, a system completion code of 013 will probably be given when the data set is first used.

//DD06 No disposition is given so the default of (NEW,DELETE) will be taken. However, with a NEW disk data set, a SPACE parameter must appear. A 'ABSOLUTE TRACKS NOT AVAILABLE' message will appear on allocation.

//DD07 (PRIVATE,RETAIN, is being used as the serial number of the disk. The volume parameter should have been: VOL=(PRIVATE,RETAIN,SER=D12).

//DD08 If a scratch tape is being used to store a temporary file, this is OK (a temporary data set name will be allocated by the system for the duration of the step).

//DD09 80,RECFM=F,LRECL=80, is all being taken as the BLKSIZE value. Parentheses are required if there is more than one sub-parameter, i.e., DCB=(BLKSIZE=80,RECFM=F,LRECL=80).

//DD10 Punching has continued beyond column 71. Note also that everything on the next card, which was meant to be a continuation card, is being treated as comments (/* in first three positions).

//DD11 Space can be allocated in units of CYL, TRK or else a numeric blocksize. Because the unit specified TRACK is more than three characters, the control program assumes it to be a numeric blocksize and then discovers that it is alphabetic.

//DD12 No DSNNAME specified. Since the disposition is OLD and the LABEL specification implies a standard labelled tape, when the data set is first used a completion code of 813 will result indicating that the temporary data set name assigned for the step is not the same as the data set name in the label of the file.

//DD13 UMIT instead of UNIT specified.

//DD14 No JOB, EXEC, or DD, i.e., no recognizable operation field.

//DD15 Will result in 'UNIT SPECIFIES INCORRECT DEVICE' at allocation time since DISC is not a valid unit specification.

9.9 Explanation of Allocation Errors Caused by JCL Errors on Handout 16

9.9a Job F001794 (EXEC card error)

By using the // EXEC procedure name form of the execute card, the system expects to find the specified name in the procedure library. The message IEF612I indicates that the procedure DEBE is not in the procedure library (DEBE is actually a program at the ISR, not a procedure) and the EXEC card should have been:

```
// EXEC PGM=DEBE
```

9.9b Job F001797 (Data set allocation error)

No space was specified on the DD card for a new disk data set. Hence the message IEF256I.

9.9c Job F001798 (Allocation error)

The data set X48070JR already exists on disk ISRES5 and therefore cannot be created again as a new data set; hence message IEF253I. (SYSUT2 was given an alternative user name of DATAOUT through the DD card //SYSUT2 DD DDNAME=DATA-OUT but the system still thinks of it as SYSUT2).

9.9d Job F001799

The volume serial number ISRD01 is spelled with a letter O instead of a zero. The operator had to cancel the job since there was no disk with the specified serial in the computer room.

REFERENCES

1. IBM, 360 Operating System Job Control Language Reference. Form GC28-6704.
2. IBM, 360 Operating System Job Control Language User's Guide. Form GC28-6703.
3. IBM, 360 Operating System Utilities. Form GC28-6586.
4. IBM, Introduction to 360 Direct Access Storage Devices and Organization Methods. Form C20-1649.
5. IBM, 360 Operating System Messages and Codes. Form GC28-6631.
6. OSIRIS III Manuals. Institute for Social Research, University of Michigan.

HANDOUTS

```
/**
/**  EXAMPLES OF BAD JOB CARDS  HANDOUT 1
/**
//M016060  JOB  (,
//          430000, 1234),BROWN
//M016061  JOB  (,
//          498980),WILSON,MGSLEVEL=(0,0)
//          JOB  (499999),SMITH
//M016063  JOB  (,
//          (499990,OEO,1,6),HEATH
//M016064  JOB  (,
//          423450,MAULDING
//M016065  (,
//          412345,ABCD,,2),POWELL,CLASS=Z
//M016066  (,
//          470000,1234.ABCD),EDEN
```



```
//M016060 JOB (,  
// 430000, 1234),BROWN
```

```
***
```

```
*** EXAMPLES OF BAD JOB CARDS
```

```
***
```

```
IEF621I EXPECTED CONTINUATION NOT RECEIVED
```

```
IEF607I JOB HAS NO STEPS
```

```
IEF272I - STEP WAS NOT EXECUTED.
```

```
//M016061 JOB (,  
// 498980),WILSON,MGSLEVEL=(0,0)  
IEF630I UNIDENTIFIED KEYWORD ON THE JOB STATEMENT  
IEF607I JOB HAS NO STEPS
```

```
IEF272I - STEP WAS NOT EXECUTED.
```

```
// JOB (999,499999),SMITH  
IEF635I JOBNAME MISSING ON THE JOB STATEMENT  
IEF607I JOB HAS NO STEPS
```

```
IEF272I - STEP WAS NOT EXECUTED.
```

```
//M016063 JOB (,  
// (499990,OEO,1,6),HEATH  
IEF622I UNBALANCED PARENTHESIS ON THE JOB STATEMENT  
IEF607I JOB HAS NO STEPS
```

```
IEF272I - STEP WAS NOT EXECUTED.
```

//M016064 JOB (008,
// 423450,MAUDLING
IEF622I UNBALANCED PARENTHESIS ON THE JOB STATEMENT
IEF607I JOB HAS NO STEPS

IEF272I - STEP WAS NOT EXECUTED.

//M016065 JOB (008,
// 412345,ABCD,,2),POWELL,CLASS=Z
IEF639I INVALID CLASS DESIGNATION Z
IEF607I JOB HAS NO STEPS

IEF272I - STEP WAS NOT EXECUTED.

//M016066 JOB (008,
// 470000,1234.ABCD),EDEN
IEF624I INCORRECT USE OF PERIOD ON THE JOB STATEMENT
IEF607I JOB HAS NO STEPS

IEF272I - STEP WAS NOT EXECUTED.

Handout 2
(page 2)

HASP SYSTEM LOG

```

$ 17.44.15 JOB 216 -- M008550 -- BEGINNING EXEC - PART 1 - CLASS C
*17.44.48 JOB 216 IEF233A M 282,SCR1 ,M008550,GO < MOUNT >
*17.47.27 JOB 216 IEC149I 813-04,M008550,GO,SYSUT1,282,SCR1 ,ABCD
*17.47.35 JOB 216 IEF450I M008550 .GO ABEND S813 TIME=17.47.35
*17.47.38 JOB 216 IEF280E K 282,SCR1 ,M008550,GO
$ 17.47.41 JOB 216 END EXECUTION.

