IBM

System/360

Operating System Facilities

Student Materials

Education Development - Poughkeepsie, New York

IBM

System/360

Operating System Facilities

Student Materials

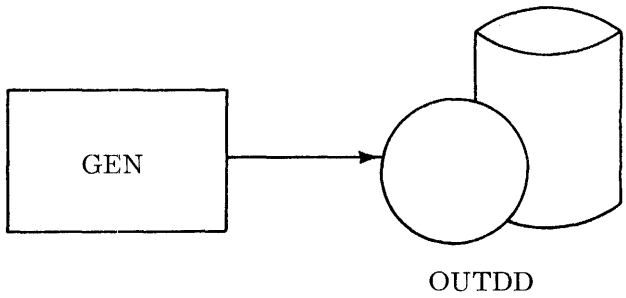
Education Development - Poughkeepsie, New York

Copies of this publication can be obtained through IBM Branch Offices.
Address comments concerning the contents of this publication to:
IBM DPD Education Development, Education Center, Poughkeepsie, New York

OS/360 JCL PROBLEMS

TAPE and DASD OUTPUT JCL PROBLEMS

Write JCL Cards for the following four problems. In each case, a tape or disk file is generated by a program called GEN.



1. Generate a 9 track tape file on volume TP009 - Standard Labels - Expiration date 66360 - Catalog it - Data Set Name is GENFILE.

2. Generate a 9 track tape file on any volume - No Label - Pass to next step - Use temporary name.

3. Generate a disk file on volume 222222 - Keep it - Name is GenFile - Retention period is 30 days - Request space for 400 80byte records - Begin on cylinder boundary - Use 50 for secondary quantity.

4. Generate a disk file on volume 111111 - Allocate 30 cylinders - Delete at end of Job step.

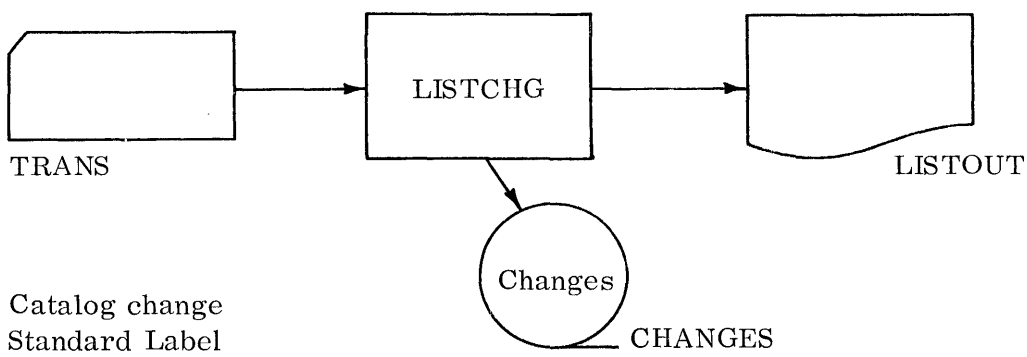
JCL PROBLEMS

Write JCL Cards for the following four problems.

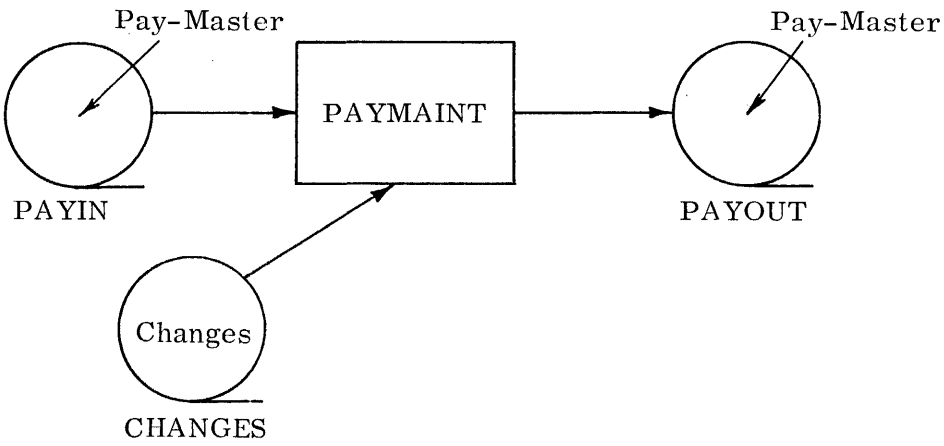
Each problem is a separate job.

DDname is written outside the I/O block -DSname is written inside the I/O Block - Program name is written inside the CPU block -- All programs are stored on SYSI.-USERLIB.

5.

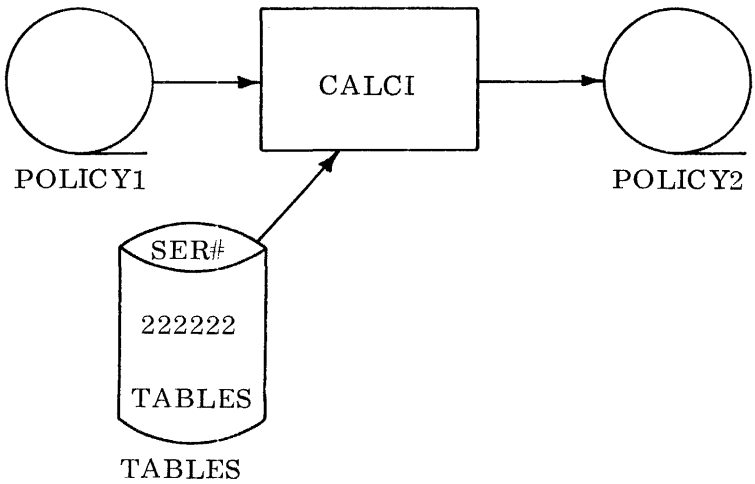


6.



Standard Labels - Changes is cataloged - Pay-Master retention period is 3 days - Pay Master is the name of a generation data group - Delete changes.

7.

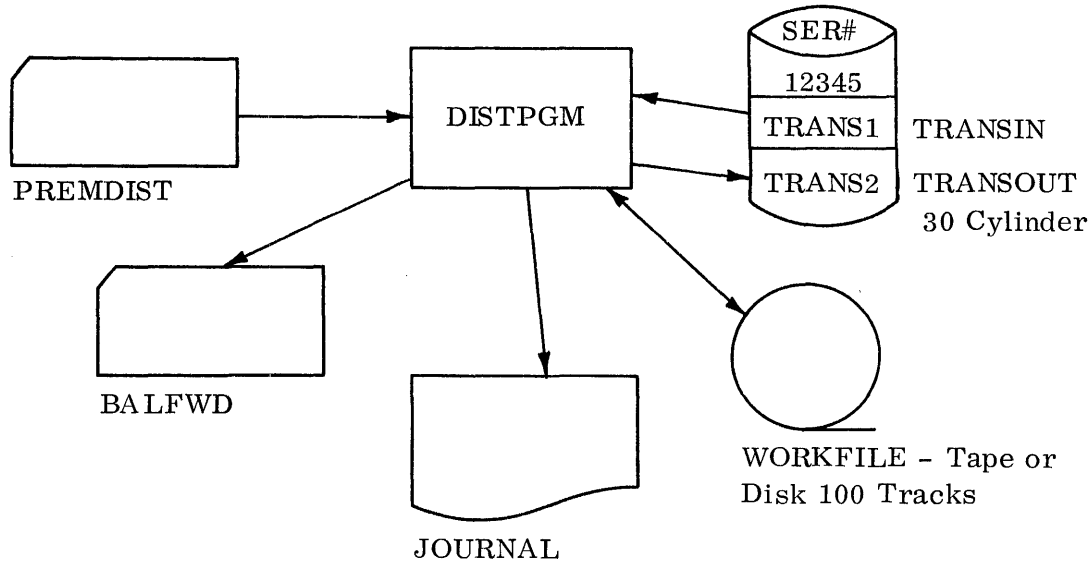


No Labels/Nothing Cataloged

Input has external volume serial no. of PO L007

Tables is located on volume 222222

8.



Standard Labels

TRANS1 and TRANS2 are not cataloged. Keep Both.

9. Given:

1. Job name is PROGA.

2. Account number is 563.

3. Programmer responsible is 'BIG OS'.

4. Control statements in addition to the errors and diagnostic messages are to be written.

5. The program to be executed is PAYROLL and is found in a private library named named PVT. LIB.

6. The input data set to be processed is cataloged by the name DATAIN and is defined on a DD card named INPUT.

7. The output data set is to be cataloged as DATAOUT and is defined on a DD card named OUTPUT.

8. Transaction cards will be read through the job stream. Use a DD card named CARDIN.

9. Delete DATAIN at completion of use.

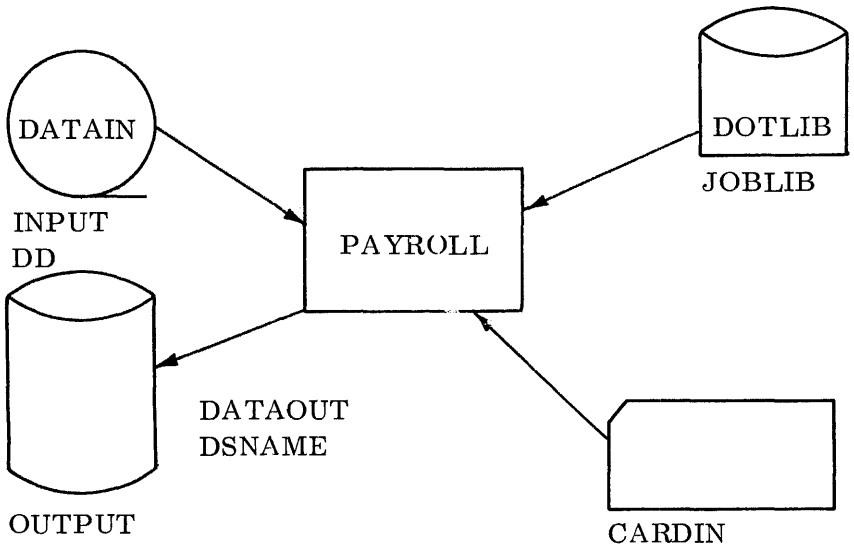
10. DATAIN is to be on tape.

11. DATAOUT is to be on disk and should have 100 tracks allocated to it.

    A. Write the JOB, EXEC, and DD cards to process this job.

    B. Catalog the above procedure with the name CALC. What are the advantages of catalog procedures? Disadvantages?

    C. Call the above cataloged procedure from the library for execution and change the procedure so that DATAOUT is written on tape.



10. Use ASMFCLG to compile and test a program with one data set which is to be printed. Its DD name is ANSWERS. Override the SYSSQ parameters in the cataloged procedure to say UNIT=SYSDA.

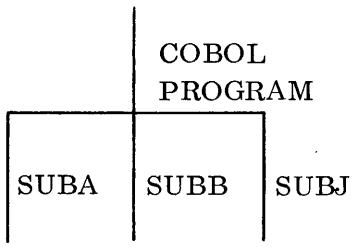
11. Write the JCL and utility control cards to punch a library called SYS1.SAMPLIB.

12. Write the JCL and utility control cards to punch a member named IEAIPL00 from the library called SYS1.SAMPLIB.

LINKAGE EDITOR PROBLEMS

13. Using procedure COBECLG, make the necessary changes and additions to:

    Compile, linkedit, and test a cobol program which calls 3 assembler language subroutines - SUBA, SUBB, SUBJ. The subroutines are on SYS1.USERLIB, SYS1,USERLIB is cataloged. Specify the following overlay structure:



14. Using procedure COBECLG, make the necessary changes and additions to:

    Compile, linkedit, and test a COBOL Program which calls 4 assembler language Subroutines - SUBA, SUBB, SUBC, and SUBJ, SUBA, SUBB, and SUBJ are on SYS1. USERLIB. SUBC is not yet written. Negate the search for SUBC.

IBM

IBM System/360 Assembler Coding Form

PROGRAM	PROBLEM 1 SOLUTION		PUNCHING INSTRUCTIONS		PAGE 1 OF 1
			GRAPHIC		CARD ELECTRO NUMBER
PROGRAMMER		DATE	PUNCH		

STATEMENT				Identification-Sequence
Name (1–8)	Operation (10–14)	Operand (16–71)		(73–80)
//JOCK	JOB	1,NAME,MSGLEVEL=1		
//SI	EXEC	PGM=GEN		
//OUTDD	DD	UNIT=2400,DSNAME=GENFILE,DISP=(CATLG),VOLUME=SER=TP009,	R	
//		LABEL=EXPDT=66360		

IBM

IBM System/360 Assembler Coding Form

PROGRAM	PROBLEM 2 SOLUTION		PUNCHING INSTRUCTIONS		PAGE 1 OF 1
			GRAPHIC		CARD ELECTRO NUMBER
PROGRAMMER		DATE	PUNCH		

STATEMENT				Identification-Sequence
Name (1–8)	Operation (10–14)	Operand (16–71)	Comments	(73–80)
//CROCK	JOB	1,NAME,MSGLEVEL=1		
//SI	EXEC	PGM=GEN		
//OUTDD	DD	UNIT=2400,LABEL=(,NL),DISP=(,PASS),DSNAME=&ACE		

IBM

IBM System/360 Assembler Coding Form

PROGRAM	PROBLEM 3 SOLUTION		PUNCHING INSTRUCTIONS		PAGE 1 OF 1
			GRAPHIC		CARD ELECTRO NUMBER
PROGRAMMER		DATE	PUNCH		

STATEMENT				Identification-Sequence
Name (1–8)	Operation (10–14)	Operand (16–71)	Comments	(73–80)
//FLOCK	JOB	1,NAME,MSGLEVEL=1		
//S1	EXEC	PGM=GEN		
//OUTDD	DD	DSNAME=GENFILE,SPACE=(80,(400,50),,,ROUND),DISP=(,KEEP),S		
//		VOLUME=SER=222222,LABEL=RETPD=30,UNIT=SYSDA		

IBM

IBM System/360 Assembler Coding Form

PROGRAM PROBLEM 4 SOLUTION		PUNCHING INSTRUCTIONS		PAGE 1 OF 1
		GRAPHIC		CARD ELECTRO NUMBER
PROGRAMMER	DATE	PUNCH		

STATEMENT				Identification-Sequence
Name	Operation	Operand	Comments	
1 8	10 14	16 20 25 30 35 40 45 50 55 60 65 71		73 80
//ROOK	JOB	1,NAME,MSGLEVEL=1		
//STEPONE	EXEC	PGM=GEN		
//OUTDD	DD	SPACE=(CYL,30),VOLUME=SER=111111,UNIT=SYSDA		

IBM

IBM System/360 Assembler Coding Form

PROGRAM PROBLEM 5 SOLUTION		PUNCHING INSTRUCTIONS	PAGE 1 OF 1
		GRAPHIC	CARD ELECTRO NUMBER
PROGRAMMER	DATE	PUNCH	

STATEMENT				Identification-Sequence
Name (1–8)	Operation (10–14)	Operand (16–71)	Comments	(73–80)
//JOJO	JOB	1,NAME,MSGLEVEL=1		
//JOBLIB	DD	DISP=OLD,DSNAME=SYS1.USERLIB		
//S1	EXEC	PGM=LISTCHG		
//LISTOUT	DD	SYSOUT=A		
//CHANGES	DD	DISP=(,CATLG),DSNAME=CHANGES,UNIT=2400		
//TRANS	DD	*		
		DECK		
/*				

IBM

IBM System/360 Assembler Coding Form

PROGRAM	PROBLEM 6 SOLUTION		PUNCHING INSTRUCTIONS		PAGE 1 OF 1
			GRAPHIC		CARD ELECTRO NUMBER
PROGRAMMER		DATE	PUNCH		

STATEMENT				Identification-Sequence
Name	Operation	Operand	Comments	
1 8	10 14	16 20 25 30 35 40 45 50 55 60 65 71		73 80
//YUK	JOB	1,NAME,MSGLEVEL=1		
//JOBLIB	DD	DSNAME=SYS1.USERLIB,DISP=(OLD,PASS)		
//S1	EXEC	PGM=PAYMAINT		
//CHANGES	DD	DSNAME=CHANGES,DISP=(OLD,DELETE)		
//PAYIN	DD	DSNAME=PAY.MASTER(0),DISP=OLD		
//PAYOUT	DD	DSNAME=PAY.MASTER(+1),DISP=(,CATLG),UNIT=2400,	X	
//		LABEL=RETPD=3		

IBM IBM System/360 Assembler Coding Form

PROGRAM PROBLEM 7 SOLUTION		PUNCHING INSTRUCTIONS	PAGE 1 OF 1
		GRAPHIC	CARD ELECTRO NUMBER
PROGRAMMER	DATE	PUNCH	

STATEMENT				Identification-Sequence
Name (1–8)	Operation (10–14)	Operand (16–71)	Comments	(73–80)
//YOCK	JOB	1,NAME,MSGLEVEL=1		
//JOBLIB	DD	DSNAME=SYS1.USERLIB,DISP=(OLD,PASS)		
//S1	EXEC	PGM=CALC1		
//POLICY1	DD	LABEL=(,NL),VOLUME=SER=POL007,UNIT=2400,DISP=OLD		
//TABLES	DD	VOLUME=SER=222222,UNIT=SYSDA,DSNAME=TABLES,DISP=OLD		
//POLICY2	DD	LABEL=(,NL),DISP=(,KEEP),UNIT=2400		

IBM

IBM System/360 Assembler Coding Form

PROGRAM	PROBLEM 8 SOLUTION	PUNCHING INSTRUCTIONS	PAGE 1 OF 1
		GRAPHIC	CARD ELECTRO NUMBER
PROGRAMMER	DATE	PUNCH	

```
//YECH    JOB   1,NAME,MSGLEVEL=1
//JOBLIB DD     DSNAME=SYS1.USERLIB,DISP=(OLD,PASS)
//S1      EXEC  PGM=DISTPGM
//BALFWD DD     UNIT=SYSCP
//JOURNAL DD    SYSOUT=A
//WORKFILE DD   UNIT=SYSSQ,SPACE=(TRK,100)
//TRANSOUT DD   VOLUME=SER=12345,DSNAME=TRANS2,UNIT=SYSDA,DISP=(,KEEP),  Z
//              SPACE=(CYL,30)
//TRANSIN DD    VOLUME=SER=12345,DSNAME=TRANS1,UNIT=SYSDA,DISP=OLD
//PREMDIST DD   *


    DATA DECK


/*
```

IBM IBM System/360 Assembler Coding Form

PROGRAM	PROBLEM 9 SOLUTION		PUNCHING INSTRUCTIONS	PAGE 1 OF 2
			GRAPHIC	CARD ELECTRO NUMBER
PROGRAMMER		DATE	PUNCH	

```
A

//PROGA  JOB    563,'BIG OZ',MSGLEVEL=1
//JOBLIB DD     DSNAME=PVT.LIB,DISP=OLD  PVT.LIB IS CATALOGED
//CARDIN DD *

   DATA DECK

/*

B

//JOBBY  JOB
//S1     EXEC   PGM=IEBUPDAT,PARM=NEW
//SYSUT2 DD     DSNAME=SYS1.PROCLIB,DISP=OLD
//SYSPRINT DD   SYSOUT=A
//SYSIN  DD     DATA
./       ADD    CALC,00,0,1
//STEP1  EXEC   PGM=PAYROLL
//INPUT  DD     DSNAME=DATAIN,DISP=(OLD,DELETE)

//OUTPUT DD     DISP=(,CATLG),UNIT=SYSDA,SPACE=(TRK,100),DSNAME=DATAOUT
/*
         ADVANTAGE IS YOU NEED NOT SUPPLY JCL EACH TIME, I.E., FEWER CARDS
         DISADVANTAGE IS IT REQUIRES DISK SPACE
```