```

5 CDS RD 28 LINES 0 CDS PUN 55 DA 1 MNTS 6DD .05M CPU OK CORE

COST: \$2.16

```

//M008550 JOB (,
//      468363,S62,,2),RATTENBURY,CLASS=C
//      EXEC COPY
//DATAIN DD DSN=ABCD,VOL=SER=SCR1,UNIT=TAPE,DISP=(OLD,PASS)
//DATAOUT DD SYSOUT=A,DCB=(BLKSIZE=80,LRECL=80,RECFM=F)
IEF236I ALLOC. FOR M008550 GO
IEF237I OE3 ALLOCATED TO SYSPRINT
IEF237I 282 ALLOCATED TO SYSUT1
IEF237I OE4 ALLOCATED TO SYSUT2
IEC149I 813-04,M008550,GO,SYSUT1,282,SCR1 ,ABCD
COMPLETION CODE - SYSTEM=813 USER=0000
IEF285I ABCD PASSED
IEF285I VOL SER NOS= SCR1 .
IEF373I STEP /GO / START 73260.1744
IEF374I STEP /GO / STOP 73260.1747 CPU OMIN 03.20SEC MAIN 24K LCS OK
IEF285I ABCD KEPT
IEF285I VOL SER NOS= SCR1 .
IEF280E K 282,SCR1 ,M008550,GO
IEF375I JOB /M008550 / START 73260.1744
IEF376I JOB /M008550 / STOP 73260.1747 CPU OMIN 03.20SEC

```

System Completion Code at an Abnormal Termination

Handout 3b

```

/**          JCL CLASS
/**      THE EXECUTE CARD
/**
//STEP1   EXEC   PGM=IEFBR14           1
//STEP2   EXEC   PA015                 2
//STEP3   EXEC   PA020,COND=EVEN      3
//STEP4   EXEC   PC210,COND=ONLY      4
//STEP5   EXEC   PGM=PC200P,PARM='000001' 5
//STEP6   EXEC   PGM=SORT,PARAM='YES'  6
//123     EXEC   PGM=A                 7
//A12345678 EXEC  PGM=B                 8
//        EXEC   PGN=XYZ               9
//        EXEC   PGM=1                 10
//STEP7   EXEC   PROGRAM=IEBPTPCH     11
//STEP8   EXEC   SORTDISK             12
//STEPA   EXEC   PA008,COND=EVEN      13
//STEPB   EXEC   PGM=IEBGENER, COND=EVEN 14
//STEPD   EXEC   PGM=IEFBR14 ,COND=ONLY 15
//STEPD   EXEC   CARDSORT             16
//STEPE   EXEC   PGM=PHA,COMD=EVEN    17
//STEPF   EXEC   PA002,TAPE=99999     18

```

```
//E032864 JOB (008,  
// 480700,SCRI),RATTENBURY  
//STEP1 EXEC PGM=IEFBRI4
```

IEF272I - STEP WAS NOT EXECUTED.

```
//STEP5 EXEC PGM=PC200P,PARM='000001'  
IEF629I INCORRECT USE OF APOSTROPHE IN THE PARM FIELD  
IEF272I - STEP WAS NOT EXECUTED.
```

```
//STEP6 EXEC PGM=SORT,PARAM='YES'  
IEF630I UNIDENTIFIED KEYWORD IN THE PGM FIELD  
IEF272I - STEP WAS NOT EXECUTED.
```

```
//123 EXEC PGM=A  
IEF647I NON-ALPHABETIC FIRST CHARACTER OF NAME ON THE EXEC STATEMENT  
IEF272I - STEP WAS NOT EXECUTED.
```

```
//A12345678 EXEC PGM=B  
IEF642I EXCESSIVE PARAMETER LENGTH ON THE EXEC STATEMENT  
IEF272I - STEP WAS NOT EXECUTED.
```

```
// EXEC PGM=XYZ  
IEF630I EXCESSIVE PARAMETER LENGTH ON THE EXEC STATEMENT  
IEF272I - STEP WAS NOT EXECUTED.
```

```
// EXEC PGM=1  
IEF647I NON-ALPHABETIC FIRST CHARACTER OF NAME IN THE PGM FIELD  
IEF272I - STEP WAS NOT EXECUTED.
```

//STEP7 EXEC PROGRAM=IEBTPCH
IEF630I UNIDENTIFIED KEYWORD ON THE EXEC STATEMENT
IEF272I - STEP WAS NOT EXECUTED.

//STEP8 EXEC SORTDISKC
IEF642I EXCESSIVE PARAMETER LENGTH ON THE EXEC STATEMENT
IEF272I - STEP WAS NOT EXECUTED.

//STEPB EXEC PGM=IEBGENER, COND=EVEN
IEF621I EXPECTED CONTINUATION NOT RECEIVED
IEF272I - STEP WAS NOT EXECUTED.

//STEP C EXEC PGM=IEFB14 ,COND=ONLY
IEF272I - STEP WAS NOT EXECUTED.

//STEPD EXEC CARDSORT
IEF612I PROCEDURE NOT FOUND
IEF272I - STEP WAS NOT EXECUTED.

//STEPE EXEC PGM=PHA,COND=EVEN
IEF630I UNIDENTIFIED KEYWORD IN THE PGM FIELD
IEF272I - STEP WAS NOT EXECUTED.

Standard Labelled Tape Description

a) Structure of Standard Labelled Tape

```

                                Volume Label

Header                          (Header 1 for file 1 )
Label                          (Header 2 for file 1 )
                                Tape Mark          )
                                Data for file 1     )
                                Tape Mark          )
Trailer                         (Trailer 1 for file 1)   File 1
Label                          (Trailer 2 for file 1)
                                Tape Mark          )

                                Header 1 for file 2 )
                                Header 2 for file 2 )
                                Tape Mark          )
                                Data for file 2     )   File 2
                                Tape Mark          )
                                Trailer 1 for file 2)
                                Trailer 2 for file 2)
                                Tape Mark          )

Header 1 for file 3
,
,
,

Trailer 1 for file N
Trailer 2 for file N
Tape Mark
Tape Mark
```


Handout 6

(page 2)

b. Contents of labels

The volume label, which must be written on a tape before it can be used as a labelled tape (use procedure PA001) is an 80 character block containing the volume serial number of the tape.