STATEMENT — Name (1–8), Operation (10–14), Operand (16–71), Comments; Identification-Sequence (73–80)

IBM

IBM System/360 Assembler Coding Form

PROGRAM	PROBLEM 9 SOLUTION (CONT'D.)	PUNCHING INSTRUCTIONS		PAGE 2 OF 2
		GRAPHIC		CARD ELECTRO NUMBER
PROGRAMMER	DATE	PUNCH		

STATEMENT				Identification-Sequence
Name	Operation	Operand	Comments	
1 8	10 14	16 20 25 30 35 40 45 50 55 60 65 71		73 80
C				
//JOBZ	JOB	563,'BIG OZ',MSGLEVEL=1		
//JOBLIB	DD	DISP=OLD,DSNAME=PVT.LIB		
//ST1	EXEC	CALC		
//STEP1.OUTPUT	DD	UNIT=2400		
//STEP1.CARDIN	DD	*		
	CARD DECK			
/*				

IBM IBM System/360 Assembler Coding Form

PROGRAM PROBLEM 10 SOLUTION		PUNCHING INSTRUCTIONS	PAGE 1 OF 1
		GRAPHIC	CARD ELECTRO NUMBER
PROGRAMMER	DATE	PUNCH	

```
//YUMYUM JOB   1,NAME,MSGLEVEL=1
//STEP1  EXEC  ASMFCLG
//ASM.SYSUT1   DD  UNIT=SYSDA
//ASM.SYSUT2   DD  UNIT=SYSDA
//ASM.SYSUT3   DD  UNIT=SYSDA
//ASM.SYSGO    DD  UNIT=SYSDA
//ASM.SYSIN    DD  *

         SOURCE DECK

/*
//GO.ANSWERS   DD  SYSOUT=A
```

IBM

IBM System/360 Assembler Coding Form

PROGRAM PROBLEM 11 SOLUTION		PUNCHING INSTRUCTIONS		PAGE 1 OF 1
		GRAPHIC		CARD ELECTRO NUMBER
PROGRAMMER	DATE	PUNCH		

STATEMENT

Name	Operation	Operand	Comments	Identification-Sequence
1 8	10 14	16 20 25 30 35 40 45 50 55 60 65 71		73 80
//PUNCHALL	JOB			
//ST1	EXEC	PGM=IEBPTPCH		
//SYSPRINT	DD	SYSOUT=A		
//SYSUT1	DD	DSNAME=SYS1.SAMPLIB,DISP=OLD		
//SYSUT2	DD	UNIT=SYSCP		
//SYSIN	DD	*		
	PUNCH	TYPORG=PO,MAXFLDS=1		
	RECORD	FIELD=(80,1,,1)		
/*				

IBM IBM System/360 Assembler Coding Form

PROGRAM PROBLEM 12 SOLUTION		PUNCHING INSTRUCTIONS	PAGE 1 OF 1
		GRAPHIC	CARD ELECTRO NUMBER
PROGRAMMER	DATE	PUNCH	

STATEMENT				Identification-Sequence
Name (1–8)	Operation (10–14)	Operand (16–71)	Comments	(73–80)
//PUNCHONE	JOB			
//ST EPPE	EXEC PGM=IEBPTPCH			
//SYSPRINT DD	SYSOUT=A			
//SYSUT1 DD	DISP=OLD,DSNAME=SYS1.SAMPLIB			
//SYSUT2 DD	UNIT=SYSCP			
//SYSIN DD *				
PUNCH	TYPORG=PO,MAXFLDS=1,MAXNAME=1			
RECORD	FIELD=(80,1,,1)			
MEMBER	NAME=(IEAIPL00)			
/*				

IBM

IBM System/360 Assembler Coding Form

PROGRAM PROBLEM 13		PUNCHING INSTRUCTIONS	PAGE 1 OF 1
		GRAPHIC	CARD ELECTRO NUMBER
PROGRAMMER	DATE	PUNCH	

STATEMENT				Identification-Sequence
Name (1–8)	Operation (10–14)	Operand (16–71)	Comments	(73–80)
//MUMU	JOB			
//S1	EXEC	COBECLG		
//COB.SYSIN	DD	*		
	COBOL SOURCE DECK			
/*				
//LKED.ZEUS	DD	DSNAME=SYS1.USERLIB,DISP=OLD		
//LKED.SYSIN	DD	*		
	OVERLAY	PT109		
	INCLUDE	ZEUS(SUBA)		
	OVERLAY	PT109		
	INCLUDE	ZEUS(SUBB)		
	OVERLAY	PT109		
	INCLUDE	ZEUS(SUBJ)		
/*				

IBM IBM System/360 Assembler Coding Form

PROGRAM	PROBLEM 14 SOLUTION		PUNCHING INSTRUCTIONS		PAGE 1 OF 1
			GRAPHIC		CARD ELECTRO NUMBER
PROGRAMMER		DATE	PUNCH		

STATEMENT				Identification-Sequence
Name (1–8)	Operation (10–14)	Operand (16–71)	Comments	(73–80)
//JUJU	JOB			
//S1	EXEC	COBECLG		
//COB.SYSIN	DD	*		
	COBOL SOURCE DECK			
/*				
//LKED.THOR	DD	DSNAME=SYS1.USERLIB,DISP=OLD		
//LKED.SYSIN	DD	*		
	INCLUDE	THOR(SUBA,SUBB,SUBJ)		
	LIBRARY	(SUBC)		
/*				

OS/360 JOB CONTROL LANGUAGE REFERENCE CHART

The following chart is provided for use as a reference during preparation of DD Statements.

Does data set follow as part of input stream or take form of a dummy?	* DUMMY DATA DDNAME= ddname	(Data set follows in input stream) (Data set is a dummy) (D/S contains JCL (//) cards (Dummy D/S assumes characteristics of real) Rest of Card Must be Blank
Is data set to be referred to by DSNAME?		NO - The OS assigns a unique name YES - Use one of the forms (DSNAME=) below.
Does DSNAME refer to: 1. Gen. data group 2. PDS 3. Fully qual. name	DSNAME=	{ dsname dsname (element)
Does DSNAME refer to another data set	DSNAME=	{ *.ddname - same step *.stepname. ddname - same job *.stepname. procstep. ddname - same job
Want the OS to make the name unique to this job	DSNAME=	{ &name &name (element) - OS puts the job name in front of the name to make it unique to this job.
Is the DCB to be completed by DD cards		NO - The DCB must be assembled complete by the user or it will be filled in from D/S label. YES - Use one of the forms (DCB=) below.
DCB parameters to come from another data set	DCB=	({ dsname *.ddname *.stepname.ddname *.stepname. procstep. ddname } (Subparameter list of key words that will override those from the other D/S)
DCB parameters to come from subparameter list	DCB=	(Subparameter list of appropriate keywords as defined by each access method's DCB)
Does this D/S want Channel Affinity with -or- Channel Separation from other D/S 's	AFF= SEP=	ddname - User's D/S can share channel with "ddname" (List of ddnames) - User's D/S must be a channel other than "ddnames"
Are volumes to be identified by this DD card?		NO - Volume ID for this D/S must have been passed or cataloged. YES - Use one of the forms (VOLUME=) below.

OS/360 JOB Control Language Reference Chart

Volumes are private and already defined - not to be further defined here.	VOLUME= VOLUME=	SER= (List of volume serial no.'s) REF= { dsname / *.ddname / *.stepname.ddname / *.stepname.procstep.ddname } Volumes Are Private
Volumes are to be further defined here	VOLUME=	([PRIVATE] [RETAIN] [Volume Seq. No.] [Volume Count] ([{SER= (See above) / REF= (See above)}])) Usage Mounting Sequence No. of Vols.
Volume is to remain mounted between jobs.		1. Use a permanent resident device or SYSRES 2. Reserve volume with "MOUNT" command. 3. Use DISP=(NEW,DELETE) and do not specify volume serial number.
Is D/S label info. to be given?		NO - OS assumes standard labels and verifies mounting. YES - If the retention period is not specified, OS assumes 0. Form is shown below.
Positional parameters may be ignored.	LABEL=	[File Seq. No.] [{NL / SL / NSL / SUL}] [{EXPDT = yyddd / RETPD= xxxx}] Sequence Type Time Period
Is the UNIT to be described by this DD card?		NO - The unit description must come from: 1. This D/S is OLD and was PASSED or cataloged, OR 2. This D/S's "VOLUME=" parameter must refer back, OR 3. This D/S uses SPACE parameters "SPLIT" or "SUBALLOC". YES - Use one of the forms (UNIT=) below.
D/S may share device with another D/S - this step only.	UNIT=	AFF= ddname
Output D/S may share pool of tape drives	UNIT=	(POOL, poolname, { 0 / 1 }) Which ddname Add. Units

Define device type, no. of units, mount instructions	UNIT=	Device # / (name, [{N / P}]) Mount [DEFER] DASD Separation [SEP= (List of ddnames)]
Is UNIT a Direct Access device?		NO - Ignore the SPACE, SPLIT, & SUBALLOC parameters. YES - Must be one of the forms below if the D/S is new.
User to handle space allocation by absolute location.	SPACE=	(ABSTR, (Qnty, beginning addr, [directory / quantity]))
OS to handle space allocation automatically	SPACE=	({TRK / CYL / avg.rec.lgth.}, (pri., [sec.], [dir]), [RLSE], [{MXIG / ALX / CONTIG}], [ROUND])
D/S to use split cylinders	SPLIT=	(n, [{CYL / avg.rec.lgth}], [pri. qnty., [sec. qnty.]])
Suballocate space from another D/S.	SUBALLOC=	({TRK / CYL / avg.rec.lgth}, (pri. qnty., [sec. qnty.], [dir. qnty.]), {ddname / stepname.ddname / stepname.procstep.ddname})
Disposition of D/S at EOJ or EOS to be stated on DD card?		NO - OS assumes NEW and DELETE. If the 2nd parameter is blank, no change of status will be made. YES - See below.
Position the device after the last record in the D/S	DISP=	(MOD [{DELETE / KEEP / PASS / CATALG / UNCATALG}], MOD indicates that D/S exists but is to be added to--not read

OS/360 JOB Control Language Reference Chart

Save D/S between job steps	DISP=	({NEW / OLD / MOD} , PASS)	- Last step should use DELETE param. to dispose of D/S. Job assumes DELETE at EOJ regardless.
Save D/S between Jobs.	DISP=	({NEW / OLD / MOD} , {KEEP / CATALG})	- CATALG implies KEEP. D/S will exist until DELETED. KEEP causes the volume to become a private volume.
D/S is to be cataloged or removed from the catalog.	DISP=	({NEW / OLD / MOD}, {CATALG / UNCATLG})	- Catalog structure must exist prior to CATALG. UNCATLG does not DELETE the D/S; it still exists in VTOC. DELETE does uncatalog.
Output D/S to be put onto a SYSOUT class device.	SYSOUT=	(classname, [progname] , [xxxx]	- No (DISP=) needed. D/S will be deleted.

APPENDIX 2

Job-Shop Example Index

1) HOW TO OVERRIDE CATALOGED PROCEDURES

You can override any parameters in a cataloged procedure, except the PGM=progname parameter in the EXEC statement; you can also add parameters or statements not specified in the procedure. When you override or add to a cataloged procedure, the overriding parameters or statements apply only during one execution.

HOW TO OVERRIDE AND ADD DD STATEMENTS

You override or add to a DD statement by using a DD statement whose name is comprised of the procedure step-name qualified by the ddname of the DD statement you are overriding.

There are a few rules that you must follow when overriding or adding a DD statement within a step in a procedure.

- Overriding DD statements must be in the same order in the input stream as they are in the cataloged procedure.
- DD statements to be added must follow overriding DD statements.
- A DD statement with an * in the operand field terminates processing of subsequent DD statements in both the procedure and the input stream.

There are also a few special cases that you should keep in mind when overriding a DD statement.

- All parameters are overriden in their entirety, except the DCB parameter. Within the DCB parameter, you can override only individual subparameters.
- To nullify a keyword parameter (except the DCB parameter), you write, in the overriding DD statement, the keyword and an equal sign followed by a comma. For example, to nullify the use of the UNIT parameter, specify UNIT=, in the overriding DD statement.
- You can nullify a parameter by specifying a mutually exclusive parameter. For example, you can nullify the SPACE parameter by specifying the SPLIT parameter in the overriding DD statement.
- You can nullify the DUMMY parameter by omitting it and specifying the DSNAME parameter in the overriding DD statement.

EXAMPLES OF OVERRIDING AND ADDING DD STATEMENTS

This section contains six examples of overriding and adding DD statements. The DDNAME parameter is not used in these examples, although this parameter is especially useful in cataloged procedures. The use of the DDNAME parameter is described in detail later in this chapter.

Example 1: The following example shows how to override DD statements. If you override more than one DD statement, the overriding DD statements must be in the same order as they are in the cataloged procedure.

The Cataloged Procedure (PROC1)		
//STEP1	EXEC	PGM=progname
//DDNAME1	DD	(any parameters except * or DDNAME)
//DDNAME2	DD	(any parameters except * or DDNAME)

The Input Stream		
//JOB1	JOB	1234,J.DUBON
//STEPA	EXEC	PROC1
//STEP1.DDNAME1	DD	(new parameters replacing those in the procedure)
//STEP1.DDNAME2	DD	(new parameters replacing those in the procedure)

Example 2: The following example shows how to override DD statements and change the last statement to a DD * statement. If you change any DD statement other than the last one to DD*, all processing of DD statements would be terminated. Any subsequent DD statements would be read as part of the data set specified by the DD * statement.

The Cataloged Procedure (PROC1)		
//STEP1	EXEC	PGM=progname
//DDNAME1	DD	(any parameters except * or DDNAME)
//DDNAME2	DD	(any parameters except * or DDNAME)

The Input Stream		
//JOB2	JOB	1234,J.DUBON
//STEPA	EXEC	PROC1
//STEP1.DDNAME1	DD	(new parameters replacing those in the procedure)
//STEP1.DDNAME2	DD	*

Example 3: The following example shows how to add DD statements. If you add more than one DD statement, only the last statement can be a DD * statement.

The Cataloged Procedure (PROC1)		
//STEP1	EXEC	PGM=progname
//DDNAME1	DD	(any parameters except * or DDNAME)
//DDNAME2	DD	(any parameters except * or DDNAME)

The Input Stream		
//JOB3	JOB	1234,J.DUBON
//STEPA	EXEC	PROC1
//STEP1.DDNAME3	DD	(any parameters except *)
//STEP1.DDNAME4	DD	(* or any other parameters)

Example 4: The following example shows how to override DD statements and add DD statements. Overriding statements must be in the same order as they appear in the procedure and must precede those statements being added.

The Cataloged Procedure (PROC1)		
//STEP1	EXEC	PGM=progname
//DDNAME1	DD	(any parameters except * or DDNAME)
//DDNAME2	DD	(any parameters except * or DDNAME)

The Input Stream		
//JOB4	JOB	1234,J.DUBON
//STEPA	EXEC	PROC1
//STEP1.DDNAME1	DD	(new parameters replacing those in the procedure)
//STEP1.DDNAME3	DD	(any parameters except *)
//STEP1.DDNAME4	DD	(* or any other parameters)

Example 5: The following example shows how to concatenate a data set with the data set defined by the last DD statement in the cataloged procedure.

The Cataloged Procedure (PROC1)		
//STEP1	EXEC	PGM=progname
//DDNAME1	DD	(any parameters except * or DDNAME)
//DDNAME2	DD	(any parameters except * or DDNAME)

The Input Stream		
//JOB5	JOB	1234,J.DUBON
//STEPA	EXEC	PROC1
//	DD	(* or parameters defining a data set)

Another way of performing the same function is to use the following cards in the input stream.

The Input Stream		
//JOB5	JOB	1234,J.DUBON
//STEPA	EXEC	PROC1
//STEP1.DDNAME2	DD	(blank operand field)
//	DD	(* or parameters defining a data set)

When you use a blank operand field, no modifications are made to the named DD statement. In the above example, the statement whose name is DDNAME2 could have been overriden; the data set defined by the unnamed DD statement would then be concatenated to the data set that was redefined by overriding.

Example 6: The following example shows how to concatenate a data set not in the input stream with any data set defined in a procedure.

The Cataloged Procedure (PROC1)		
//STEP1	EXEC	PGM=progname
//DDNAME1	DD	(any parameters except * or DDNAME)
//DDNAME2	DD	(any parameters except * or DDNAME)

The Input Stream		
//JOB6	JOB	1234,J.DUBON
//STEPA	EXEC	PROC1
//STEP1.DDNAME1	DD	(blank operand field)
//	DD	(parameters defining a data set)

HOW TO OVERRIDE AND ADD TO EXEC STATEMENTS

You override or add to an EXEC statement in one of two ways:

- Specify, in the operand field of the EXEC statement calling the procedure, the keyword, the procedure step-name, and the parameters. If you are overriding or adding to a multi-step procedure, parameters in the calling EXEC statement must be specified step by step i.e., the parameters for one step must be specified before those of the next step.

- Specify only the keywords and parameters in the operand field of the statement calling the procedure. If you call a multi-step procedure, the specified parameters (with the exception PARM) apply to all steps in the procedure. The PARM keyword overrides the first EXEC statement and nullifies any subsequent PARM keywords. The TIME keyword overrides all TIME keywords in the procedure, and specifies the time allotted for the entire procedure. The COND and ACCT keywords apply to all steps in the procedure.

EXAMPLES OF OVERRIDING AND ADDING TO THE EXEC STATEMENT

This section contains examples of overriding and adding to the EXEC statement.

Example 1: The following example shows how to override a parameter in the EXEC statement. The COND parameter in the procedure specifies that this step is to be bypassed if 200 is greater that the return code of any previous step. The input stream statements override the COND parameter so that the step is to be bypassed if 300 is greater than the return code of any previous step.

The Cataloged Procedure (PROC2)

```
//STEP1   EXEC   PGM=progname, COND=(200,GT)
//DD1     DD     (any parameters except * or DDNAME)
//DD2     DD     (any parameters except * or DDNAME)
```

The Input Stream

```
//JOB1        JOB    1234,J.DUBON
//STEPA       EXEC   PGM=progname
//DDNAME1     DD     (any parameters except * or DDNAME)
//DDNAME2     DD     (any parameters except * or DDNAME)
//STEPB       EXEC   PROC2,COND.STEP1=(300,GT)
```

Example 2: The following example shows how to override a parameter in the EXEC statement, and how to add a new parameter.

The Cataloged Procedure (PROC2)

```
//STEP1   EXEC   PGM=progname,PARM=(NOLOAD,DECK)
//DD1     DD     (any parameters except * or DDNAME)
//DD2     DD     (any parameters except * or DDNAME)
```

The Input Stream

```
//JOB2    JOB    1234,J.DUBON
//STEPA   EXEC   PROC2,PARM.STEP1=(LOAD,NODECK),ACCT=(1234)
```

Example 3: The following example shows how to override individual parameters in each EXEC statement in a multi-step procedure. All overriding parameters for one step of the procedure must be specified before those for the next step are specified.

The Cataloged Procedure (PROC2)

```
//STEP1     EXEC  PGM=progname,ACCT=(1234),PARM=(XREF,LIST)
//DDNAME1   DD    (any parameters except * or DDNAME)
//DDNAME2   DD    (any parameters except * or DDNAME)
//STEP2     EXEC  PGM=progname,ACCT=(1234),COND=(100,LT,STEP1)
//DD1       DD    (any parameters except * or DDNAME)
//DD2       DD    (any parameters except * or DDNAME)
```

The Input Stream

```
//JOB3   JOB   1234,J.DUBON
//STEPA  EXEC  PROC2,PARM.STEP1=,COND.STEP2=(200,LT,STEP1)
```

Example 4: The following example shows how to override parameters in all EXEC statements in a multi-job procedure. The COND keyword applies to all steps in the procedure; the PARM keyword is added to the first step of the procedure, and nullifies subsequent PARM keywords; the ACCT keyword applies to all steps in the procedure.