Header 1 and Trailer 1 for each file are 80 character blocks containing the following information:

Character	1-4	HDR1	(for header)
		EOF1	(for trailer)
	5-21	Data set name of file	
	22-27	Volume serial number of tape	
	28-31	Volume sequence number giving the number of this volume relative to the one on which the data set begins	
	32-35	File number	
	36-39	Generation number of file	
	42-47	Creation date of file (YYDDD)	
	48-53	Expiration date of file (YYDDD)	
	55-60	Trailer only - gives a count of the number of physical blocks in the file	

Header 2 and Trailer 2 for each file are 80 character blocks containing the following information:

Character	1-4	HDR2	(for header)
		EOF2	(for trailer)
	5	Record format F = fixed V = variable U = undefined	
	6-10	Block length (or maximum block length for V and U type records)	
	11-15	Record length (0 for U type records)	
	16	Tape density = 2 for 800 bpi	
	17	1 if this is not the first volume of the data set, 0 otherwise	
	18-25	Job name of job creating the data set	
	26	Slash (/)	
	27-34	Stepname of step creating the data set	
	37	Blank records do not contain printer control characters characters A records contain ASA control characters M records contain machine control characters	

Each file of a standard labelled tape actually consists of three files--the header label, the data and the trailer label, but they are processed as one file when the specification SL is given in the LABEL parameter on a DD card.

CONTENTS OF VTOC ON VOL ISRD01							
DATA SET NAME	CREATED	PURGE	FILE TYPE	EXTENTS	FILE SERIAL	VOL. SEQ.	SECURITY
X46206.BIAS100C	30170	33570	SEQUENTIAL	00004	ISR001	00001	NO
X46206.BIAS130C	32770	35070	SEQUENTIAL	00003	ISR001	00001	NO
X46206.BIAS130A	32770	35070	NOT DEFINED	00001	ISR001	00001	NO
DATA.FD35	32470	33570	SEQUENTIAL	00001	ISR001	00001	NO
X453948G.FTL	13270	36599	PARTITIONED	00004	ISR001	00001	NO
X46206.FD170C	32370	33570	SEQUENTIAL	00001	ISR001	00001	NO
X46206.SKDDDDC	32770	33570	SEQUENTIAL	00001	ISR001	00001	NO
X45758.CAR763	29970	36570	SEQUENTIAL	00013	ISR001	00001	NO
OSIRIS2	13170	33399	PARTITIONED	00007	ISR001	00001	NO
OSIRISUB	13170	33399	PARTITIONED	00001	ISR001	00001	NO
X46206.BIAS100A	30170	33570	SEQUENTIAL	00001	ISR001	00001	NO
X49513DR	31170	33399	PARTITIONED	00001	ISR001	00001	NO
X48042.OICT	32070	33070	SEQUENTIAL	00001	ISR001	00001	NO
X46206.DEPRDC	31070	33570	SEQUENTIAL	00001	ISR001	00001	NO
X46206.MN30C	32070	33570	SEQUENTIAL	00001	ISR001	00001	NO
X46206.DEPRDA	31070	33570	SEQUENTIAL	00001	ISR001	00001	NO
X46206.ERS10C	31370	33570	SEQUENTIAL	00002	ISR001	00001	NO
X46206.ERS10A	31370	33570	SEQUENTIAL	00004	ISR001	00001	NO
X46206.FD170T	32370	33570	SEQUENTIAL	00007	ISR001	00001	NO
X48042.DATA	32070	33070	SEQUENTIAL	00001	ISR001	00001	NO
X46206.SK2E1DC	32170	33070	SEQUENTIAL	00001	ISR001	00001	NO
X46206.SK2E1DA	32170	33070	SEQUENTIAL	00001	ISR001	00001	NO
X46206.MN30T	32070	33570	SEQUENTIAL	00003	ISR001	00001	NO
X48070.OIC4	32470	33870	NOT DEFINED	00001	ISR001	00001	NO
X7350.PARTY	13470	35070	INDEXED SEQ	00003	ISR001	00001	NO
X48436CU	19770	36599	PARTITIONED	00007	ISR001	00001	NO
X46206.FDSDT	32470	33570	SEQUENTIAL	00001	ISR001	00001	NO
X48070.DAT4	32470	33870	SEQUENTIAL	00001	ISR001	00001	NO
X48026.TESTIS	18370	30097	INDEXED SEQ	00002	ISR001	00001	NO

THERE ARE 0113 EMPTY CYLINDERS PLUS 0166 EMPTY TRACKS ON THIS VOLUME
THERE ARE 0160 BLANK DSCRS IN THE VTOC ON THIS VOLUME

VTOC Listing by Program IEHLIST

Handout 7a

ISRD01 TABLE OF CONTENTS FOR VOLUME "ISRD01" 11/23/70 DAY=327 16:51 PAGE 1

DEVICE DESCRIPTION: TYPE=2314 DISK PACK NDCYLS=203 TRKS/CYL=20 TRKSIZE=7294 DSCB/TRK=25 PDS/TRK=17
 VTDC DESCRIPTION: NUM DSCBS=200 AVAIL DSCBS=160 VTDC EXTENT=0000.0002-0000.0009 NUMALT=60 NEXTALT=00C8.0000
 AVAILABLE SPACE: 2426 TRACKS IN 13 EXTENTS, INCLUDING 113 FULL CYLINDERS.

-----DSNAME-----	SERIAL	SEQ	CREDI	EXPI	QSD	RECEM	BLKSZ	LRECL	KEY	OP	IRKAL	IRKUS	EX	SECQU	I
DATA.FD35	ISRD01	0001	11/20/70	12/31/70	PS	FB	3526	86	0	00	11	11	1	22	B
DSIRISUB	ISRD01	0001	05/11/70	11/29/99	PD	U	7294	0	0	00	50	30	1	5	T
DSIRIS2	ISRD01	0001	05/11/70	11/29/99	PD	U	7294	0	0	00	720	706	7	5	T
X453948G.FTL	ISRD01	0001	05/12/70	12/31/99	PD	U	7294	0	0	00	65	56	4	5	T
X45758.CAR763	ISRD01	0031	10/26/70	12/31/70	PS	FB	3476	79	0	00	48	48	13	2	T
X46206.BIAS100A	ISRD01	0001	10/28/70	12/01/70	PS	FB	3458	247	0	00	9	9	1	10	B
X46206.BIAS100C	ISRD01	0001	10/28/70	12/01/70	PS	FB	1600	80	0	00	4	4	4	2	B
X46206.BIAS130A	ISRD01	0001	11/23/70	12/16/70			0	0	0	00	14	1	1	10	B
X46206.BIAS130C	ISRD01	0001	11/23/70	12/16/70	PS	FB	1600	80	0	00	3	3	3	2	B
X46206.DEPDA	ISRD01	0001	11/06/70	12/01/70	PS	FB	3507	21	0	00	4	4	1	2	B
X46206.DEPDCA	ISRD01	0001	11/06/70	12/01/70	PS	FB	1600	80	0	00	1	1	1	1	B
X46206.ERSIDA	ISRD01	0001	11/09/70	12/01/70	PS	FB	3392	212	0	00	55	55	4	50	B
X46206.ERSIDC	ISRD01	0001	11/09/70	12/01/70	PS	FB	1600	80	0	00	2	2	2	5	B
X46206.FDSDT	ISRD01	0001	11/20/70	12/01/70	PS	FB	3526	86	0	00	40	40	1	9	B
X46206.FD17DC	ISRD01	0001	11/19/70	12/01/70	PS	FB	1600	80	0	00	1	1	1	1	B
X46206.FD17DT	ISRD01	0001	11/19/70	12/01/70	PS	FB	3526	86	0	00	28	28	7	5	B
X46206.MN3DC	ISRD01	0001	11/16/70	12/01/70	PS	FB	1600	80	0	00	1	1	1	1	B
X46206.MN3DT	ISRD01	0001	11/16/70	12/01/70	PS	FB	3570	34	0	00	6	6	3	3	B
X46206.SK0000C	ISRD01	0001	11/23/70	12/01/70	PS	FB	3444	123	0	00	5	5	1	9	B
X46206.SK2E1DA	ISRD01	0001	11/17/70	11/26/70	PS	FB	3500	70	0	00	11	11	1	5	B
X46206.SK2E1DC	ISRD01	0001	11/17/70	11/26/70	PS	FB	1600	80	0	00	1	1	1	1	B
X48026.TEST1S	ISRD01	0001	07/02/70	10/27/97	IS	FB	360	360	9	2A	40		2	0	A
X48042.QUATA	ISRD01	0001	11/16/70	11/26/70	PS	FB	3222	537	0	00	115	115	1	20	T
X48042.QUACT	ISRD01	0001	11/16/70	11/26/70	PS	FB	1600	80	0	00	6	6	1	2	T
X48070.QUAT4	ISRD01	0001	11/20/70	12/04/70	PS	VB	3520	3516	0	00	2	1	1	1	T
X48070.QUIC4	ISRD01	0001	11/20/70	12/04/70			0	0	0	00	1	1	1	1	T
X48436CU	ISRD01	0001	07/16/70	12/31/99	PD	U	7294	80	0	00	80	76	7	10	C
X49513OR	ISRD01	0001	11/07/70	11/29/99	PD	U	7294	0	0	00	150	121	1	10	T
X7350.PARTY	ISRD01	0001	05/14/70	12/16/70	IS	F	87	87	12	10	80		3	0	A

<<<<< END VTDC 29 DATA SETS >>>>>

VTDC Listing by Program SPVTOC

Handout 7b

GENERAL INFORMATION FOR CATALOG ON VOL ISRES4 DS OR INDEX NAME	ENTRY TYPE	VOL. ID.	SEQ. NO.	DEV. TYPE	ALIAS OF
ALCINT	DATA SFT	ISRD02	000000	30002008	
ALLN	DATA SFT	ISRD02	000000	30002008	
CAUGHT1	DATA SET	1493	000001	30008001	
CAUGHT2	DATA SET	1493	000002	30008001	
CHINA	DATA SET	5031	000002	30008001	
CHINAC	DATA SET	5031	000001	30008001	
CSFRDATA	DATA SET	ISRES4	000000	30002008	
CSFRDICT	DATA SET	ISRES4	000000	30002008	
CSFTDATA	DATA SET	ISRES4	000000	30002008	
CSFTDICT	DATA SET	ISRES4	000000	30002008	
DACORAEA	DATA SET	2625	000003	30008001	
DACORAE8	DATA SET	1585	000004	30008001	
DAFA1A8	DATA SET	1744	000002	30008001	
DAFAM800	DATA SET	2928	000002	30008001	
DAFAP00L	DATA SET	1706	000002	30008001	
DAFR00LC	DATA SET	2413	000002	30008001	
DAMIC013	DATA SET	765	000004	30008001	
DAMIC014	DATA SET	764	000006	30008001	
DAMIC015	DATA SET	765	000006	30008001	
DAMIC016	DATA SET	764	000007	30008001	
DAMIC017	DATA SET	765	000008	30008001	
DAMIC018	DATA SET	764	000003	30008001	
DAMMPAFA	DATA SET	2591	000005	30008001	
DAMMPAER	DATA SET	2590	000006	30008001	
DATAAUG1	DATA SET	1583	000002	30008001	
DATAMPF	DATA SET	1968	000003	30008001	
DATERPF	DATA SET	2590	000006	30008001	
DATASYSC	DATA SET	ISRES4	000000	30002008	
DATZ044	DATA SET	2783	000004	30008001	
DATZ046	DATA SET	2889	000002	30008001	
DATZ7M	DATA SET	492	000008	30008001	
DATZ74M	DATA SET	492	000002	30008001	
DATZ75M	DATA SET	492	000004	30008001	
DATZ76M	DATA SET	492	000006	30008001	
DATFAM69	DATA SET	2674	000002	30008001	
DAT69FAM	DATA SET	2446	000002	30008001	
DAT70FAM	DATA SET	2807	000002	30008001	
DAT9	DATA SET	5034	000002	30008001	
DARR9FAM	DATA SET	2774	000002	30008001	
D469FAM	DATA SET	2548	000002	30008001	
DA70FAM	DATA SET	2916	000002	30008001	
DA70FAML	DATA SET	2507	000002	30008001	
DA70FINL	DATA SET	1605	000002	30008001	
DETAFL	DATA SET	GUARDN	000000	30002008	
DICFAM68	DATA SET	2874	000001	30008001	
DICORAEA	DATA SET	2625	000002	30008001	
DICORAE8	DATA SET	1585	000003	30008001	
DICTAIG1	DATA SET	1583	000001	30008001	
DICTMPPF	DATA SET	1968	000002	30008001	
DICTPERF	DATA SFT	2590	000005	30008001	
DICTSYSC	DATA SET	ISRES4	000000	30002008	
DICTZ044	DATA SET	2783	000003	30008001	
DICTZ046	DATA SET	2889	000001	30008001	