The Cataloged Procedure (PROC2)

```
//STEP1   EXEC  PGM=progname
//DD1     DD    (any parameters except * and DDNAME)
//DD2     DD    (any parameters except * and DDNAME)
//STEP2   EXEC  PGM=progname,PARM=(XREF,LIST,LET),COND=(5,LT,STEP1)
//DD3     DD    (any parameters except * and DDNAME)
//DD2     DD    (any parameters except * and DDNAME)
//STEP3   EXEC  PGM=progname,COND=((5,LT,STEP1),(5,LT,STEP2))
//DD4     DD    (any parameters except * and DDNAME)
//DD5     DD    (any parameters except * and DDNAME)
//DD2     DD    (any parameters except * and DDNAME)
```

The Input Stream

```
//JOB4   JOB   1234,J.DUBON
//STEPA  EXEC  PROC2,PARM=DECK,COND=(3,LT),ACCT=(12345678,DEPT.123,
                    PROJ.456)
```

2) HOW TO USE THE DDNAME PARAMETER

You use the DDNAME parameter to define a dummy data set that can assume the characteristics of an actual data set, defined by a subsequent DD statement within the step. If a matching DD statement is found, its characteristics, with the exception of its ddname, replace those of the statement using the DDNAME parameter. If a matching DD statement is not found within the step, the statement using the DDNAME parameter remains a dummy.

This section contains examples showing the use of the DDNAME parameter with cataloged procedures.

The rules that you must adhere to when using the DDNAME parameter are as follows:

- You cannot use a backward reference (e.g., *.ddname) to a DD statement referred to be a DDNAME parameter because the statement that you refer to loses its identity.

- You can use a backward reference to a statement containing a DDNAME parameter, but only after the statement to which the DDNAME parameter refers has been encountered. If you use a backward reference before the dummy data set (defined by DDNAME) has been given real characteristics, these real characteristics will not be transferred to the DD statement that contains the backward reference. For example, if you use DCB=*.ddname (where ddname is the name of a statement containing an unresolved DDNAME parameter), the DCB fields that are transferred are blank.

- You can place unnamed DD statements after a statement containing the DDNAME parameter (this indicates concatenation), but you cannot place unnamed DD statements after a statement referred to be a DDNAME parameter.

- You can use the DDNAME parameter five times in a step, but each DDNAME parameter must refer to a different statement.

- You cannot use the DDNAME parameter in a statement named JOBLIB.

When using the DDNAME parameter, you should also keep the following in mind.

- The name of the DD statement that you refer to does not replace the name of the referencing statement.

- If you override a statement that contains the DDNAME parameter, it is nullified.

- If you override with a statement that contains the DDNAME parameter, all parameters in the overridden statement are nullified.

The following DD statements:

```
//S1    EXEC      PGM=progname
//D1    DD          DDNAME=D3
//D2    DD      (parameters X,Y,Z)
//D3    DD      (parameters U,T,V)
```

will result in the same data definition produced by the following statements.

```
//S1    EXEC      PGM=progname
//D1    DD      (parameters U,T,V)
//D2    DD      (parameters X,Y,Z)
```

Example 1: The following example shows how to override the first DD statement in a cataloged procedure with a DD * statement, and allow subsequent statements to be processed. Without the DDNAME parameter, replacing the first DD statement with a DD * statement would terminate processing of subsequent statements.

The Cataloged Procedures (PROC3)

```
//STEP1   EXEC    PGM=progname
//DD1     DD      (any parameters except *)
//DD2     DD      (any parameters except *)
```

The Input Stream

```
//JOB1          JOB   1234,J.DUBON
//S1            EXEC  PROC3
//STEP1.DD1     DD    DDNAME=D1
//D1            DD    *
```

The STEP1.DD1 statement overrides the DD1 statement; the DD2 statement is processed; then the D1 statement is processed.

Example 2: The following example shows how to override the first DD statement in a cataloged procedure with a DD * statement and add a DD statement.

The Cataloged Procedures (PROC3)

```
//STEP1   EXEC    PGM=progname
//DD1     DD      (any parameters except * )
//DD2     DD      (any parameters except * )
```

The Input Stream		
//JOB2	JOB	1234,J.DUBON
//S1	EXEC	PROC3
//STEP1.DD1	DD	DDNAME=DD4
//DD3	DD	(any parameters except *)
//DD4	DD	*

The DD4 statement effectively overrides the DD1 statement, after the DD2 statement has been processed and the DD3 statement has been added.

Example 3: The following example shows how to concatenate a data set in the input stream with a data set defined by a DD statement in a cataloged procedure.

The Cataloged Procedures (PROC3)		
//STEP1	EXEC	PGM=progname
//DD1	DD	(any parameters except *)
//DD2	DD	(any parameters except *)

The Input Stream		
//JOB3	JOB	1234,J.DUBON
//S1	EXEC	PROC3
//STEP1.DD1	DD	(blank operand field)
//	DD	DDNAME=DD3
//DD3	DD	*

The data set in the input stream is concatenated to the data set defined by the DD1 statement after the DD2 statement has been processed.

Example 4: The following example shows how to concatenate a data set in the input stream with a data set defined by a DD statement in a cataloged procedure and add a DD statement.

The Cataloged Procedures (PROC3)		
//STEP1	EXEC	PGM=progname
//DD1	DD	(any parameters except *)
//DD2	DD	(any parameters except *)

The Input Stream

```
//JOB4          JOB   1234,J.DUBON
//S1            EXEC  PROC3
//STEP1.DD2     DD    (blank operand field)
//              DD    DDNAME=DD4
//DD3           DD    (any parameters except *)
//DD4           DD    *
```

The data set in the input stream is concatenated to the data set defined by the DD2 statement after the DD3 statement has been added.

Example 5: The use of the DDNAME parameter shown in Examples 3 and 4 requires more DD statements in the input stream than would be required if the DDNAME parameter were contained within the catalog procedure. If procedure PROC3 is modified as follows:

The Cataloged Procedures (PROC3)

```
//STEP1   EXEC   PGM=progname
//DD1     DD     (any parameters except * )
//        DD     DDNAME=DD3
//DD2     DD     (any parameters except * )
//        DD     DDNAME=DD4
```

The data set in the input stream can then be concatenated with the data set defined by the DD1 statement by using the following statements:

The Input Stream

```
//JOB5    JOB    1234,J.DUBON
//S1      EXEC   PROC3
//DD3     DD     *
```

Because there isn't a statement named DD4, the data set defined by the DD2 statement will be concatenated with a dummy data set.

3) IBM - SUPPLIED PROCEDURES

IBM distributes cataloged procedures with the Operating System. However, you need not incorporate these procedures when you generate a system.

The following text describes the conventions used in IBM-supplied procedures.

Procedure Naming Conventions

Procedure names begin with the abbreviated name of the processor program that is to compile source statements or whose output is to be linkage edited. The names assigned to each processor are as follows:

Assembler	ASM
COBOL	COB
FORTRAN	FORT
Linkage Editor	LKED
Sort	SORT
Testran Editor	TTED

If applicable, a design-point indicator follows the processor's abbreviated name.

1. The Linkage Editor is not followed by a design-point indicator because all IBM-supplied procedures invoke the largest linkage editor that you select during the system generation procedure. This linkage editor is given the alias IEWL during the system generation process. This name is included in all procedures that invoke the Linkage Editor.

2. Both the Sort and the Testran Editor have only one design point; neither name is followed by a design-point indicator.

The processor's abbreviated name and, if applicable, design point are followed by C, L, G or any combination of them. The "C" indicates compile, the "L" linkage edit, and the "G" go (i.e., execute). Hence, procedure ASMEC is a single step procedure which compiles a program using the Assembler E processor; FORTECLG is a three step procedure, wherein the first step compiles a program using FORTRAN E, the second step linkage edits the output of the first step, and the third step executes the output of the linkage editor.

Step Names in Procedures

In a cataloged procedure, the step name is the same as the abbreviated processor name. The step that executes a compiled and linkage edited program is named GO.

For example, in the procedure named COBECLG, the first step is named COB; the second step is named LKED, and the third step is named GO.

Unit Names in Procedures

The three unit names used in IBM-supplied cataloged procedures are as follows:

SYSSQ	any magnetic tape or direct-access device
SYSDA	any direct-access device
SYSCP	any card punch

A pool of units must be assigned to these unit names during the system generation procedure. For example, you may elect to have only 2311 Disk Storage Drives assigned to the SYSSQ name. Then again, you may assign both 2400 Magnetic Tape Units and 2311 Disk Storage Drives to the SYSSQ name. Although you can assign devices to these classes, you cannot control device selection.

Tapes that are assigned to SYSSQ should have standard labels. If your tapes are unlabeled and you are using the IBM-supplied procedures, you should modify the procedures accordingly.

Data Set Names in Procedures

When DSNAME=&name is used in a DD statement, the specified data set is given a unique name by the scheduler, and is assumed to be a temporary data set which will be deleted when the job completes. If you want to keep the data set, you must override the DD statement with a permanent data set name and appropriate COND parameters.

IBM-Supplied Procedures

Procedures ASMEC/ASMFC: These are single step procedures to assemble a symbolic program and are identical, except that procedure ASMFC specifies Assembler F (IEUASM) in the EXEC statement. Procedure ASMEC is shown here.

```
//ASM        EXEC  PGM=IETASM                                          00020000
//SYSLIB     DD    DSNAME=SYS1.MACLIB,DISP-OLD                        00040000
//SYSUT1     DD    UNIT=SYSSQ,SPACE=(1700,(400,50))                   00060000
//SYSUT2     DD    UNIT=SYSSQ,SPACE=(1700,(400,50))                   00080000
//SYSUT3     DD    UNIT=(SYSSQ,SEP=(SYSUT2,SYSUT1,SYSLIB)),           X00100000
//                 SPACE=(1700,(400,50))                              00120000
//SYSPRINT   DD    SYSOUT=A                                           00140000
//SYSPUNCH   DD    UNIT=SYSCP                                         00160000
```

You must supply the following DD statement in the input stream.

```
//ASM.SYSIN    DD    (* or parameters defining the input data set).
```

If you use the DD * statement, you must follow the source statements with the Delimiter statement.

Note: For a more detailed description of the use of the Assembler E or F procedures, refer to the publication IBM System/360 Operating System: Assembler (E,F) Programmer's

Guide, Form C28-6595.

Procedures ASMECL/ASMFCL: These are two step procedures to assemble a program and linkage edit the output of the assembler and are identical, except that procedure ASMFCL specifies Assembler F (IEUASM) in the EXEC statement for the assembly step. Procedure ASMECL is shown here.

```
//ASM        EXEC  PGM=IETASM                                          00020000
//SYSLIB     DD    DSNAME=SYS1.MACLIB,DISP=OLD                         00040000
//SYSUT1     DD    UNIT=SYSSQ,SPACE=(1700,(400,50))                    00060000
//SYSUT2     DD    UNIT=SYSSQ,SPACE=(1700,(400,50))                    00080000
//SYSUT3     DD    UNIT=(SYSSQ,SEP=(SYSUT2,SYSUT1,SYSLIB)),           X00100000
//                 SPACE=(1700,(400,50))                               00120000
//SYSPRINT   DD    SYSOUT=A                                            00140000
//SYSPUNCH   DD    DSNAME=&LOADSET,UNIT=SYSSQ,                         X00160000
//                 SPACE=(80,(200,50)),DISP=(MOD,PASS)                 00180000
//LKED       EXEC  PGM=IEWL,PARM=(XREF,LIST,NCAL)                      00200000
//SYSLIN     DD    DSNAME=&LOADSET,DISP=(OLD,DELETE)                   00220000
//          DD    DDNAME=SYSIN                                        00240000
//SYSLMOD    DD    DSNAME=&TEMP(PDS),UNIT=SYSDA,                       X00260000
//                 SPACE=(1024,(50,20,1))
//SYSUT1     DD    UNIT=(SYSDA,SEP=(SYSIN,SYSLMOD)),                   X00280000
//                 SPACE=(1024,(50,20))                                00300000
//SYSPRINT   DD    SYSOUT=A                                            00320000
```

You must supply the following DD statement in the input stream.

```
//ASM.SYSIN    DD    (* or parameters defining the input data set)
```

If you want to linkage edit previously compiled modules with the output of the current compilation, you must also supply the following DD statement.

```
//LKED.SYSIN    DD    (* or parameters defining the input data set)
```

If you use the DD * statement, in either of the above cases, you must follow the input data with the Delimiter statement.

Procedures ASMECLG/ASMFCLG: These are three step procedures to assemble, linkage edit, and execute the compiled program and are identical, except that procedure ASMFCLG specifies Assembler F (IEUASM) in the EXEC statement for the assembly step. Procedure ASMECLG is shown here.

```
//ASM       EXEC  PGM=IETASM                                    00020000
//SYSLIB    DD    DSNAME=SYS1.MACLIB,DISP=OLD                   00040000
//SYSUT1    DD    UNIT=SYSSQ,SPACE=(1700,(400,50))              00060000
//SYSUT2    DD    UNIT=SYSSQ,SPACE=(1700,(400,50))              00080000
//SYSUT3    DD    UNIT=(SYSSQ,SEP=(SYSUT2,SYSUT1,               X00100000
//                SYSLIB)),SPACE=(1700,(400,50))              00120000
//SYSPRINT  DD    SYSOUT=A                                      00140000
//SYSPUNCH  DD    DSNAME=&LOADSET,UNIT=SYSSQ,                   X00160000
//                SPACE=(80,(200,50)),DISP=(MOD,PASS)         00180000
//LKED      EXEC  PGM=IEWL,PARM=(XREF,LET,LIST,NCAL)            00200000
//SYSLIN    DD    DSNAME=&LOADSET,DISP=(OLD,DELETE)             00220000
//              DD  DDNAME=SYSIN                                00240000
//SYSLMOD   DD    DSNAME=&GOSET(GO),UNIT=SYSDA,                 X00260000
//                SPACE=(1024,(50,20,1)),DISP=(MOD,PASS)      00280000
//SYSUT1    DD    UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),            X00300000
//                SPACE=(1024,(50,20))                        00320000
//SYSPRINT  DD    SYSOUT=A                                      00340000
//GO        EXEC  PGM=*.LKED.SYSLMOD                            00360000
```

You must supply the following DD statement in the input stream.

```
//ASM.SYSIN DD  (* or parameters defining the input data set)
```

If a problem program refers to data sets, you must supply a DD statement defining each data set. If more than one data set is referred to, and one data set is in the input stream, only the last statement can be a DD * statement. The format of the DD statements is as follows:

```
//GO.ddname DD  (* or parameters defining the data set)
```

If you use the DD * statement, you must follow the input data with the Delimiter statement.

```
//FORT      EXEC  PGM=IEJFAA0                                   00020000
//SYSPRINT  DD    SYSOUT=A                                      00040000
//SYSPUNCH  DD    UNIT=SYSCP                                    00060000
//SYSUT1    DD    UNIT=SYSSQ,SEP=SYSPUNCH,                      X00080000
//                SPACE=(TRK,(30,10))
//SYSUT2    DD    UNIT=SYSSQ,SEP=SYSUT1,                        X00100000
//                SPACE=(TRK,(30,10))
//SYSLIN    DD    DSNAME=&LOADSET,DISP=(MOD,PASS),              100120000
//                UNIT=SYSSQ,
//                SEP=SYSPUNCH,SPACE=(TRK,(30,10))            00140000
```

You must supply the following DD statement in the input stream.

```
//FORT.SYSIN DD  (* or parameters defining an input data set)
```

If several FORTRAN source programs are batched for compilation, you can use only the DD * statement. When you use the DD * statement, you must follow the input data with the Delimiter statement.

Note: For a more detailed description of the use of the FORTRAN E procedures, refer to the publication IBM System/360 Operating System: FORTRAN (E) Programmer's Guide, Form C28-6603.

Procedure FORTECLG: A three step procedure to do a compile, load, go using the FORTRAN E compiler.

```
//FORT       EXEC  PGM=IEJFAAA0                                    00020000
//SYSPRINT   DD    SYSOUT=A                                        00040000
//SYSPUNCH   DD    UNIT=SYSCP                                      00060000
//SYSUT1     DD    UNIT=SYSSQ,SEP=SYSPUNCH,                        00080000
//               SPACE=(TRK,(30,10))
//SYSUT2     DD    UNIT=SYSSQ,SEP=SYSUT1,                          00100000
//               SPACE=(TRK,(30,10))
//SYSLIN     DD  DSNAME=&LOADSET,DISP=(MOD,PASS),                 X100120000
//               UNIT=SYSSQ,                                      X
//               SEP=SYSPUNCH,SPACE=(TRK,(30,10))                  00140000
//LKED       EXEC  PGM=IEWL,PARM=(XREF,LIST,LET),                 X00160000
//               COND=(5,LT,FORT)
//SYSLIB     DD    DSNAME=SYS1.FORTLIB,DISP=OLD                    00180000
//SYSLMOD    DD    DSNAME=  GOSET(MAIN),DISP=(NEW,PASS),           X00200000
//               UNIT=SYSDA,SPACE=(1024,(50,20,1))                 00220000
//SYSUT1     DD    UNIT=SYSDA,SEP=(SYSLMOD,SYSLIB),                X00240000
//               SPACE=(1024,(30,20))
//SYSPRINT   DD    SYSOUT=A                                        00260000
//SYSLIN     DD  DSNAME=*.FORT.SYSLIN,DISP=(OLD,DELETE),          100280000
//               DCB=(RECFM=F,BLKSIZE=80),                         00300000
//               UNIT=SYSSQ
//           DD    DDNAME=SYSIN                                    00320000
//GO         EXEC  PGM=*.LKED.SYSLMOD,                            X00340000
//               COND=((5,LT,FORT),(5,LT,LKED))
//FT02F001   DD    UNIT=SYSCP                                      00360000
//FT03F001   DD    SYSOUT=A                                        00380000
//FT01F001   DD    DDNAME=SYSIN                                    00400000
```

You must supply the following DD statement in the input stream.

```
//FORT.SYSIN  DD  (* or parameters defining an input data set)
```

If several FORTRAN source programs are batched for compilation, you can use only the DD * statement. For a more detailed description, refer to the publication IBM System/360 Operating System: FORTRAN (E) Programmer's Guide, Form C28-6603.

If you want to linkage edit previously compiled modules with the output of the current compilation, you must also supply the following DD statement.

```
//LKED.SYSIN  DD  (* or parameters defining an input data set)
```

If the problem program refers to SYSIN, you must supply the following DD statement.

```
//GO.SYSIN  DD  (* or parameters defining an input data set)
```

If SYSIN is other than data set reference number 1, you must nullify the DD statement whose name is FT01F001, and define the alternative SYSIN data set in a DD statement in the input stream.

If the problem program refers to other data sets, you must use predefined names for the DD statements that define these data sets. For a more detailed description, refer to the publication IBM System/360 Operating System: FORTRAN (E) Programmer's Guide.

Procedure FORTELG: A two step procedure to linkage edit and execute the output of a FORTRAN E compilation.

```
//LKED      EXEC  PGM=IEWL,PARM=(XREF,LIST,LET)                    00020000
//SYSLIB    DD    DSNAME=SYS1.FORTLIB,DISP=OLD                     00040000
//SYSLMOD   DD    DSNAME=&GOSET(MAIN),DISP=(NEW,PASS),            X00060000
//                UNIT=SYSDA,SPACE=(1024,(50,20,1))                00080000
//SYSUT1    DD    UNIT=SYSDA,SEP=(SYSLMOD,SYSLIB),                X00100000
//                SPACE=(1024,(30,20))
//SYSPRINT  DD    SYSOUT=A                                         00120000
//SYSLIN    DD    DDNAME=SYSIN                                     00140000
//GO        EXEC  PGM=*.LKED.SYSLMOD,COND=(5,LT,LKED)              00160000
//FT02F001  DD    UNIT=SYSCP                                       00180000
//FT03F001  DD    SYSOUT=A                                         00200000
//FT01F001  DD    DDNAME=SYSIN                                     00220000
```

You must supply the following DD statement in the input stream.

```
//LKED.SYSIN  DD  (* or parameters defining the input data set)
```

If the problem program refers to SYSIN, you must supply the following DD statement.

```
//GO.SYSIN  DD  (* or parameters defining an input data set)
```

If SYSIN is other than data set reference number 1, you must nullify the DD statement whose name is FT01F001, and define the alternative SYSIN data set in a DD statement in the input stream.

If the problem program refers to other data sets, you must use predefined names for the DD statements that define these data sets. For a more detailed description, refer to the publication IBM System/360 Operating System: FORTRAN (E) Programmer's Guide.

Procedure SORT: A single step procedure to execute the SORT program.

```
//SORT     EXEC  PGM=IERRCO00                                          00020000
//SYSOUT   DD    SYSOUT=A                                              00040000
//SYSPRINT DD    DUMMY                                                 00060000
//SYSLMOD  DD    UNIT=SYSDA,SPACE=(1024,(50,20,1))                     00080000
//SYSLIN   DD    UNIT=SYSDA,SPACE=(80,(150,10))                        00100000
//SORTLIB  DD    DSNAME=SYS1.SORTLIB,DISP=OLD                          00120000
//SYSUT1   DD    UNIT=(SYSDA,SEP=(SORTLIB,SYSLMOD,SYSLIN)), X00140000
//               SPACE=(1024,(50,20))                                  00160000
```

You must supply the following statements in the input stream.

Input for a Sort run

//SORTIN DD (parameters defining an input data set)

Input for a Merge run

//SORTIN01 DD (parameters defining an input data set)

- Up to 16 data sets can be specified

//SORTIN16 DD (parameters defining an input data set)

Output for a Sort/Merge run

//SORTOUT DD (parameters defining an output data set)

Intermediate storage for a Sort/Merge run

//SORTWK01 DD (parameters defining an intermediate work data set)

//SORTWK32 DD (parameters defining an intermediate work data set)

If you have written routines to modify the Sort/Merge program, and these routines are in the data set defined by the DD statement whose name is SYSIN, you must supply the following DD statements.

//SORTMODS DD (paramenters defining a temporary partitioned data set)

This partitioned data set must be large enough to contain all the routines that are in the SYSIN data set.

//SYSIN DD (* or parameters defining an input data set)

If you use the DD * statement, you must follow the input data with a Delimiter statement.

If the modification routines are in a partitioned data set you must supply a statement that defines the data set.

For a detailed description and examples of DD statements used with Sort/Merge, refer to the publication IBM System/360 Operating System: Sort/Merge, Form C28-6543.

Procedure LKED: A single step procedure that linkage edits a module and passes the load module to another step in the same job.

```
//LKED     EXEC  PGM=IEWL,PARM='XREF,LIST,LET,NCAL'                  00020000
//SYSPRINT DD    SYSOUT=A                                            00040000
//SYSLIN   DD    DDNAME=SYSIN                                        00060000
//SYSLMOD  DD    DSNAME=&GOSET(GO),SPACE=(1024,(50,20,1)),           C0008000
//               UNIT=SYSDA,DISP=(MOD,PASS)                          00100000
//SYSUT1   DD    UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),                  C0012000
//              SPACE=(1024,(200,20))                                00140000
```

You must supply the following DD statement in the input stream.

//SYSIN DD (* or parameters defining an input data set)

If you use the DD * statement you must follow the input data with the Delimiter statement.

Procedure LKEDG: A two step procedure that linkage edits a module and executes the load module.

```
//LKED     EXEC  PGM=IEWL,PARM='XREF,LIST,LET,NCAL'                  00020000
//SYSPRINT DD    SYSOUT=A                                            00040000
//SYSLIN   DD    DDNAME=SYSIN                                        00060000
//SYSLMOD  DD    DSNAME=&GOSET(GO),SPACE=(1024,(50,20,1)),           C0008000
//               UNIT=SYSDA,DISP=(MOD,PASS)                          00100000
//SYSUT1   DD    UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),                  C0012000
//              SPACE=(1024,(200,20))                                00140000
//GO       EXEC  PGM=*.LKED.SYSLMOD                                  00160000
```

You must supply the following DD statement in the input stream.

//LKED.SYSIN DD (* or parameters defining an input data set)

If you use the DD * statement, you must follow the input data with a Delimiter statement.

Procedure TTED: A single step procedure that executes the Testran Editor.

```
//EDIT      EXEC  PGM=IEGTTEDT                          00020000
//SYSUT1    DD    UNIT=SYSDA,SPACE=(500,100))           00040000
//SYSPRINT  DD    SYSOUT=A                              00060000
```

You must supply the following DD statement in the input stream.

```
//SYSTEST DD (parameters defining a data set)
```

You should use a unique DSNAME parameter and appropriate DISP parameters in the above statement.

Procedure TASMEGED: A four step procedure that assembles a program using the Testran interpreter, linkage edits the output of the assembler, executes the load module, and executes the Testran Editor to analyze the output of the assembled program.

```
//ASM       EXEC  PGM=IETASM,PARM=(LOAD,TEST)          00020000
//SYSLIB    DD    DSNAME=SYS1.MACLIB,DISP=(OLD)         00040000
//SYSUT1    DD    UNIT=(SYSSQ,SEP=(SYSLIB)),            C00060000
//                SPACE=(400,(150,20))                  00080000
//SYSUT2    DD    UNIT=(SYSSQ,SEP=(SYSUT1)),            C00100000
//                SPACE=(400,(150,20))
//SYSUT3    DD    UNIT=(SYSSQ,SEP=(SYSLIB,SYSUT2)),     C00120000
//                SPACE=(400,(150,20))                  00140000
//SYSPRINT  DD    SYSOUT=A                              00160000
//SYSPUNCH  DD    DSNAME=&LOADSET,UNIT=SYSDA,           C00180000
//                SPACE=(80,(200,50)),DISP=(MOD,PASS)   00200000
//LKED      EXEC  PGM=IEWL,                             C00220000
//                PARM=(XREF,LIST,LET,NCAL,TEST)
//SYSPRINT  DD    SYSOUT=A                              00240000
//SYSLIN    DD    DSNAME=&LOADSET,DISP=(OLD)            00260000
//          DD    DDNAME=SYSIN                          00280000
//SYSLMOD   DD    DSNAME=&GOSET(GO),SPACE=(1024,(50,20,1)) C00300000
//                UNIT=SYSDA,DISP=(MOD,PASS)            00320000
//SYSUT1    DD    UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),    C00340000
//                SPACE=(1024,(200,20))                 00360000
//GO        EXEC  PGM=*.LKED.SYSLMOD                    00380000
//SYSTEST   DD    DSNAME=&TESTSET,SPACE=(300,(100)),    C00400000
//                UNIT=SYSSQ,DISP=(NEW,PASS)            00420000
//EDIT      EXEC  PGM=IEGTTEDT                          00440000
//SYSUT1    DD    UNIT=SYSDA,SPACE=(500,(100))          00460000
//SYSTEST   DD    DSNAME=&TESTSET,UNIT=SYSSQ,           C00480000
//                SEP=(SYSUT1),DISP=(OLD,DELETE)        00500000
//SYSPRINT  DD    SYSOUT=A                              00520000
```

If you want to keep the SYSTEST data set for subsequent testing, you should override the data set name &TESTSET with another data set name, and override the DELETE disposition with KEEP.

If you want to select what is to be tested, you should add parameters to the EXEC statement whose name is EDIT.

When a job step abnormally terminates, that step and any subsequent steps in the job are skipped. If your problem program abnormally terminates, and you are using the procedure TASMEGED, the Testran Editor will not be executed. Furthermore, since the disposition of the data set defined by the SYSTEST DD statement is PASS, that data set will be deleted and all test data will be lost.

If you expect that a problem program being tested will abnormally terminate, you should use the procedure TASMEG as one job and follow it with the procedure TTED as another job. You should also assign a KEEP disposition to the data set defined by the SYSTEST DD statement. This will allow you to recover any test data placed in this data set.

Procedure TASME: A two step procedure that assembles a program using the Testran interpreter and linkage edits the output of the assembler.

```
//ASM        EXEC  PGM=IETASM,PARM=(LOAD,TEST)                    00020000
//SYSLIB     DD    DSNAME=SYS1.MACLIB,DISP=(OLD)                  00040000
//SYSUT1     DD    UNIT=(SYSSQ,SEP=(SYSLIB)),                     C00060000
//                 SPACE=(400,(150,20))                           00080000
//SYSUT2     DD    UNIT=(SYSSQ,SEP=(SYSUT1)),                     C00100000
//                 SPACE=(400,(150,20))
//SYSUT3     DD    UNIT=(SYSSQ,SEP=(SYSLIB,SYSUT2))               C00120000
//                 SPACE=(400,(150,20))                           00140000
//SYSPRINT   DD     SYSOUT=A                                      00160000
//SYSPUNCH   DD    DSNAME=&LOADSET,UNIT=SYSDA,                    C00180000
//                 SPACE=(80,(200,50)),DISP=(MOD,PASS)           00200000
//LKED       EXEC  PGM=IEWL,                                      C00220000
//                 PARM=(XREF,LIST,LET,NCAL,TEST)
//SYSPRINT   DD     SYSOUT=A                                      00240000
//SYSLIN     DD    DSNAME=&LOADSET,DISP=(OLD)                     00260000
//           DD     DDNAME=SYSIN                                  00280000
//SYSLMOD    DD     DSNAME=&GOSET(GO),SPACE=(1024,(50,20,1))      C00300000
//                  UNIT=SYSDA,DISP=(MOD,PASS)                   00320000
//SYSUT1     DD     UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),           C00340000
//                 SPACE=(1024,(200,20))                          00360000
```

You must supply the following DD statement in the input stream.

```
//ASM.SYSIN  DD  (* or parameters defining an input data set)
```

If you use the DD * statement you must follow the input data with a Delimiter statement.

Procedure TASMEG: A three step procedure that assembles a program using the Testran interpreter, linkage edits the output of the assembler, and executes the load module.

```
//ASM       EXEC  PGM=IETASM,PARM=(LOAD,TEST)                    00020000
//SYSLIB    DD    DSNAME=SYS1.MACLIB,DISP=(OLD)                  00040000
//SYSUT1    DD    UNIT=(SYSSQ,SEP=(SYSLIB)),                     C0006000
//                SPACE=(400,(150,20))                           00080000
//SYSUT2    DD    UNIT=(SYSSQ,SEP=(SYSUT1)),                     C0010000
//                SPACE=(400,(150,20))
//SYSUT3    DD    UNIT=(SYSSQ,SEP(SYSLIB,SYSUT2)),               C0012000
//                SPACE=(400,(150,20))                           00140000
//SYSPRINT  DD    SYSOUT=A                                       00160000
//SYSPUNCH  DD    DSNAME=&LOADSET,UNIT=SYSDA,                    C0018000
//                SPACE=(80,(200,50)),DISP=(MOD,PASS)            00200000
//LKED      EXEC  PGM=IEWL,PARM=(XREF,LIST,LET,MCAL,TEST)        00220000

//SYSPRINT  DD    SYSOUT=A                                       00240000
//SYSLIN    DD    DSNAME=&LOADSET,DISP=(OLD)                     00260000
//          DD    DDNAME=SYSIN                                   00280000
//SYSLMOD   DD    DSNAME=&GOSET(GO),SPACE=(1024,(50)(20(1)),C0030000
//                UNIT=SYSDA,DISP=(MOD,PASS)                     00320000
//SYSUT1    DD    UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),             C0034000
//                SPACE=(1024,(200,20))                          00360000
//GO        EXEC  PGM=*.LKED.SYSLMOD                             00380000
```

You must supply the following DD statements in the input stream.

```
//SYSTEST DD (parameters defining a data set)
```

You should use a unique DSNAME parameter and appropriate DISP parameters in the above statement.

```
//ASM.SYSIN DD (* or parameters defining an input data set)
```

If you use the DD * statement, you must follow the input data with a Delimiter statement.

Procedure MOD: A single step procedure to execute the system utility program that permits modifying system control data.

```
//MOD       EXEC  PGM=IEHPROGM                                   00020000
//DDSRV     DD    VOLUME=REF=SYS1.SVCLIB,DISP=OLD                00040000
//SYSPRINT  DD    SYSOUT=A                                       00060000
```

You should supply the following DD statement in the input stream.

```
//SYSIN DD (* or parameters defining an input data set)
```

If you use the DD * statement, you should follow the input data with a Delimiter statement.

Procedure LIST: A single step procedure to execute the utility program that lists system control data.

```
//LIST     EXEC  PGM=IEHLIST                                  00020000
//DDSRV    DD    VOLUME=REF=SYS1.SVCLIB,DISP=OLD              00040000
//SYSPRINT DD    SYSOUT=A                                     00060000
```

You should supply the following DD statement in the input stream.

```
//SYSIN  D  (* or parameters defining an input data set)
```

If you use the DD * statement, you should follow the input data with a Delimiter statement.

Procedure COBEC: A single step procedure to execute the COBOL E compiler.

```
//COB      EXEC   PGM=IEPCBL00                                 00020000
//SYSPRINT DD     SYSOUT=A                                     00040000
//SYSPUNCH DD     UNIT=SYSCP                                   00060000
//SYSUT1   DD     UNIT=SYSSQ,SPLIT=(2,CYL,(40,10))            00080000
//SYSUT2   DD     UNIT=SYSSQ,SPLIT=4                           00100000
//SYSUT3   DD     UNIT=SYSSQ,SPLIT=4                           00120000
```

You must supply the following DD statement in the input stream:

```
//COB.SYSIN  DD  (* or parameters defining an input data set)
```

When you use the DD * statement you must follow the input data with the Delimiter statement (/*).

Procedure COBELG: A two step procedure to linkage edit and execute the output of a COBOL E compilation.

```
//LKED     EXEC   PGM=IEWL,PARM=(XREF,LIST,LET)              00020000
//SYSLIN   DD     DDNAME=SYSIN                               00040000
//SYSLMOD  DD     DSNAME=&GODATA(RUN),DISP=(NEW,PASS),      X00060000
//                UNIT=SYSDA,SPACE=(1024,(50,20,1))          00080000
//SYSLIB   DD     DSNAME=SYS1.COBLIB,DISP=(OLD,KEEP)         00100000
//SYSUT1   DD     UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),        C00120000
//                SPACE=(1024,(50,20,1))
//SYSPRINT DD     SYSOUT=A                                   00140000
//GO       EXEC   PGM=*.LKED.SYSLMOD,COND=(9,LT,LKED)        00160000
//SYSABEND DD     SYSOUT=A                                   00180000
```

You must supply the following DD statement in the input stream:

```
//LKED.SYSIN  DD  (* or parameters defining the input data set)
```

If the problem program refers to SYSIN, you must supply the following DD statement:

```
//GO.SYSIN  DD  (* or parameters defining the input data set)
```

If the problem program refers to other data sets, you must provide DD statements that define these data sets.

Procedure COBECLG: A three step procedure to do a compile, linkage edit, and execute using the COBOL compiler.

```
//COB         EXEC   PGM=IEPCBL00                                    00020000
//SYSPRINT    DD     SYSOUT=A                                        00040000
//SYSUT1      DD     UNIT=SYSSQ,SPLIT=(2,CYL,(40,10))                00060000
//SYSUT2      DD     UNIT=SYSSQ,SPLIT=4                              00080000
//SYSUT3      DD     UNIT=SYSSQ,SPLIT=4                              00100000
//SYSPUNCH    DD     DSNAME=&LOADSET,DISP=(MOD,PASS),               X00120000
//                   UNIT=SYSSQ,SPACE=(TRK,(50,10))                  00140000
//LKED        EXEC   PGM=IEWL,PARM=(XREF,LIST,LET)                   00160000
//SYSLIN      DD     DSNAME=&LOADSET,DISP=(OLD,DELETE)               00180000
//                   DDNAME=SYSIN                                    00200000
//SYSLMOD     DD     DSNAME=&GODATA(RUN),DISP=(NEW,PASS),           X00220000
//                   UNIT=SYSDA,SPACE=(1024,(50,20,1))               00240000
//SYSLIB      DD     DSNAME=SYS1.COBLIB,DISP=(OLD,KEEP)              00260000
//SYSUT1      DD     UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),             X00280000
//                   SPACE=(1024,(50,20,1))
//SYSPRINT    DD     SYSOUT=A                                        00300000
//GO          EXEC   PGM=*.LKED.SYSLMOD,COND=(9,LT,LKED)             00320000
//SYSABEND    DD     SYSOUT=A                                        00340000
```

You must supply the following DD statement in the input stream:

```
//COB.SYSIN  DD  (* or parameters defining an input data set)
```

If the problem program refers to SYSIN, you must supply the following DD statement:

```
//GO.SYSIN  DD  (* or parameters defining an input data set)
```

If the problem program refers to other data sets, you must provide DD statements that define these data sets.