Listing of Catalog

Handout 8

Handout 9

DD Card Error Examples (DISP and DSN)

```

//E016644 JOB (008,
// 48070,SCR1),RATTENBURY
***
*** EXAMPLES OF ERRORS WITH DISP AND DSNAM=PARAMETERS HANDOUT 9
***
// EXEC PGM=IEFBRI4
***
//A DD DISP=(OLD,PASS),DSNAME=S6DATA
***
//B DD DISP=OLD,PASS,DSN=BC
IEF632I FORMAT ERROR IN THE DISP FIELD
***
//C DD DISP=(OLD),DSNAME=DATA
***
//D DD DISP=OLD,DSNAME=49999.DICT
IEF647I NON-ALPHABETIC FIRST CHARACTER OF NAME IN THE DSNAM=FIELD
***
//E DD DISP=(NEW,CATALOG),DSN=S123
IEF643I UNIDENTIFIED POSITIONAL PARAMETER IN THE DISP FIELD
***
//F DD DISP=(MOD,KEEP),DSN=ABCDEFGHI
IEF642I EXCESSIVE PARAMETER LENGTH IN THE DSNAM=FIELD
***
//G DD DISP=(NEW,PASS),DSNAME=S601:DICT
IEF623I SOURCE TEXT CONTAINS UNDEFINED OR ILLEGAL CHARACTERS IN THE DSNAM=FIELD
***
//H DD DISS=(,KEEP),DSNAME='S601,DICT'
IEF630I UNIDENTIFIED KEYWORD ON THE DD STATEMENT

```

IEF272I - STEP WAS NOT EXECUTED.

Handout 10

Exercises on DD Cards for Input Files

Suppose a program is being used which requires some data to be input; according to the user writeup for this program, the details of the data must be specified on a DD card with DDname INDATA.

Give the DD cards which would be used for the following data files:

- (i) File ABCD on the 3rd file of a standard labelled tape, tape number 1000.
- (ii) File DATA.AL234 on the first file of a standard labelled tape 9999.
- (iii) Data is on the second file of unlabelled tape 12345 (record characteristics have been set in the program).
- (iv) File STUDYCEN which was catalogued when it was created on the ninth file of standard labelled tape 45.
- (v) File X40000A on public disk ISRP03.
- (vi) STUDY6 on private disk USER6 which is to be catalogued for future use.
- (vii) Use STUDY6 after it has been catalogued.

Handout 11

Exercises on DD Cards for Output Files .

Procedure PA004 is to be used to copy about 5000 cards to tape and disk files. The ddname for the DD card describing the output file is DATAOUT. The procedure uses program IEBGENER which does not provide DCB information for the output file since this varies from run to run.

Write DD cards to specify the output files described below:

- (i) File 2 of standard labelled tape number 900 with a name CARDATA.
Put three card images in one block and catalog the data set.

- (ii) File 1 of unlabelled tape number 90000, unblocked.

- (iii) File 8 of a standard labelled tape, unblocked with file name RAWS60.

- (iv) 10 cards to a block on disk ISRA with name X499999D to keep for one week.

- (v) File name S60 to a user disk (USER20) blocked so as to use as little space as possible. No expiration date.

- (vi) Temporary file to be passed to the next step and deleted at the end of the job
 - (i) Put it on disk
 - (ii) Put it on tape

Handout 12

Output from Executing Procedure SCRAT

```

//E016137 JOB (008,
// 48070,SCR1),RATTENBURY
*** EXAMPLE OF A CATALOGUED PROCEDURE HANDBOUT 12
***
// EXEC SCRAT
XX PROC VOL=ISR02 00000010
XXGO EXEC PGM=IEHPRGM 00000020
XXSYSPRINT DD SYSOUT=A 00000030
XXDD1 DD UNIT=SYSDA,VOL=SER=ISRES4,DISP=OLD 00000040
XXDD2 DD UNIT=SYSDA,VOL=SER=ISRES5,DISP=OLD 00000050
XXDD3 DD UNIT=SYSDA,VOL=SER=ISR01,DISP=OLD 00000060
XXDD4 DD UNIT=SYSDA,VOL=SER=&VOL,DISP=OLD 00000070
IEF653I SUBSTITUTION JCL - UNIT=SYSDA,VOL=SER=ISR02,DISP=OLD
//SYSIN DD *
X/SYSIN DD DSN=SYS1.PARMLIB(SCRATCH),DISP=OLD 00000080

```

SYSTEM SUPPORT UTILITIES ----- IEHPRGM

Page 0001

```

SCRATCH DSNAME=A,VOL=2314=ISR01,PURGE
IEH207I STATUS OF USERS REQUEST TO SCRATCH DATA SET A
      VOLUME I.D. ACTION TAKEN REASON FOR TAKING THIS ACTION ERROR
      ISR01 None DATA SET OR MEMBER NOT FOUND ***
END OF ERROR ANALYSIS LISTING ... UNUSUAL END

```

UTILITY END

Listing of Four Catalogued Procedures

PAGE 001

MEMBER NAME	SCRAT		
	//	PROC	VOL=ISRB
	//GO	EXEC	PGM=IEHPRGM
	//SYS	PRINT	DD SYSOUT=A
	//DD1	DD	UNIT=SYSDA,VOL=SER=ISRES5,DISP=OLD
	//DD2	DD	UNIT=SYSDA,VOL=SER=ISRES5,DISP=OLD
	//DD3	DD	UNIT=SYSDA,VOL=SER=ISRA,DISP=OLD
	//DD4	DD	UNIT=SYSDA,VOL=SER=&VOL,DISP=OLD
	//SYS	SIN	DD DSN=SYS1.PARMLIB(SCRATCH),DISP=OLD
			00000010
			00000020
			00000030
			00000040
			00000050
			00000060
			00000070
			00000080
MEMBER NAME	COPY		
	//GO	EXEC	PGM=IEBGENER
	//SYS	PRINT	DD SYSOUT=A
	//SYS	UT1	DD DDNAME=DATAIN
	//SYS	UT2	DD DDNAME=DATAOUT
	//SYS	SIN	DD DUMMY
			00000010
			00000020
			00000030
			00000040
			00000050
			00000050
MEMBER NAME	LABPRT		
	//LAB	PRT	PROC FILE=1,LIB=OSIRPGM
	//GO	EXEC	PGM=LABPRT
	//STE	PLIB	DD DSN=&LIB,DISP=SHR
	//FT0	6FOO1	DD SYSOUT=A
	//FT0	8FOO1	DD VOL=SER=&TAPE,LABEL=(&FILE,BLP,,IN),
	//		DISP=(OLD,PASS),UNIT=TAPE
			00000010
			00000020
			00000030
			00000040
			00000050
			00000050
			00000060
MEMBER NAME	FORTGCL		
	//FORT	EXEC	PGM=IEYFORT
	//SYS	PRINT	DD SYSOUT=A
	//SYS	PUNCH	DD SYSOUT=B
	//SYS	LIN	DD DSNAME=&LOADSET,DISP=(MOD,PASS),UNIT=SYSSQ,
	//		SPACE=(80,(200,100),RLSE),DCB=BLKSIZE=80
	//LKED	EXEC	PGM=IEWL,PARM=(XREF,LET,LIST),COND=(4,LT,FORT)
	//SYS	LIB	DD DSNAME=SYS1.FORTLIB,DISP=SHR
	//SYS	LMOD	DD DSNAME=&GOSET(MAIN),DISP=(NEW,PASS),UNIT=SYSDA,
	//		SPACE=(1024,(20,10,1),RLSE),DCB=BLKSIZE=1024
	//SYS	PRINT	DD SYSOUT=A
	//SYS	UT1	DD UNIT=SYSDA,SPACE=(1024,(100,10),RLSE),
	//		DCB=BLKSIZE=1024,DSNAME=&SYSUT1
	//SYS	LIN	DD DSNAME=&LOADSET,DISP=(OLD,DELETE)
	//		DD DDNAME=SYSIN
			00000010
			00000020
			00000030
			00000040
			00000050
			00000060
			00000070
			00000080
			00000090
			00000100
			00000110
			00000120
			00000130
			00000140