Appendix 2 - Section II

EXAMPLES OF JOB CONTROL LANGUAGE PROCEDURES

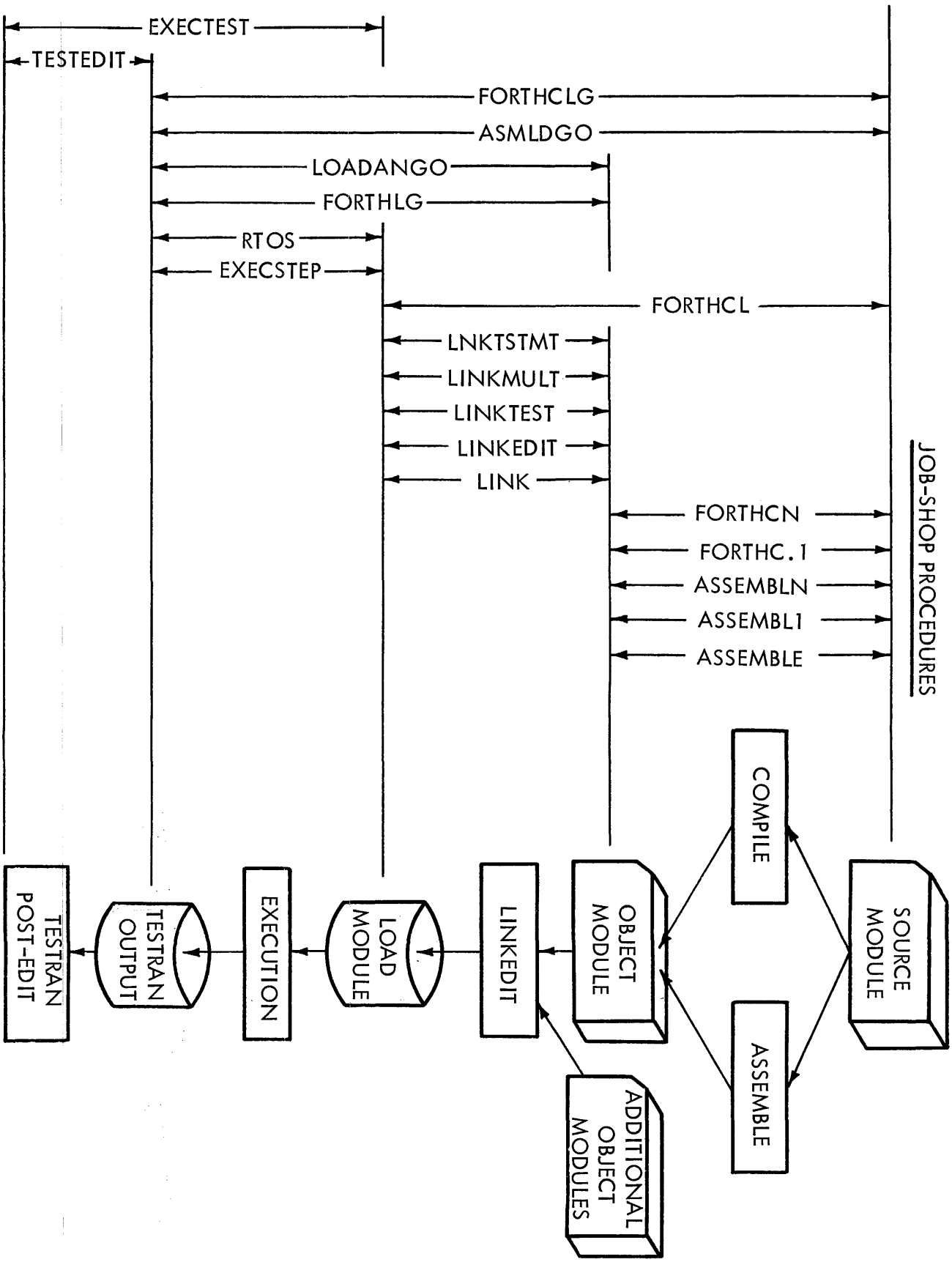

EXECTEST
TESTEDIT
FORTHCLG
ASMLDGO
LOADANGO
FORTHLG
RTOS
EXECSTEP
FORTHCL
LNKTSTMT
LINKMULT
LINKTEST
LINKEDIT
LINK
FORTHCN
FORTHC.1
ASSEMBLN
ASSEMBL1
ASSEMBLE
JOB-SHOP PROCEDURES
TESTRAN POST-EDIT
TESTRAN OUTPUT
EXECUTION
LOAD MODULE
LINKEDIT
OBJECT MODULE
COMPILE
SOURCE MODULE
ASSEMBLE
ADDITIONAL OBJECT MODULES

Figure 1

PROGRAM ASSEMBLE

OBTAIN OBJECT DECK ONLY

DATE

```
         Name           Operation            Operand
1               8   10          14  16      20        25        30        35        40
//ASMJØB1  JØB  MSGLEVEL=1
//        EXEC  ASSEMBLE
//SYSIN     DD  *
                  ASSEMBLER LANGUAGE
                  SOURCE DECK
/*
```

```
//ASMJØB2  JØB MSGLEVEL=1
//STEP1   EXEC ASSEMBLE
//SYSIN     DD *
                 ASSEMBLER LANGUAGE
                 SOURCE DECK
/*
```

```
//ASMJOB3  JOB MSGLEVEL=1
//STEP1   EXEC ASSEMBLE
//STEP1.ASM.SYSIN DD *
                 ASSEMBLER LANGUAGE
                 SOURCE DECK
/*
```

PROGRAM: ASSEMBLE USING ADDITIONAL MACRO LIBRARY

PROGRAMMER | DATE

```
//ASMJØB4   JOB MSGLEVEL=1
//        EXEC ASSEMBLE
//ASM.SYSLIB DD
//          DD DSNAME=SMA.MACLIB,                                           C
//             DISP=ØLD,                                                    C
//             VØLUME=SER=SMA205
//SYSIN     DD *
                 ASSEMBLER
                SOURCE MODULE
/*
```

PROGRAM ASSEMBLE AND CREATE LOAD MODULE
ØNE ØBJECT INTØ ØNE LØAD M

DATE

```
//ASMLNK1   JØB MSGLEVEL=1
//         EXEC ASSEMBL1
//SYSIN      DD *

                 ASSEMBLER LANGUAGE
                 SOURCE DECK

/*
//         EXEC LINKEDIT
```

```
//ASMLNK2   JØB MSGLEVEL=1
//         EXEC ASSEMBL1
//SYSIN      DD *

                 ASSEMBLER LANGUAGE
                 SØURCE DECK

/*
//         EXEC LINK
```

PROGRAM ASSEMBLE AND CREATE A LOAD MODULE NAMED PRGMOD1

ØNE ØBJECT INTØ ØNE LØAD MØDULE

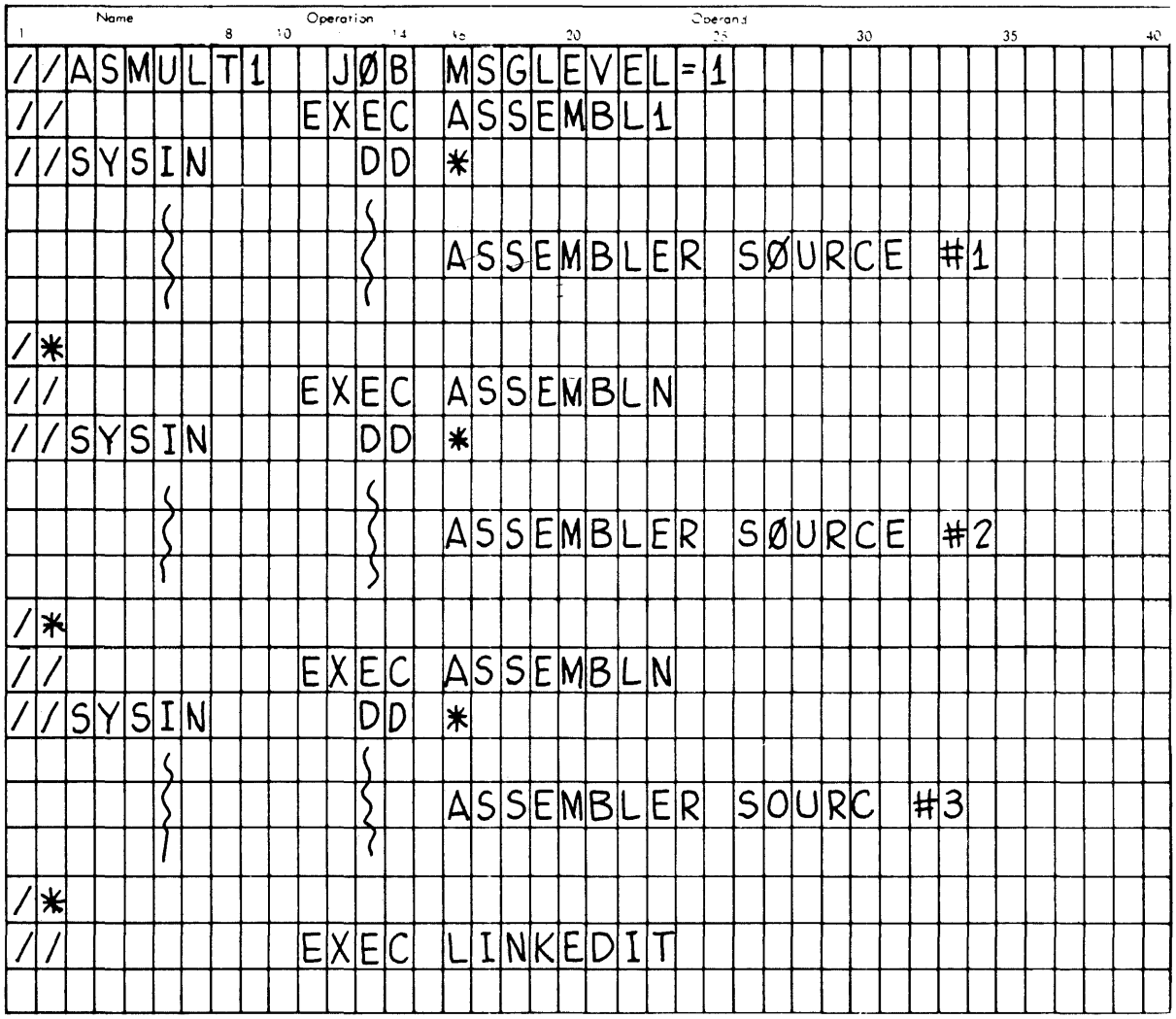
DATE

```
//ASMLNK3   JOB MSGLEVEL=1
//         EXEC ASSEMBL1
//SYSIN      DD *

               ASSEMBLER SOURCE

/*
//         EXEC LINKEDIT
//LNK.SYSLMØD  DD DSNAME=&GØFILE(PRGMOD1)
```

```
//ASMLNK4   JØB MSGLEVEL=1
//         EXEC ASSEMBL1
//SYSIN      DD *

               ASSEMBLER SOURCE

/*
//         EXEC LINK
//LNK.SYSLMOD DD DSNAME=&GØFILE
//SYSIN      DD *
         NAME  PRGMOD1
/*
```

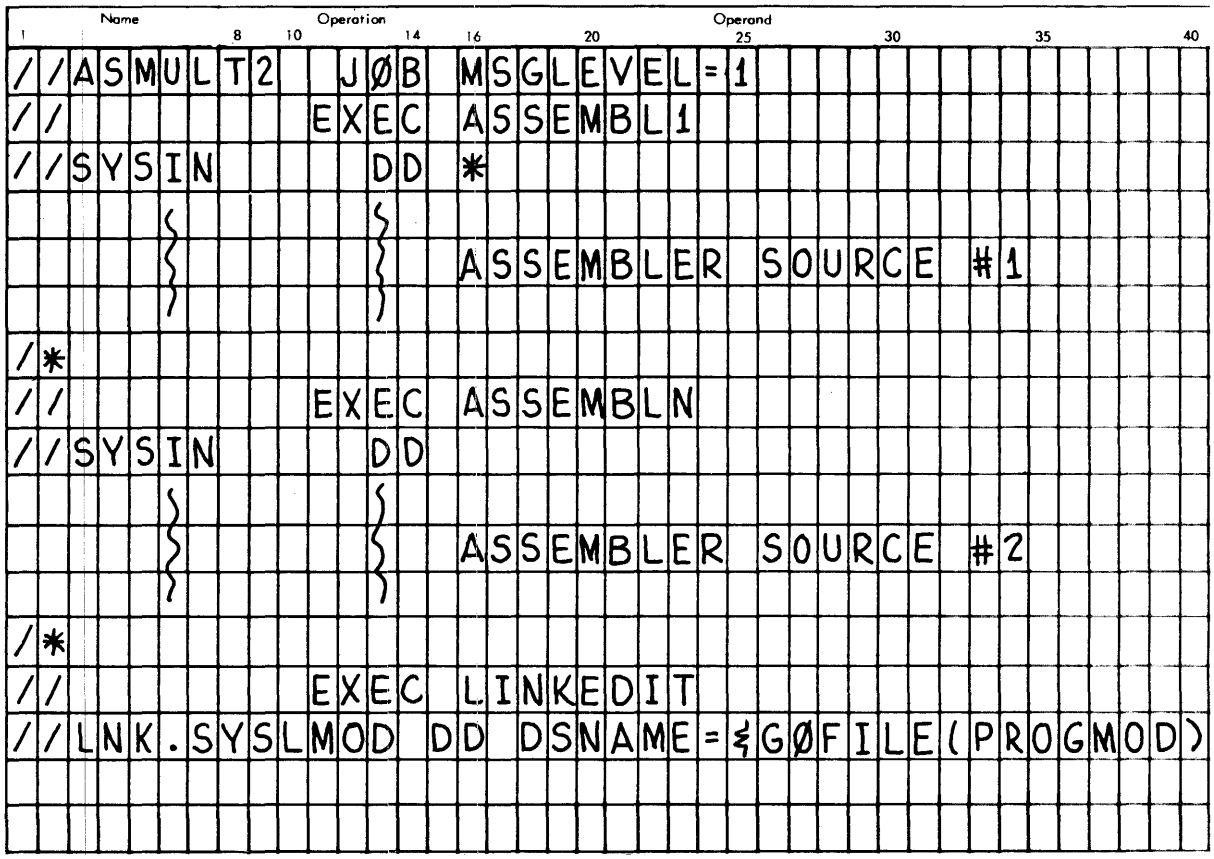
PROGRAM ASSEMBLE (MULTIPLE) AND CREATE ØNE LOAD MODULE

MULTIPLE OBJECT INTO LOAD M | DATE



PROGRAM ASSEMBLE (MULTIPLE) AND CREATE LOAD MODULE PROGMOD

MULTIPLE ØBJECT INTØ ØNE LØAD | DATE



PROGRAM **ASSEMBLE, LINKEDIT, AND EXECUTE**
ØNE ØBJECT, ØNE LØAD MØDULE

DATE

```
//ASMLNKG1  JØB  MSGLEVEL=1
//          EXEC ASMLDGØ
//SYSIN     DD   *
    {         {
    {         {    ASSEMBLER SOURCE   {
    {         {
/*
```

```
//ASMLNKG2  J0B  MSGLEVEL=1
//          EXEC ASSEMBL1
//SYSIN     DD
    {         {
    {         {    ASSEMBLER SOURCE   {
    {         {
/*
//          EXEC LINKEDIT
//          EXEC EXECSTEP
```

PROGRAM ASSEMBLE, LINKEDIT, AND EXECUTE - MODULE = PRGMOD4
ØNE ØBJECT, ØNE LØAD MØDULE

DATE

```
//ASMLNKG3 JØB MSGLEVEL=1
//         EXEC ASMLDGØ
//SYSIN     DD *
     ~          ~
     ~          ~  ASSEMBLER SOURCE   ~
     ~          ~
/*
//LNK.SYSLMØD  DD DSNAME=&GØFILE(PRGMØD4)
```

```
//ASMLNKG4 JØB MSGLEVEL=1
//         EXEC ASSEMBL1
//SYSIN     DD *
     ~          ~
     ~          ~  ASSEMBLER SOURCE   ~
     ~          ~
/*
//         EXEC LINKEDIT
//LNK.SYSLMØD DD DSNAME=&GØFILE(PRGMOD4)
//         EXEC EXECSTEP
```

PROGRAM ASSEMBLE & COMPILE & CREATE MULTIPLE LOAD MODULES
SINGLE OBJECT INTO EACH LOAD MODULE | DATE

```
//JOBMIX1   JOB  MSGLEVEL=1
//         EXEC ASSEMBL1
//SYSIN     DD  *
                                  ASSEMBLER SOURCE
                                  DECK #1
/*
//         EXEC LINKEDIT
//LNK.SYSLMOD  DD  DSNAME=&GØ(ØNE)
//         EXEC FØRTHC1
//SYSIN     DD  *
                                  FØRTRAN SØURCE
                                  DECK #1
/*
//LNK.SYSLMØD  DD  DSNAME=&GØ(TWØ),DISP=(MOD,PASS)
//         EXEC FORTHCL
//SYSIN     DD
                                  FØRTRAN SOURCE
                                  DECK #2
/*
//LNK.SYSLMØD  DD  DSNAME=&GO(THREE),DISP=(MØD,PASS)
```

PROGRAM ASSEMBLE LOAD/AND/GO WITH TESTRAN POSTEDIT
ONE JOB

DATE

PUNCHING INSTRUCTIONS

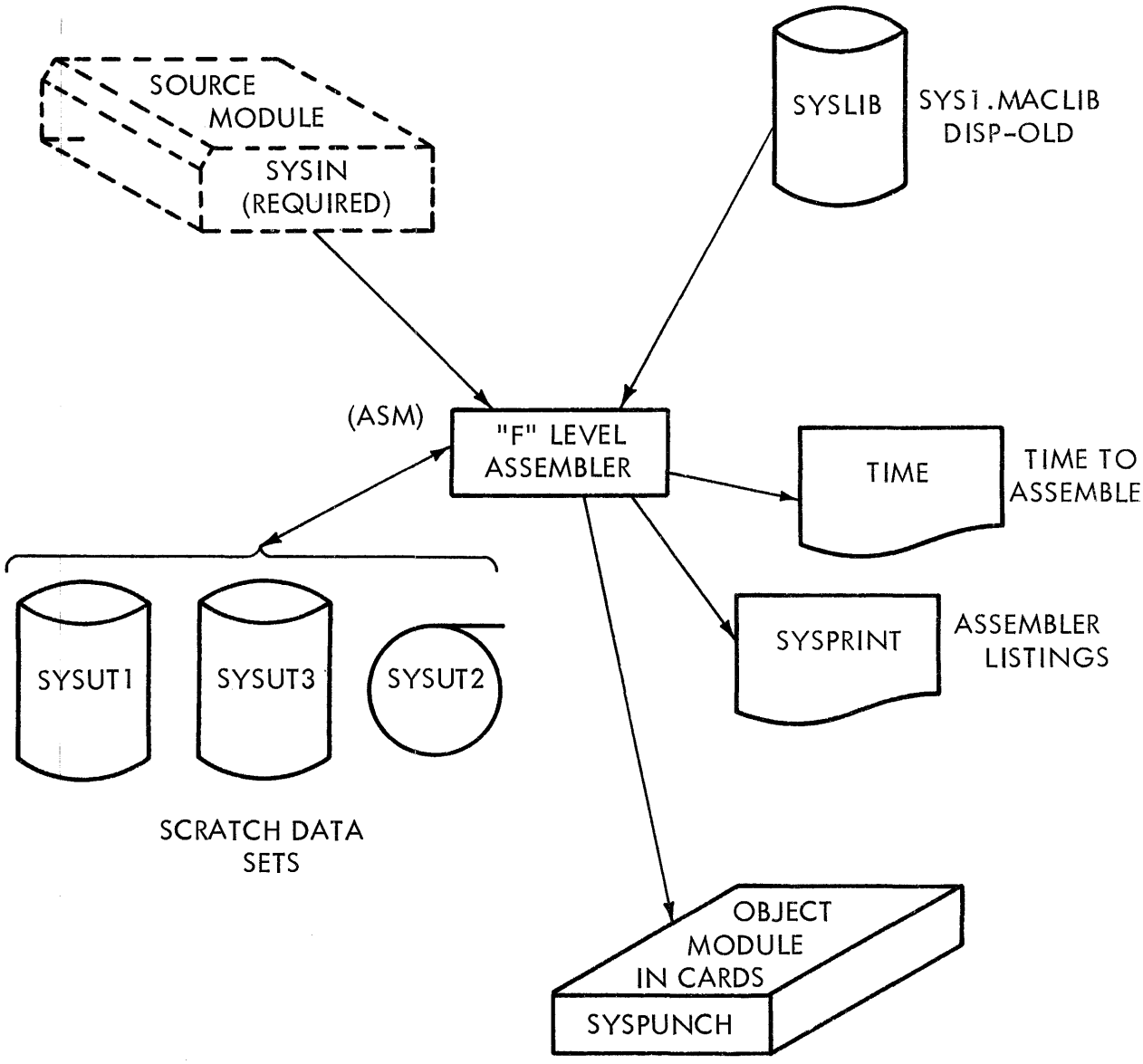
```
//ASMTR1   JØB  MSGLEVEL=1
//         EXEC ASMLDGØ
//SYSIN    DD   *
        { SOURCE }
        { DECK   }
/*
//EXEC.SYSTEST DD UNIT=DISK,DISP=(NEW,PASS),                         C
//              VØLUME=SER=RTØSA,                                     C
//              SPACE=(TRK,(100,10))
//         EXEC TESTEDIT
```

PROGRAM ASSEMBLE LØAD AND GØ WITH TESTRAN ØUTPUT AND POSTEDIT IN SEPARATE JOB

DATE

```
//ASMTR2   JØB MSGLEVEL=1
//        EXEC ASMLDGØ
//SYSIN     DD *
       { SOURCE DECK }
/*
//EXEC.SYSTEST DD DSNAME=BØGSTEST,UNIT=DISK,                            C
//             VOLUME=SER=RTØSA,DISP=(,KEEP),                           C
//             SPACE=(TRK,(100,10))
//ASMTR3   JØB MSGLEVEL=1
//        EXEC TESTEDIT
//EDT.SYSTEST DD DSNAME=BØGSTEST,                                       C
//             VØLUME=SER=RTOSA
```