Handout 13

(page 2)

```
MEMBER NAME    COSOME
//COSOME      PROC
//GOA        EXEC    PGM=IEBGENER
//SYSPRINT   DD     SYSOUT=A
//SYSIN      DD     DUMMY
//SYSUT2     DD     UNIT=TAPE,DISP=(NEW,PASS),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=1600)
//GOB        EXEC    PGM=IERRCOoo
//SYSOUT     DD     SYSOUT=A
//SORTWK01   DD     UNIT=SYSDA,SPACE=(TRK,(400),,CONTIG)
//SORTWK02   DD     UNIT=SYSDA,SPACE=(TRK,(400),,CONTIG)
//SORTWK03   DD     UNIT=SYSDA,SPACE=(TRK,(400),,CONTIG)
//SORTWK04   DD     UNIT=SYSDA,SPACE=(TRK,(400),,CONTIG)
//SORTIN     DD     DSN=*.GOA.SYSUT2,DISP=(OLD,KEEP)
//SORTOUT    DD     UNIT=SYSDA,SPACE=(TRK,(200,20)),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=1600),DISP=(NEW,PASS)
//GOC        DD     PGM=ZMERCH
//FT01F001   DD     DDNAME=SYSIN
//FT06F001   DD     SYSOUT=A
//DATAIN     DD     DSN=*.GOB.SORTOUT,DISP=(OLD,DELETE)
//DATAOUT    DD     UNIT=TAPE,DISP=(NEW,KEEP)
```

Handout 14a

Allocation Messages

```

//F016313 JOB (008,
// . 48070,SCR1),RATTENBURY
***
*** ALLOCATION AND DEALLOCATION MESSAGES HANDOUT 14
***
// EXEC PC210
XX PROC LIB=ISRLIB 00000010
XXGO EXEC PGM=PC210P 00000020
XXSTEPLIB DD DSNAME=&LIB,DISP=OLD 00000030
IEF653I SUBSTITUTION JCL - DSNAME=ISRLIB,DISP=OLD
XXFT47F001 DD DSN=SYS1.DATA,DISP=OLD,DCB=(RECFM=F,LRECL=80,BLKSIZE=80) 00000031
XXFT48F001 DD DSN=SYS1.DICT,DISP=OLD,DCB=(RECFM=F,LRECL=80,BLKSIZE=80) 00000032
XXFT01F001 DD DSN=SYS1.SETUP,DISP=OLD, 00000040
XX DCB=(RECFM=F,LRECL=80,BLKSIZE=80) 00000050
XXFT06F001 DD SYSOUT=A 00000060
XXISR10 DD UNIT=DISK,SPACE=(CYL,(20)) 00000070
XXFT49F001 DD DDNAME=SYSIN 00000080
XXFT50F001 DD DSN=X48070.NEWS,DISP=OLD,LABEL=(,,IN) 00000090
//DICTIN DD DSN=ISRDICT,DISP=(OLD,DELETE)
X/DICTIN DD DSN=SYS1.DICT,DISP=OLD 00000110
//DATAIN DD DSN=CHINA,DISP=(OLD,CATLG),UNIT=TAPE,VOL=SER=SCR1
X/DATAIN DD DSN=SYS1.DATA,DISP=OLD 00000120
//DATAOUT DD DSN=WORKA,UNIT=SYSDA,VOL=SER=ISR001,DISP=(NEW,PASS),
// SPACE=(TRK,(10))
//SYSIN DD *

```

```

IEF236I ALLOC. FOR E016313 GO
IEF237I 131 ALLOCATED TO STEPLIB
IEF237I 130 ALLOCATED TO FT47F001
IEF237I 130 ALLOCATED TO FT48F001
IEF237I 130 ALLOCATED TO FT01F001
IEF237I 130 ALLOCATED TO ISR10
IEF237I 00A ALLOCATED TO FT49F001
IEF237I 131 ALLOCATED TO FT50F001
IEF237I 130 ALLOCATED TO DICTIN
IEF237I 281 ALLOCATED TO DATAIN
IEF237I 132 ALLOCATED TO DATAOUT

```

Handout 14b

Deallocation Messages

IEF285I	ISRLIB	KEPT
IEF285I	VOL SER NOS= ISRES5.	
IEF285I	SYS1.DATA	KEPT
IEF285I	VOL SER NOS= ISRES4.	
IEF285I	SYS1.DICT	KEPT
IEF285I	VOL SER NOS= ISRES4.	
IEF285I	SYS1.SETUP	KEPT
IEF285I	VOL SER NOS= ISRES4.	
IEF285I	SYSOUT	SYSOUT
IEF285I	VOL SER NOS=	
IEF285I	SYS70328.T230830.RP015.E016313.R0000051	DELETED
IEF285I	VOL SER NOS= ISRES4.	
IEF285I	X48070.NEWS	KEPT
IEF285I	VOL SER NOS= ISRES5.	
IEF283I	ISRDICT	NOT DELETED 8
IEF283I	VOL SER NOS= ISRES4 3.	
IEF287I	CHINA	NOT CATLGD 2
IEF287I	VOL SER NOS= SCR1 .	
IEF285I	WORKA	PASSED
IEF285I	VOL SER NOS= ISRDO1.	
IEF280E	K 281,SCR1 ,E016313,GO	

Handout 15

Examples of Good and Bad DD Cards

```

//E016123 JOB (008,
// 48070,SCR1),RATTENBURY
***
*** EXAMPLES OF GOOD AND BAD DD CARDS.
***
// EXEC PGM=IEFBRL4
//DD01 DD DSN=A,VOL=SER=9,DISP=OLD
//DD02 DD DSN=B,UNIT=TAPE,DISP=OLD,
//DD03 DD DSN=ABCDEFGH1,UNIT=TAPE,DISP=OLD,VOL=SER=99
IEF642I EXCESSIVE PARAMETER LENGTH IN THE DSNAME FIELD
//DD04 DD DSN=C,UNIT=TAPE,LABEL=NL,DCB=BLKSIZE=80,VOL=SER=99
IEF644I INVALID NUMERIC IN THE LABEL FIELD
//DD05 DD DCB=(BLKSIZE=900,LRECL=80,RECFM=FB),DSN=D,DISP=OLD
//DD06 DD UNIT=DISK,DSN=E,VOL=SER=ISRD01,LABEL=EXPDT=70100
//DD07 DD UNIT=DISK,DSN=G,DISP=(OLD,PASS),VOL=SER=(PRIVATE,RETAIN,
// SER=D12)
IEF622I UNBALANCED PARENTHESIS IN THE SER SUBPARAMETER OF THE VOLUME FIELD
//DD08 DD UNIT=TAPE,VOL=SER=SCR1
//DD09 DD UNIT=TAPE,VOL=SER=SCR1,DCB=BLKSIZE=80,RECFM=F,LRECL=80,
IEF630I UNIDENTIFIED KEYWORD IN THE BLKSIZE SUBPARAMETER OF THE DCB FIELD
// DISP=(NEW,PASS)
IEF605I UNIDENTIFIED OPERATION FIELD
//DD10 DD UNIT=DISK,VOL=(PRIVATE,RETAIN,SER=DISK4),SPACE=(3600,(100,20)),
IEF618I OPERAND FIELD DOES NOT TERMINATE IN COMMA OR BLANK
/* DSN=ADATA
//DD11 DD UNIT=SYSDA,SPACE=(TRACK,(20))
IEF605I INVALID NUMERIC IN THE SPACE FIELD
//DD12 DD UNIT=TAPE,LABEL=6,VOL=SER=1000,DISP=(OLD)
//DD13 DD UNIT=TAPE,VOL=SER=9999,DSN=ABC,DISP=(NEW,PASS)
IEF630I UNIDENTIFIED KEYWORD ON THE DD STATEMENT
//DD14 UNIT=DISC,VOL=(PRIVATE,RETAIN,SER=A1),DISP=OLD,DSN=M
IEF605I UNIDENTIFIED OPERATION FIELD
//DD15 DD UNIT=DISC,VOL=(PRIVATE,RETAIN,SER=A1),DISP=OLD,DSN=M

IEF272I - STEP WAS NOT EXECUTED.

```

Handout 16

Examples of Allocation Errors Caused by JCL Errors

```
//F001794 JOB (008,
// 480700,SCR1),RATTENBURY
// EXEC DEBE
IEF612I PROCEDURE NOT FOUND
```

```
//F001797 JOB (008,
// 480700,SCR1),RATTENBURY
// EXEC COPY
XXGO EXEC PGM=IEBGENER 00000010
XXSYSPRINT DD SYSOUT=A 00000020
XXSYSIN DD DUMMY 00000030
XXSYSUT1 DD DDNAME=DATAIN 00000040
XXSYSUT2 DD DDNAME=DATAOUT 00000050
//GO.DATAOUT DD UNIT=SYSDA,VOL=SER=ISR001,DISP=(NEW,KEEP),
// LABEL=EXPDT=70300,DSN=X4999D
//GO.DATAIN DD *
```

```
IEF256I E033357 GO SYSUT2 DIRECT ACCESS-ABSOLUTE TRACK NOT AVAILABLE
IEF272I - STEP WAS NOT EXECUTED.
```

```
//F)1798 JOB (008,
// 480700,SCR1),RATTENBURY
// EXEC COPY
XXGO EXEC PGM=IEBGENER 00000010
XXSYSPRINT DD SYSOUT=A 00000020
XXSYSIN DD DUMMY 00000030
XXSYSUT1 DD DDNAME=DATAIN 00000040
XXSYSUT2 DD DDNAME=DATAOUT 00000050
//GO.DATAOUT DD DSN=X48070JR,DISP=(NEW,KEEP),VOL=SER=ISRES5,
// UNIT=SYSDA,SPACE=(TRK,(5))
//GO.DATAIN DD *
```

```
IEF253I F033358 GO SYSUT2 DIRECT ACCESS-DUPLICATE NAME ON VOLUME
IEF272I - STEP WAS NOT EXECUTED.
```

```
//F001799 JOB (015,
// 48070,SCR1),RATTENBURY
// EXEC COPY
XXGO EXEC PGM=IEBGENER 00000010
XXSYSPRINT DD SYSOUT=A 00000020
XXSYSIN DD DUMMY 00000030
XXSYSUT1 DD DDNAME=DATAIN 00000040
//GO.SYSUT2 DD UNIT=DISK,VOL=SER=ISR001,DISP=OLD,DSN=X9907A
X/SYSUT2 DD DDNAME=DATAOUT 00000050
//GO.DATAIN DD *
```

```
IEF251I F001799 JOB CANCELED
IEF272I - STEP WAS NOT EXECUTED
```

INDEX

- ABEND, 11
- Abnormal termination, 11, 67-69
- Accounting fields (on Job card), 8-10
- Address, 1
- AFF-affinity (as part of UNIT specification of DD card), 41
- Aliases, 58
- Allocation messages (from job scheduler), 64-65
- Arithmetic unit, 1
- AVR (Automatic Volume recognition), 39

- Bit, 1
- BLKSIZE (subparameter of DCB parameter on DD card), 44
- Block, physical record, 17-18
- Block length, 19
- Blocking, 17-18
- BP1 (tape density), 21-22
- Buffer, 19-20, 44
- BUFNO (subparameter of DCB parameter on DD card), 44
- Byte, 1