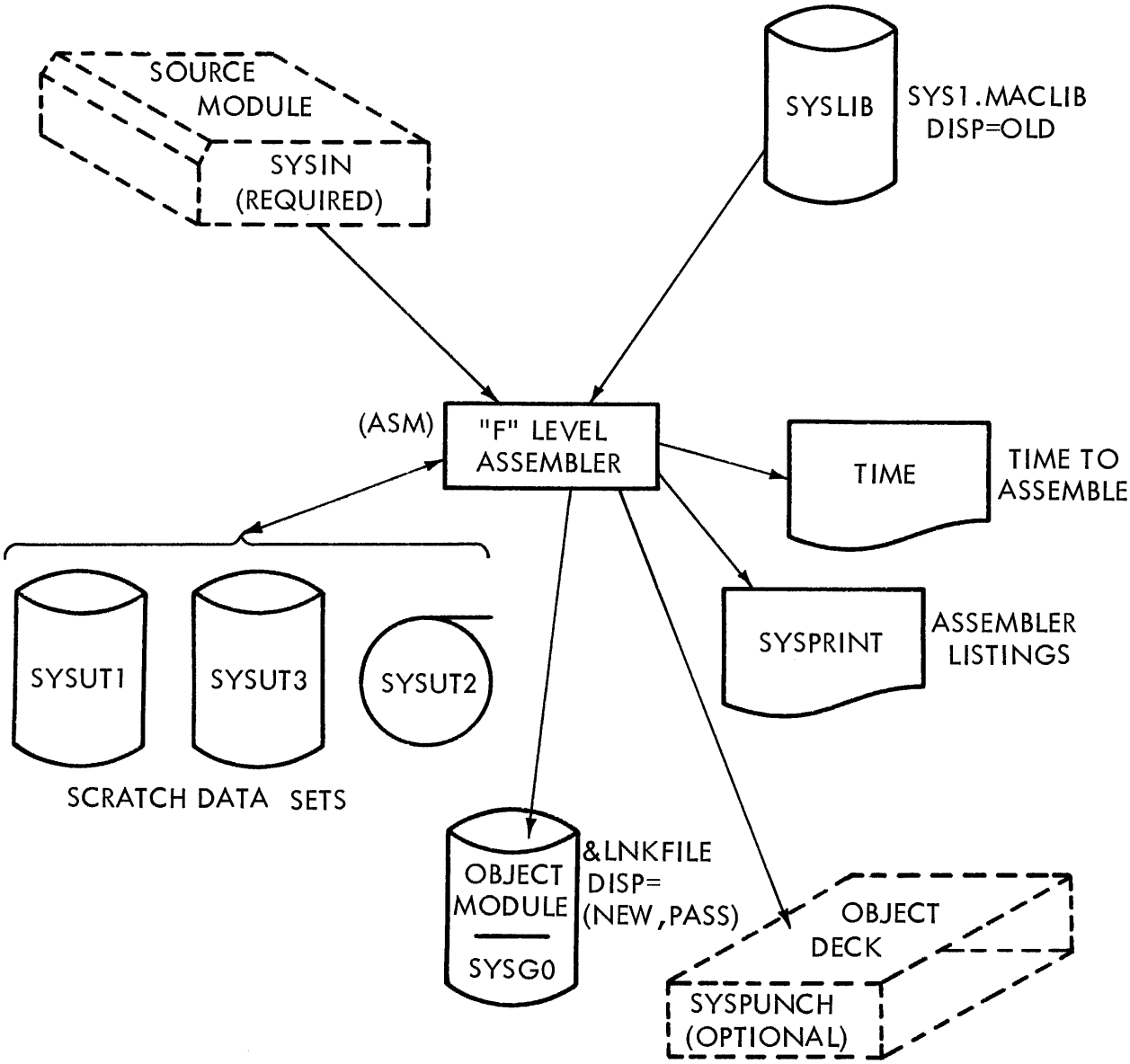
ASSEMBLE (ASM)




```
//ASM     EXEC   PGM=IEUASM                                          X
//               *                                                   X
//               * ASSEMBLY PROCEDURE FOR ONE ASSEMBLY               X
//               *
//SYSPRINT DD    SYSOUT=A
//SYSUT1 DD      UNIT=DISK,SPACE=(CYL,(30,10)),VOLUME=SER=RTOSA
//SYSUT2   DD    UNIT=TAPE,LABEL=(,NL),SEP=(SYSUT1)
//SYSUT3 DD      UNIT=DISK,SPACE=(CYL,(50,10)),VOLUME=SER=RTOSB
//TIME   DD      SYSOUT=A
//SYSLIB DD    DSNAME=SYS1.MACLIB,DISP=OLD
//SYSPUNCH DD    UNIT=PUNCH
```

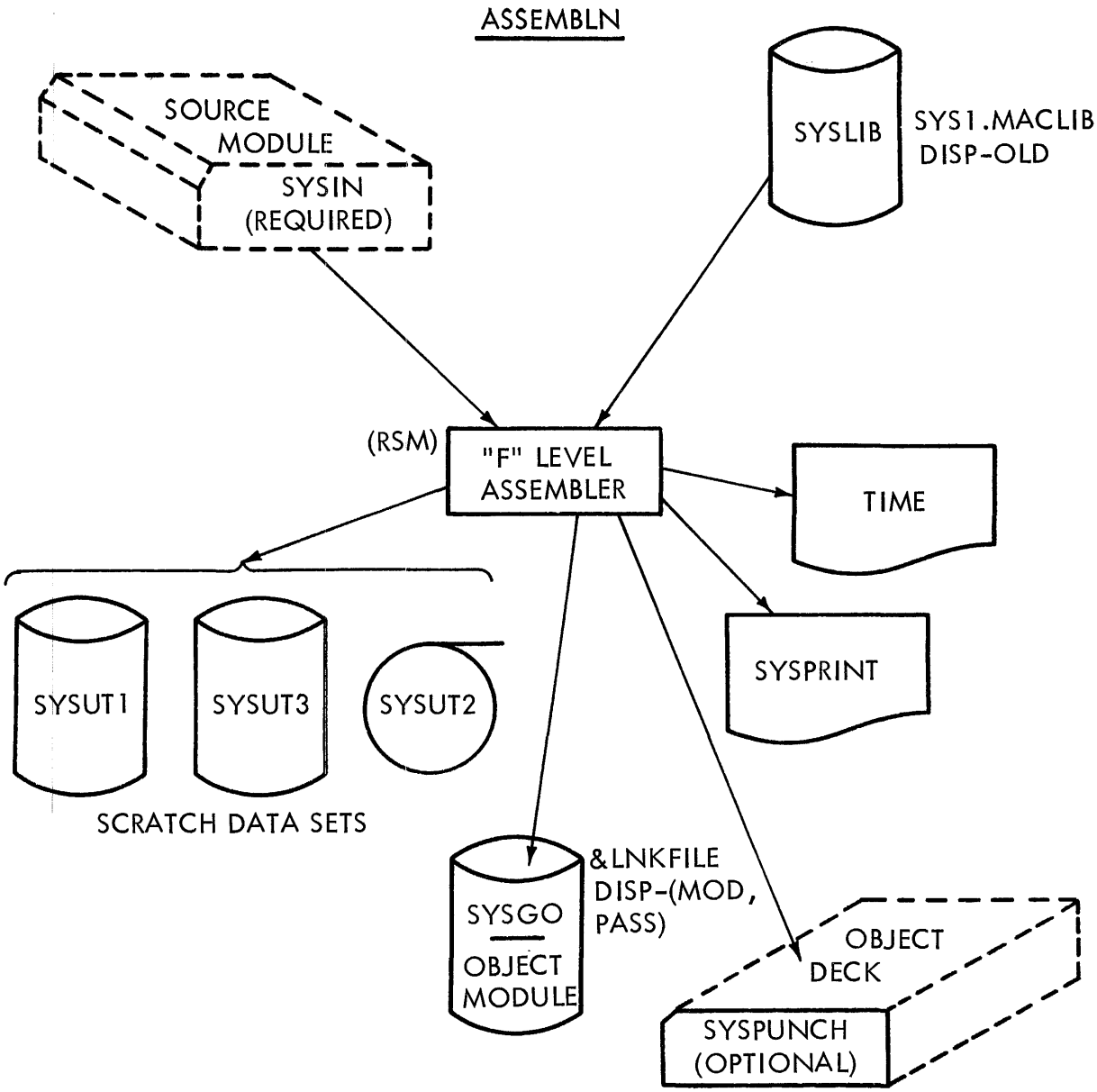
NOTE: Ref. figure on pg. 4.4.101

ASSEMBL1



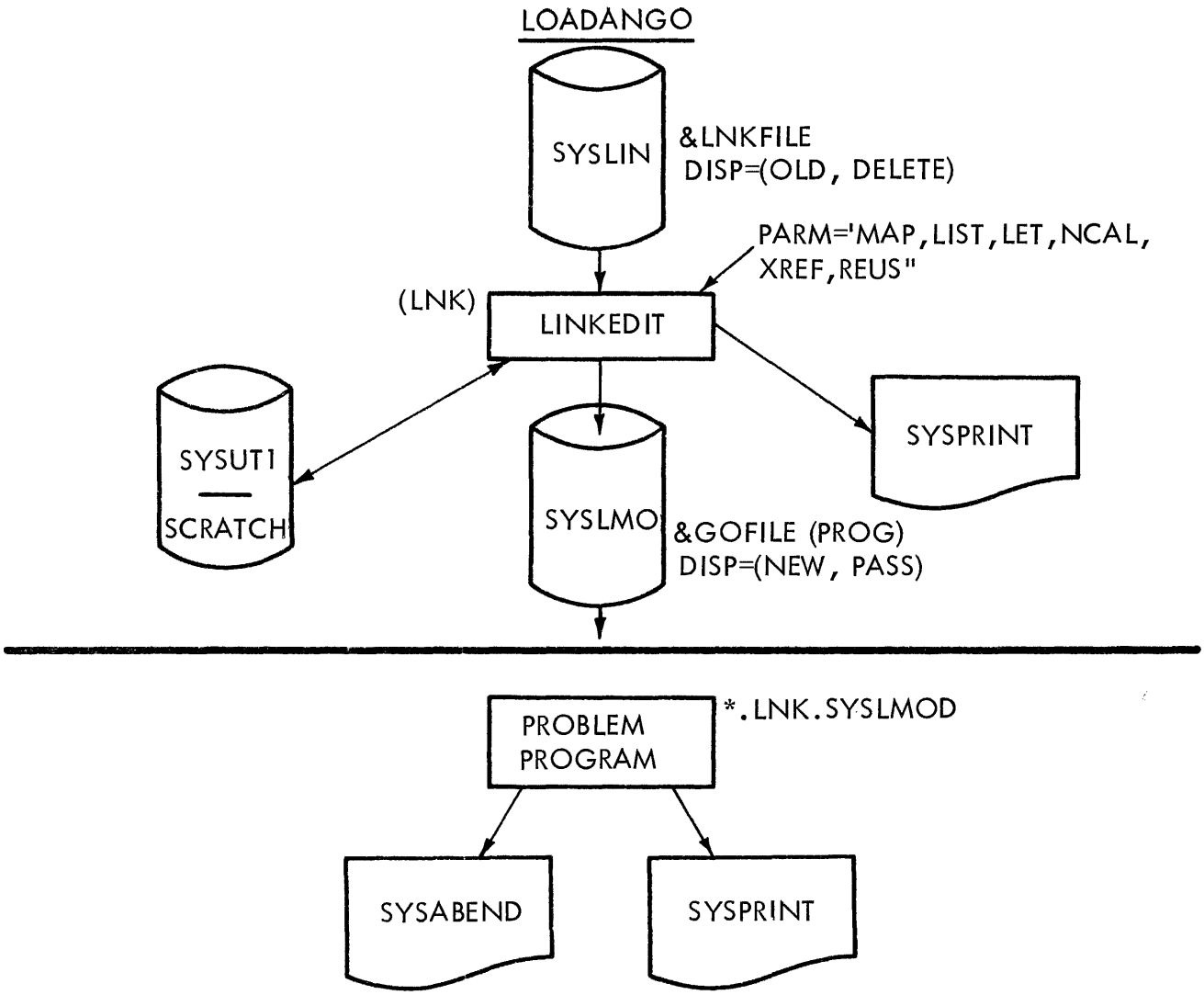

```
//ASM     EXEC   PGM=IEUASM                                          X
//               *                                                   X
//               * ASSEMBLE STEP -- FIRST                            X
//               *                                                   X
//               *FIRST ASSEMBLY PROCEDURE IN STACKED ASSEMBLY       X
//               *
//SYSPRINT DD    SYSOUT=A
//TIME     DD    SYSOUT=A
//SYSUT1   DD    UNIT=DISK,SPACE=(CYL,(30,10)),VOLUME=SER=RTOSA
//SYSUT2    DD   UNIT=TAPE,LABEL=(,NL),SEP=(SYSUT1)
//SYSUT3   DD    UNIT=DISK,SPACE=(CYL,(50,10)),VOLUME=SER=
//SYSGO    DD    UNIT=DISK,SPACE=(CYL,(3,1)),VOLUME=SER=RTOSA,       X
//               DISP=(NEW,PASS),DSNAME=&LNKFILE
//SYSLIP DD   DSNAME=SYS1.MACLIB,DISP=OLD
```

NOTE: Ref. figure on pg. 4.4.101




```
//ASM     EXEC  PGM=IEUASM                                          X
//              *                                                   X
//              *ASSEMBLE STEP -- OTHER THAN FIRST                  X
//              *                                                   X
//              *FOLLOW-UP TO ASSEMBLE IN STACKED ASSEMBLY          X
//              *
//SYSPRINT DD   SYSOUT=A
//SYSUT1 DD     UNIT=DISK,SPACE=(CYL,(30,10)),VOLUME=SER=RTOSA
//SYSUT2   DD   UNIT=TAPE,LABEL=(,NL),SEP=(SYSUT1)
//SYSUT3 DD     UNIT=DISK,SPACE=(CYL,(50,10)),VOLUME=SER=RTOSB
//SYSGO  DD     UNIT=DISK,SPACE=(CYL,(3,1)),VOLUME=SER=RTOSA,       X
//              DISP=(NEW,PASS),DSNAME=&LNKFILE
//TIME   DD     SYSOUT=A
//SYSLIB DD   DSNAME=SYS1.MACLIB,DISP=OLD
```

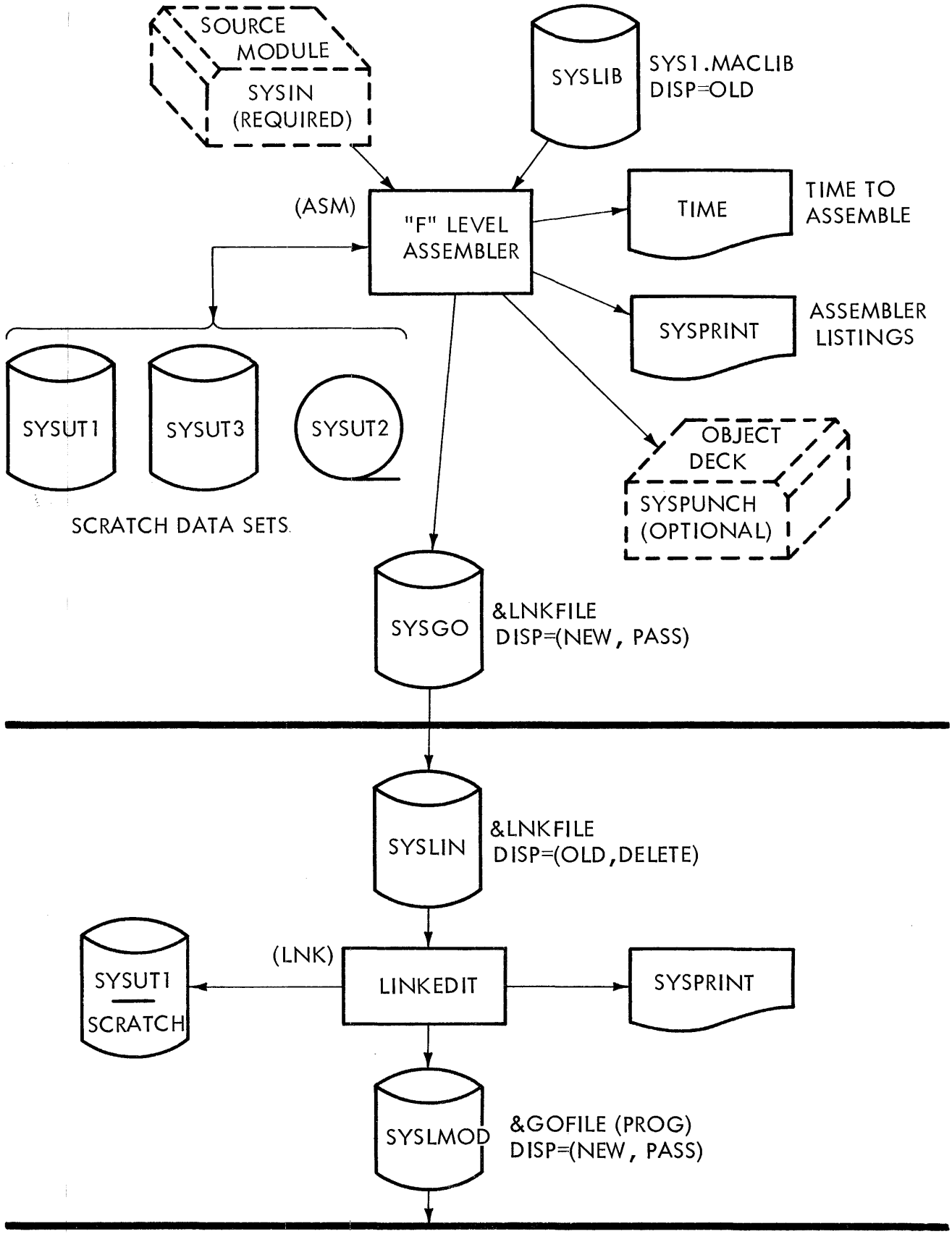
NOTE: Ref. figure on pg. 4.4.101

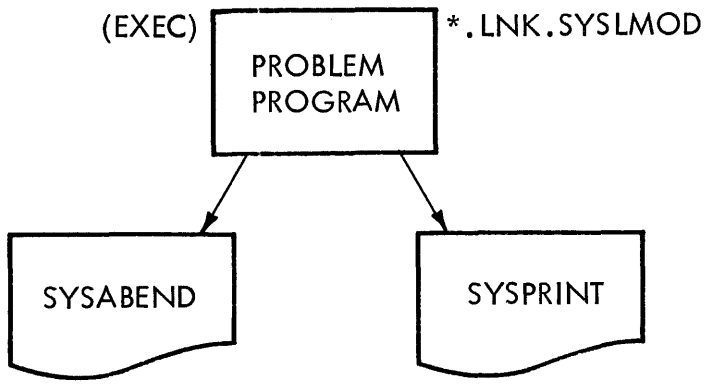



```
//LNK   EXEC  PGM=LINKEDIT,PARM='MAP,LIST,LET,NCAL,XREF,REUS'        X
//             *                                                     X
//             * LOADAND GO PROCEDURE USED AFTER  STACKED ASSEMBLY  X
//             *
//SYSUTI DD    UNIT=DISK,SPACE=(CYL,(R,1)),VOLUME=SER=RTOSA
//SYSPRINT DD  SYSOUT=A
//SYSLMOD DD DSNAME=&GOFILE(PROG),UNIT=DISK,DISP=(,PASS),           C
//             SPACE=(TRK,(8,1,1)),VOLUME=SER=RTOSB
//SYSLIN DD    UNIT=DISK,DISP=(OLD,DELETE),VOLUME=SER=RTOSA,        X
//             DSNAME=&LNKFILE,DCB=(RECFM=F,BLKSIZE=80)
//EXEC   EXEC  PGM=*.LNK.SYSLMOD
//SYSPRINT DD  SYSOUT=A
//SYSABEND  DD SYSOUT=A
```