- Card, punched, 2
- Card punch, 2
 - Specifying on DD card, 51
- Card reader, 2
 - Specifying on DD card, 52
- Cards, 2
- Cataloging, uncataloging
 - data sets, 35
- Catalog, 31-32
- Catalogued procedure, 11, 55-62

- Central processing unit, 19
- Character, 1
- Comments (on JCL cards), 6
- Compiler, 4
- Completion codes, 67-69
- Concatenation, 53
- COND (keyword on EXEC card), 11
- Condition code, 11
- Consol (operators), 3
- CONTIG (subparameter of SPACE parameter on DD cards), 47
- Contiguous disk space, 47
- Continuation (JCL), 6
- Control cards, 5
- Control program, 4
- Control unit, 1
- Core, 18, 20
- CYL (subparameter of SPACE parameter on DD cards), 46
- Cylinder (disk), 28

- DASDI (initializing disks), 29
- Data, 16
 - Data control block, 43-45
 - Data definition, 15
 - Data set, 17
 - Data set name, 36-37
 - Data set name, default, 36
 - Data set organization, 30-31
 - Data set, temporary, 36
 - DCB (parameter on DD card), 43-45
 - DCB (when to specify), 44-45
 - DD card, 33-54
 - DD cards (general), 15-16, 33-34
 - ddname, 15, 33
 - DD parameters (general), 15-16
 - DD parameters (which to use), 49

- Deallocation messages (from job scheduler), 65
- Deleting data from disk, 30
- Density (tape), 2, 21-22
- Direct access devices, 3
- Directory (for partitioned data sets), 31
- Disk, 3, 27-32
- Disk capacity, 28
- Disk characteristics, 31
- Disk pack, 28
- Disk surfaces, 27-28
- DISP (parameter on DD card), 34-35
- Double buffering, 19-20
- DSN (parameter on DD card), 36-37
- Errors from FORTRAN program, 69
- Errors in allocation & deallocation, 66-67
- Errors (in disk space allocation), 47-48
- Errors (in JCL), 66-67, 69-71
- Errors (input/output), 67-68
- Errors (on DD cards), 38, 47-48, 69-70
- Errors (on EXEC cards), 13
- Errors (on Job cards), 9-10
- Errors, types, 65-67
- EXEC card, 10-12
- Expiration date, 25,40
- Field width, 17
- Fields (in data), 17
- File, 17
- File number (specification on DD card), 38, 39-40
- Files (tape), 24-25
- Fixed length records, 17, 44
- Flowchart symbols (for system flowcharts), 58-59
- FORTRAN error messages, 69
- Hardware, 3
- HASP, 4
- Input devices, 1
- Input media, 2-3
- Input units, 1, 2-3
- Instructions, 1, 3
- Interblock gap, 18, 22
- JCL comments, 6
- JCL continuation cards, 6
- JCL names, 5
- JCL operators, 5
- JCL parameters, 5
- JCL punching conventions, 6
- JCL statement--general format, 5
- Job, 5, 7
- Job card, 7
- Job cards used at the ISR, 8-9
- Job control language, 4-5
- JOBLIB (DD card), 52-54
- Jobname, 7
- Job scheduler, 63-64
- Job step, 5
- K, 2
- Keyword parameter, 5
- Label checking, 23-24, 36-37
- LABEL (parameter on DD card), 38-40
- Label types, 22-23, 39-40
- Labels (tape), 22-24
- Library, 31
- Line printer, 2
- Log (operator's), 3
- Logical record, 17, 44
- LRECL (subparameter of DCB parameter on DD card), 44
- Magnetic tape, 2-3, 21-25
- Member (of partitioned data set), 31
- Memory, 1-2
- Messages (errors)--see errors
- Messages (on printout), 64-65
- MSGLEVEL (on Job card), 7
- Multifile tape, 24
- Names (JCL), 5
- Operating system, 4
- OS, 4, 63
- Output, printed, 2
- Output units/devices, 1
- Parameters (JCL), 5
- Parity, 21
- PARM (keyword on EXEC card), 12
- Partitioned data sets, 31
 - PDS directory, 31

- PGM (keyword on EXEC card), 10-11
- Positional parameters, 5
- Print chain, 52
- Printer, 2
 - specifying on DD card, 51
- Private disk, special disk, 31, 42
- Procedure name (on EXEC card), 11.
- Procedure (see catalogued procedure)
- Program libraries 52-54, 61-62
- Program name (on EXEC card), 11
- Programs, 3
- Public disk, 31, 42, 43
- Punch (see card punch)

- Random access data sets, 31
- Reader (see card reader)
- RECFM (subparameter of DCB
 - parameter on DD card), 44
- Record, 17
- Record length, 17
- Record type, formats, 18-19, 44
- Referring back, 43
- Releasing disk space, 47
- Ring (for tape), 22
- RLSE (subparameter of SPACE
 - parameter on DD card), 47

- Scheduler (see job scheduler)
- Scratching data sets from disk, 55-56
- Setup, 30-31
- Sequential data sets, 30-31
- Software, 3-4
- Space allocation on disk, 30, 46-48
- SPACE (parameter on DD card), 46-48
- Standard labels, 23-24, 38-50
- Step (jobstep), 5, 57
- STEPLIB (DD card), 52-54, 61
- Stepname, 10, 56-57
- Symbolic parameters (in catalogued
 - procedures), 60-62
- System input, 52
- System output, 51
- System parameters (optional,
 - on JOB and EXEC cards), 7, 11-12

- Tape drive, 3
- Tape labels, 22-24
- Tape (magnetic), 2-3, 21-25
- Tape marks, 22
- Tape, runaway, 25
- Tape, work/scratch, 43
- Tracks (disk), 3, 18, 27-28
- Tracks (magnetic tape), 2, 3, 21

- TRK (subparameter of SPACE
 - parameter on DD card), 46
- Typewriter (operator), 3

- UCS parameter, 52
- Uncataloging data sets, 35, 56
- Undefined length records, blocks, 19, 44
- UNIT (parameter on DD card), 40-41
- Units (input and output), 1, 2-3, 40
- User name (on Job card), 7

- Variable length records, 17, 44
- Variables, 16
- VOL (parameter on DD card), 41-43
- Volume label, 23
- VTOC listing, 29-30
- VTOC--volume table of contents
 - (disk), 28-30

- Word, 2
- Write ring, 22
- Writing data on disk, 30

**SURVEY RESEARCH CENTER • INSTITUTE FOR SOCIAL RESEARCH
THE UNIVERSITY OF MICHIGAN**