NOTE: Ref. figure on pg. 4.4.101

ASMLDGO


SOURCE MODULE
SYSIN (REQUIRED)
SYSLIB
SYS1.MACLIB DISP=OLD
(ASM)
"F" LEVEL ASSEMBLER
TIME
TIME TO ASSEMBLE
SYSPRINT
ASSEMBLER LISTINGS
SYSUT1
SYSUT3
SYSUT2
SCRATCH DATA SETS
OBJECT DECK
SYSPUNCH (OPTIONAL)
SYSGO
&LNKFILE DISP=(NEW, PASS)
SYSLIN
&LNKFILE DISP=(OLD,DELETE)
(LNK)
LINKEDIT
SYSUT1 SCRATCH
SYSPRINT
SYSLMOD
&GOFILE (PROG) DISP=(NEW, PASS)




```
//ASM     EXEC  PGM=IEUASM                                          X
//              *                                                   X
//              *   ASSEMBLE,LOAD , AND GO PROCEDURE                X
//              *
//SYSPRINT DD   SYSOUT=A
//SYSUT1 DD     UNIT=DISK,SPACE=(CYL,(30,10)),VOLUME=SER=RTOSA
//SYSUT2   DD   UNIT=TAPE,LABEL=(,NL),SEP=(SYSUT1)
//SYSUT3 DD     UNIT=DISK,SPACE=(CYL,(50,10)),VOLUME=SER=RTOSB
//TIME    DD    SYSOUT=A
//SYSLIB DD   DSNAME=SYS1.MACLIB,DISP=OLD
//SYSGO   DD    UNIT=DISK,SPACE=(CYL,(3,1)),VOLUME=SER=RTOSA,       X
//              DISP=(NEW,PASS),DSNAME=&LNKFILE
//LNK   EXEC  PGM=LINKEDIT,PARM='MAP,LIST,LET,NCAL,XREF,REUS'       X
//              *                                                   X
//              *LINKEDIT PROCEDURE                                 X
//              *
//SYSUT1 DD     UNIT=DISK,SPACE=(CYL,(4,1)),VOLUME=SER=RTOSA
//SYSPRINT DD   SYSOUT=A
//SYSLMOD DD DSNAME=&GOFILE(PROG),UNIT=DISK,DISP=(,PASS),           C
//              SPACE=(TRK,(10,5,1)),VOLUME=SER=RTOSB
//SYSLIN DD     UNIT=DISK,DISP=(OLD,DELETE),VOLUME=SER=RTOSA,       X
//              DSNAME=&LNKFILE,DCB=(RECFM=F,BLKSIZE=80)
//EXEC    EXEC  PGM=*.LNK.SYSLMOD                                   X
//              *                                                   X
//              * EXECUTION PROCEDURE                               X
//              *
//SYSPRINT DD   SYSOUT=A
//SYSABEND DD   SYSOUT=A
```

NOTE: Ref. figure on pg. 4.4.101

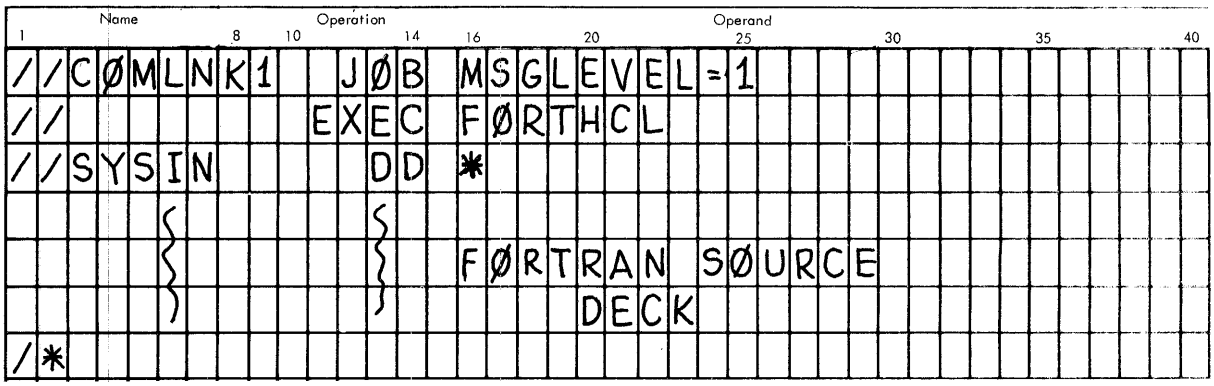
```
ASMEC
       //ASM     EXEC   PGM=IETASM
       //SYSLIB  DD     DSNAME=SYS1.MACLIB,DISP=OLD
       //SYSUT1  DD     UNIT=SYSSQ,SPACE=(400,(400,50))
       //SYSUT2  DD     UNIT=SYSSQ,SPACE=(400,(400,50))
       //SYSUT3  DD     UNIT=(SYSSQ,SEP=(SYSUT2,SYSUT1,SYSLIB)),
       //                SPACE=(400,(400,50))
       //SYSPRINT DD    SYSOUT=A
       //SYSPUNCH DD    UNIT=SYSCP


LKEDC
       //LKED      EXEC   PGM=IFWL,PARM='XREF,LIST,LET,NCAL'
       //SYSPRINT  DD     SYSOUT=A
       //SYSLIN    DD     DDNAME=SYSIN
       //SYSLMOD   DD     DSNAME=&GOSET(GO),SPACE=(1024,(50,20,1)),
       //                 UNIT=SYSDA,DISP=(MOD,PASS)
       //SYSUT1    DD     UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),
       //                 SPACE=(1024,(200,20))
       //GO        EXEC   PGM=*.LKED.SYSLMOD
                        SPACE

ASMECLG
       //ASM     EXEC   PGM=IETASM
       //SYSLIB  DD     DSNAME=SYS1.MACLIB,DISP=OLD
       //SYSUT1  DD     UNIT=SYSSQ,SPACE=(400,(400,50))
       //SYSUT2  DD     UNIT=SYSSQ,SPACE=(400,(400,50))
       //SYSUT3  DD     UNIT=(SYSSQ,SEP=(SYSUT2,SYSUT1,SYSLIB)),
       //                SPACE=(400,(400,50))
       //SYSPRINT DD    SYSOUT=A
       //SYSPUNCH DD    DSNAME=&LOADSET,UNIT=SYSSQ,SPACE=(80,(200,50)),
       //                DISP=(MOD,PASS)
       //LKED    EXEC   PGM=IEWL,PARM=(XREF,LET,LIST,NCAL)
       //SYSLIN  DD     DSNAME=&LOADSET,DISP=(OLD,DELETE)
       //        DD     DDNAME=SYSIN
       //SYSLMOD DD     DSNAME=&GOSET(GO),UNIT=SYSDA,SPACE=(1024,(50,20,1)),
       //                DISP=(MOD,PASS)
       //SYSUT1  DD     UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),
       //                SPACE=(1024,(50,20))
       //SYSPRINT DD    SYSOUT=A
       //GO      EXEC   PGM=*LKED.SYSLMOD
DATA FOR SDS OR MEMBER
```

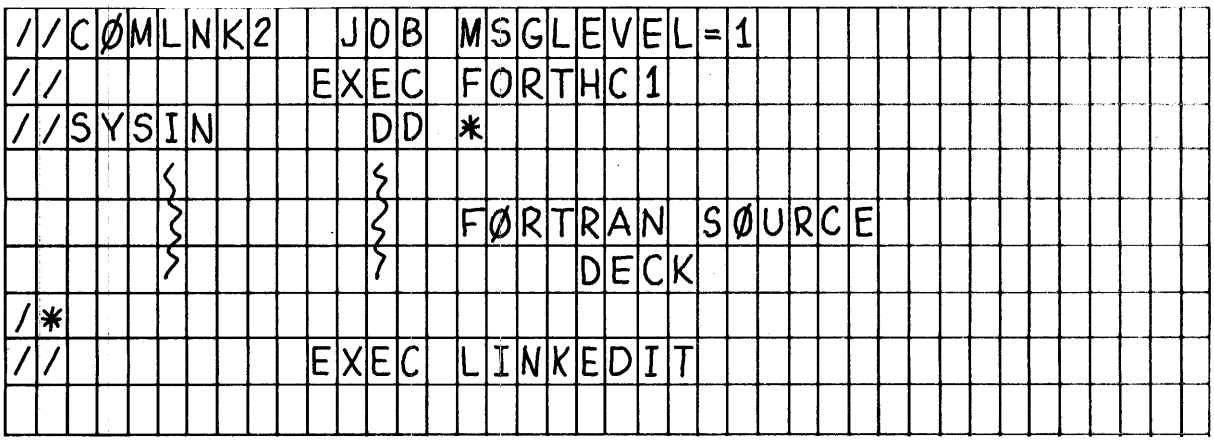
COBOL

```
 THE PROCLIB
NAME  COBEC
            //COB       EXEC PGM=IEPCBLOO
            //SYSPRINT  DD  SYSOUT=A
            //SYSPUNCH  DD  UNIT=SYSCP
            //SYSUT1    DD  UNIT=SYSSQ,SPLIT=92,CYL,(40.10))
            //SYSUT2    DD  UNIT=SYSSQ,SPLIT=4
            //SYSUT3    DD  UNIT=SYSSQ,SPLIT=4

 THE PROCLIB
NAME  COBELG
            //LKED      EXEC PGM=IEWL,PARM=(XREF,LIST,LET)
            //SYSLIN    DD  DDNAME=SYSIN
            //SYSLMOD   DD  DSNAME=&GODATA(RUN),DISP=(NEW,PASS),UNIT=STSDA,          X
            //                SPACE=(1024,(50,20,1))
            //SYSLIB    DD  DSNAME=SYS1,COBLIB,DISP=(OLD,KEEP)
            //SYSUT1    DD  UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),SPACE=(1024,(50,20,1))
            //SYSPRINT  DD  SYSOUT=A
            //GO        EXEC PGM=*.LKED.SYSLMOD,COND=(9,LT,LKED)
            //SYSABEND  DD  SYSOUT=A

 THE PROCLIB
NAME  COBECLG
            //COB       EXEC PGM=IEPCBLOO
            //SYSPRINT  DD  SYSOUT=A
            //SYSUT1    DD  UNIT=SYSSQ,SPLIT=(2,CYL,(40,10))
            //SYSUT2    DD  UNIT=SYSSQ,SPLIT=4
            //SYSUT3    DD  UNIT=SYSSQ,SPLIT=4
            //SYSPUNCH  DD  DSNAME=&LOADSET,DISP=(MOD,PASS),UNIT=SYSSQ,              X
            //                SPACE=(TRK,(50,10))
            //LKED      EXEC PGM=IEWL,PARM=(XREF,LIST,LET)
            //SYSLIN    DD  DSNAME=&LOADSET,DISP=(OLD,DELETE)
            //                DDNAME=SYSIN
            //SYSLMOD   DD  DSNAME=&GODATA(RUN),DISP=(NEW,PASS),UNIT=SYSDA,          X
            //                SPACE=(1024,(50,20,1))
            //SYSLIB    DD  DSNAME=SYS1.COBLIB,DISP=(OLD,KEEP)
            //SYSUT1    DD  UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),SPACE=(1024,(50,20,1))
            //SYSPRINT  DD  SYSOUT=A
            //GO        EXEC PGM=*.LKED.SYSLMOD,COND=(9,LT,LKED)
            //SYSABEND  DD  SYSOUT=A
DATA FOR SDS OR MEMBER
```

PROGRAM COMPILE AND CREATE A LOAD MODULE

ØNE ØBJECT INTØ ØNE LØAD M | DATF





PROGRAM CØMPILE AND CREATE A LØAD MØDULE NAMED PRGMØD2
ØNE ØBJECT INTØ ØNE LØAD MØDULE

DATE

```
//CØMLNK3   JOB MSGLEVEL=1
//        EXEC FORTHCL
//SYSIN     DD *
      {         {
      {         {  FØRTRAN SØURCE
      {         {      DECK
/*
//LNK.SYSLMOD DD DSNAME=&GØFILE(PRGMØD2)
```

```
//COMLNK4   JOB MSGLEVEL=1
//        EXEC FØRTHCL
//SYSIN     DD
      {         {
      {         {  FØRTRAN SØURCE
      {         {      DECK
/*
//LNK.SYSLMØD DD DSNAME=&GØFILE
//SYSIN     DD *
         NAME   PRGMØD2
/*
```

PROGRAM CØMPILE(MULTIPLE) AND CREATE ØNE LØAD MØDULE

MULTIPLE ØBJECT INTØ ØNE LØAD M. | DATE

Name | Operation | Operand

```
//COMULT1  JOB  MSGLEVEL=1
//        EXEC  FORTHC1
//SYSIN     DD  *
                FØRTRAN SØURCE #1
/*
//        EXEC  FORTHCN
//SYSIN     DD  *
                FØRTRAN SØURCE #2
/*
//        EXEC  FORTHCN
//SYSIN     DD  *
                FØRTRAN SØURCE #3
/*
//        EXEC  LINKEDIT
```

PROGRAM CØMPILE (MULTIPLE) AND CREATE LØAD MØDULE PRGMØD3
MULTIPLE ØBJECT INTØ ØNE LØAD M.

DATE

```
Name          Operation         Operand
1      8  10     14  16    20    25     30    35    40
//CØMULT2  JØB MSGLEVEL=1
//        EXEC FØRTHC1
//SYSIN     DD *
    }        }
    }        }  FØRTRAN SØURCE #1
    }        }
/*
//        EXEC FØRTHCN
//SYSIN     DD *
    }        }
    }        }  FØRTRAN SØURCE #2
    }        }
/*
//        EXEC LINKEDIT
//LNK.SYSLMØD DD DSNAME=&GØFILE(PRGMØD3)
```

PROGRAM COMPILE, LINKEDIT, AND EXECUTE
ONE OBJECT, ONE LOAD MODULE

DATE

```
//COMLNKG1 JOB  MSGLEVEL=1
//         EXEC FORTHCLG
//SYSIN    DD   *
    ~        ~
    ~        ~  FORTRAN SOURCE
    ~        ~      DECK
/*
```

```
//COMLNKG2 JOB  MSGLEVEL=1
//         EXEC FORTHC1
//SYSIN    DD   *
    ~        ~
    ~        ~  FORTRAN SOURCE
    ~        ~      DECK
/*
//         EXEC LINKEDIT
//         EXEC EXECSTEP
```

PROGRAM CØMPILE, LINKEDIT, AND EXECUTE - MODULE = PRGMØD5

ØNE ØBJECT INTØ ØNE LØAD MØDULE | DATE

```
//CØMLNKG3  JØB  MSGLEVEL=1
//          EXEC FØRTHCLG
//SYSIN     DD
                 FØRTRAN SØURCE
                 DECK
/*
//LNK.SYSLMØD DD  DSNAME=&GØFILE(PRGMØD5)
```

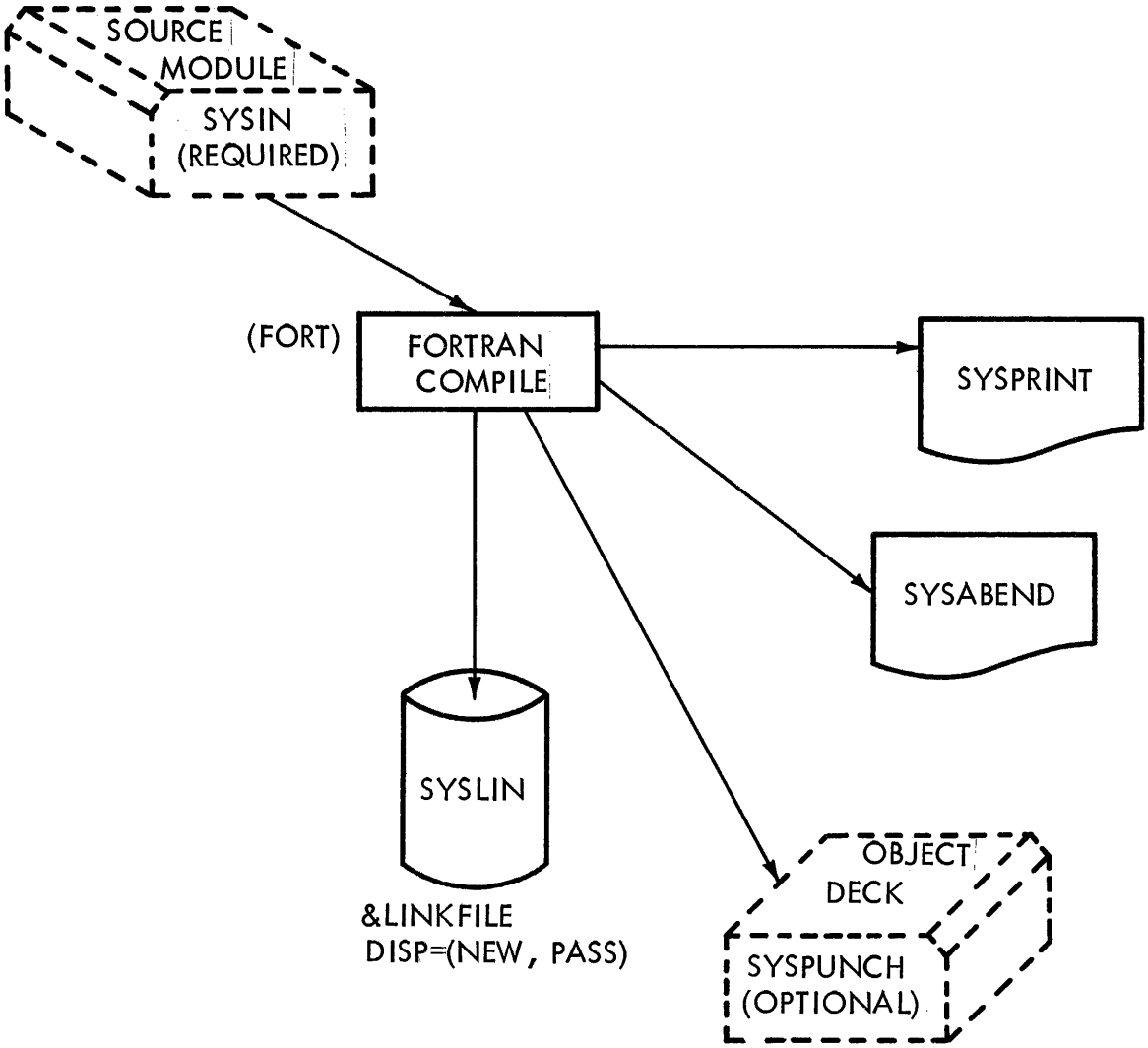
```
//CØMLNKG4  JØB  MSGLEVEL=1
//          EXEC FØRTHC1
//SYSIN     DD   *
                 FØRTRAN SØURCE
                 DECK
/*
//          EXEC LINKEDIT
//SYSLMØD   DD   DSNAME=&GØFILE(PRGMØD5)
//LNK.SYSLMØD DD  DSNAME=&GØFILE(PRGMØD5)
//          EXEC EXECSTEP
```

PROGRAM CØMPILE FØRTRAN WITH ASSEMBLER & LINKEDIT MULTIPLE ØBJECT INTØ ØNE LØAD M.

DATE

```
//FØRTASM1  JØB MSGLEVEL=1
//          EXEC FØRTHC1
//SYSIN       DD  *

                  FØRTRAN SØURCE
                      DECK #1
/*
//          EXEC ASSEMBLN
//SYSIN       DD  *

                  ASSEMBLER SØURCE
                      DECK #1
/*
//          EXEC FØRTHCN
//SYSIN       DD  *

                 FØRTRAN SØURCE
                     DECK #2
/*
//          EXEC ASSEMBLN
//SYSIN       DD  *

                 ASSEMBLER SØURCE
/*                   DECK #2
//          EXEC LINKEDIT
```

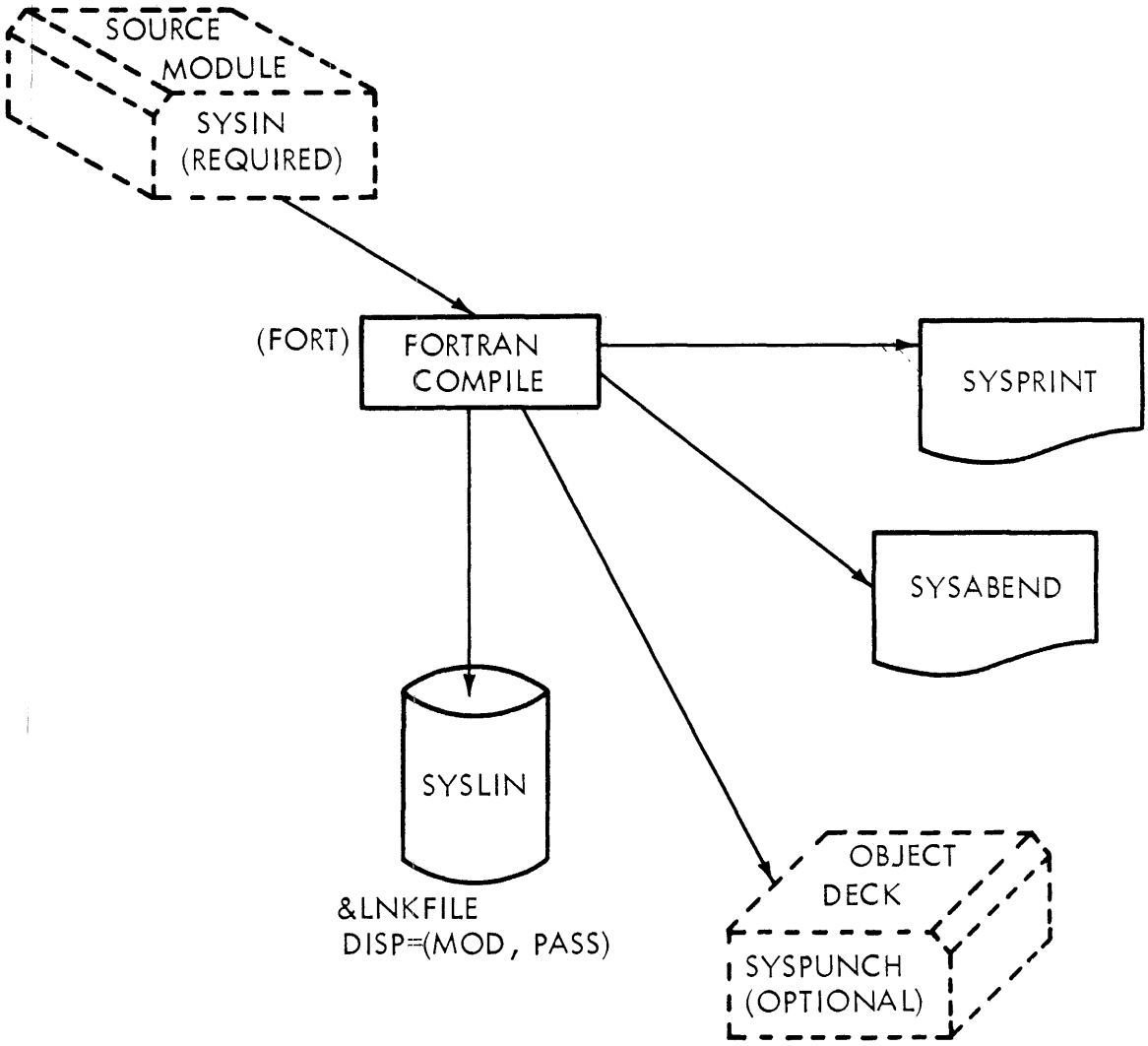
FORTHC1




```
//FORT    EXEC  PGM=FORTRAN,PARM='LIST,MAP'                          X
//              *
//              *  FIRST COMPILATION PROCEDURE IN FORTRAN H          X
//              *  STACKED COMPILATION                               X
//              *
//SYSLIN   DD  DSNAME=&LNKFILE,DISP=(NEW,PASS),UNIT=DISK,            X
//               VOLUME=SER=RTOSA,SPACE=(TRK,(10,5))
//SYSPRINT DD  SYSOUT=A
//SYSABEND DD  SYSOUT=A
```

NOTE: Ref. figure on pg. 4.4.101

FORTHCN

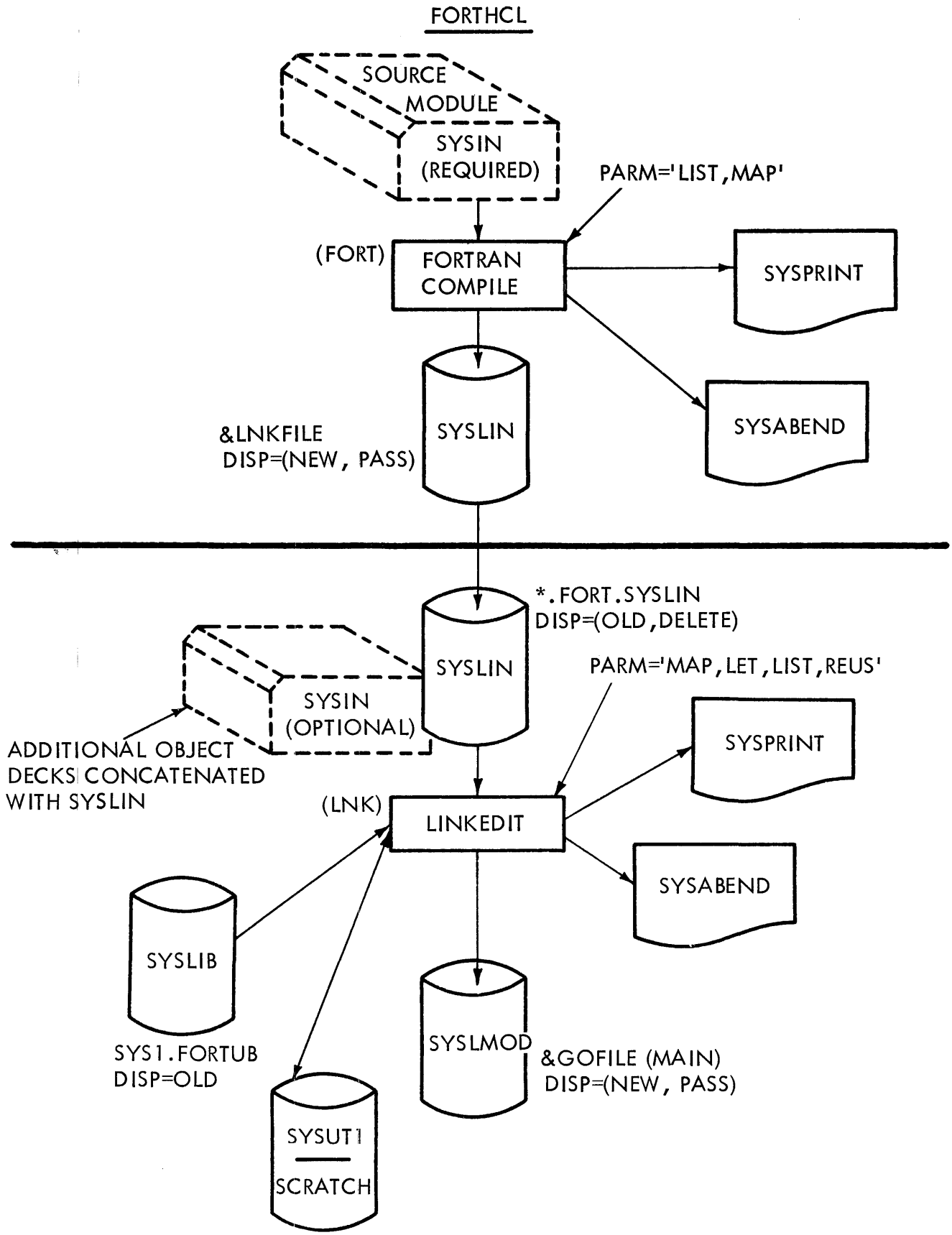



```
//FORT    EXEC  PGM=FORTRAN,PARM='LIST,MAP'                              X
//              *
//              * FORTRAN R COMPILATION STEP--OTHER THAN FIRST            X
//              *
//SYSLIN   DD   DSNAME=&LNKFILE,DISP=(MOD,PASS),UNIT=DISK,               X
//               VOLUME=SER=RTOSA,SPACE=(TRK,(10,5))
//SYSPRINT DD   SYSOUT=A
//SYSABEND DD   SYSOUT=A
```

NOTE: Ref. figure on pg. 4.4.101

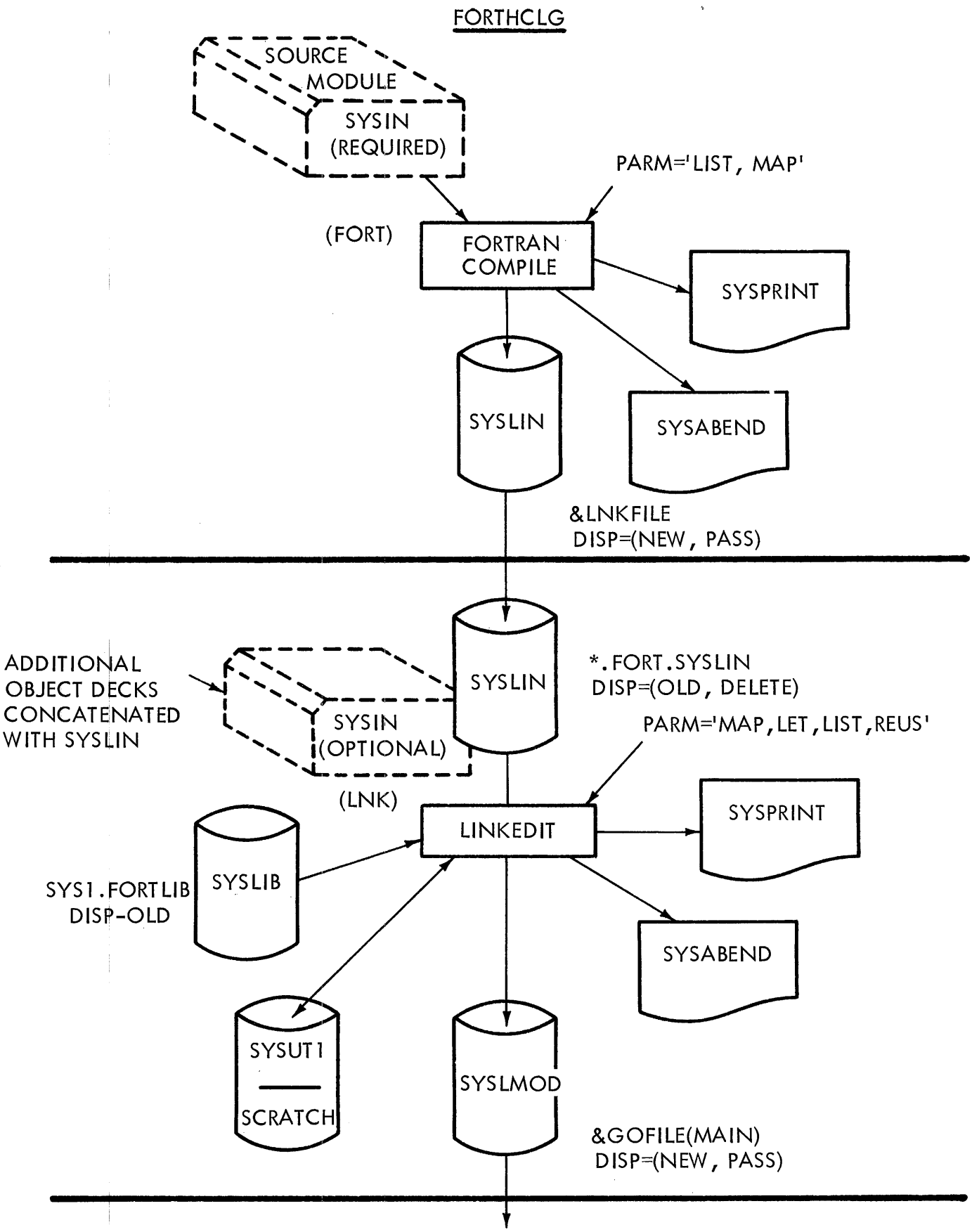
```
//FORT    EXEC  PGM=FORTRAN,PARM='LIST,MAP'                          X
//              *
//              * FORTRAN H COMPILE AND LINKEDIT PROCEDURE           X
//              *
//SYSLIN   DD   DSNAME=&LNKFILE,DISP=(NEW,PASS),UNIT=DISK,           X
//               VOLUME=SER=RTOSA,SPACE=(TRK,(10,5))
//SYSPRINT DD   SYSOUT=A
//SYSABEND DD   SYSOUT=A
//LNK     EXEC  PGM=LINKEDIT,PARM='MAP,LET,LIST,REUS'
//SYSLIB DD     DSNAME=SYS1.FORTLIB,DISP=OLD
//SYSPRINT DD   SYSOUT=A
//SYSABEND DD   SYSOUT=A
//SYSUT1   DD   UNIT=DISK,SPACE=(TRK,(10,5)),VOLUME=SER=RTOSA
//SYSLMOD   DD  DSNAME=&GOFILE(MAIN),DISP=(,PASS),UNIT=DISK,         X
//               SPACE=(TRK,(10,5,1)),VOLUME=SER=RTOSB
//SYSLIN   DD   DSNAME=*.FORT.SYSLIN,DISP=(OLD,DELETE),              X
//               VOLUME=REF=*.FORT.SYSLIN
//         DD    DDNAME=SYSIN
```

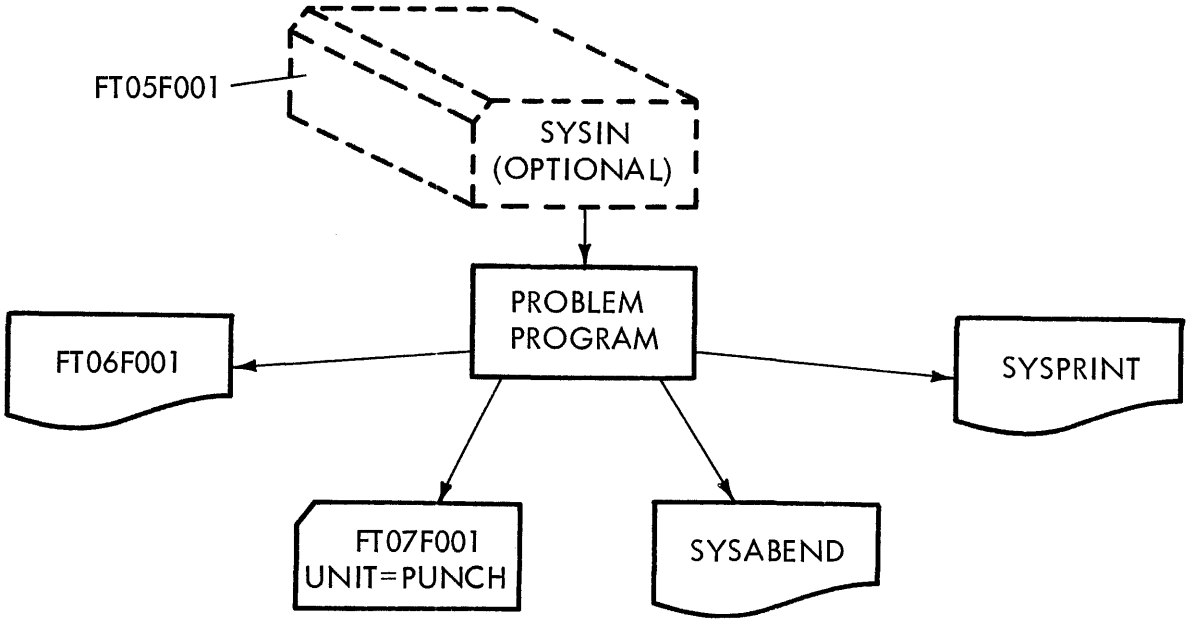
NOTE: Ref. figure on pg. 4.4.101


FORTHCL
SOURCE MODULE
SYSIN (REQUIRED)
PARM='LIST,MAP'
(FORT)
FORTRAN COMPILE
SYSPRINT
SYSABEND
&LNKFILE DISP=(NEW, PASS)
SYSLIN
*.FORT.SYSLIN DISP=(OLD,DELETE)
SYSLIN
PARM='MAP,LET,LIST,REUS'
SYSIN (OPTIONAL)
ADDITIONAL OBJECT DECKS CONCATENATED WITH SYSLIN
SYSPRINT
(LNK)
LINKEDIT
SYSABEND
SYSLIB
SYS1.FORTUB DISP=OLD
SYSLMOD
&GOFILE (MAIN) DISP=(NEW, PASS)
SYSUT1
SCRATCH

```
//LNK     EXEC  PGM=LINKEDIT,PARM='MAP,LET,LIST,REUS'                  X
//              *
//              * FORTRAN H LOAD AND GO PROCEDURE                      X
//              *
//SYSLIB DD     DSNAME=SYS1.FORTLIB,DISP=OLD
//SYSLMOD  DD   DSNAME=&GOFILE(MAIN),DISP=(,PASS),UNIT=DISK,          X
//              SPACE=(TRK,(10,5,1)),VOLUME=SER=RTOSE
//SYSUT1 DD     UNIT=DISK,SPACE=(TRK,(10,5)),VOLUME=SER=RTOSA
//SYSPRINT DD   SYSOUT=A
//SYSABEND DD SYSOUT=A
//SYSLIN DD     UNIT=DISK,VOLUME=SER=RTOSA,DSNAME=&LNKFILE,           X
//              DISP=(OLD,DELETE)
//        DD    DDNAME=SYSIN
//EXEC  EXEC  PGM=*.LNK.SYSLMOD
//SYSABEND DD   SYSOUT=A
//SYSPRINT DD SYSOUT=A
//FT05F001 DD   DDNAME-SYSIN
//FT06F001 DD   SYSOUT=A
//FT07F001   DD   UNIT=PUNCH
```

NOTE: Ref. figure on pg. 4.4.101


FORTHCLG
SOURCE MODULE
SYSIN (REQUIRED)
PARM='LIST, MAP'
(FORT)
FORTRAN COMPILE
SYSPRINT
SYSLIN
SYSABEND
&LNKFILE
DISP=(NEW, PASS)
ADDITIONAL OBJECT DECKS CONCATENATED WITH SYSLIN
SYSIN (OPTIONAL)
SYSLIN
*.FORT.SYSLIN
DISP=(OLD, DELETE)
PARM='MAP,LET,LIST,REUS'
(LNK)
LINKEDIT
SYSPRINT
SYS1.FORTLIB
DISP-OLD
SYSLIB
SYSABEND
SYSUT1
SCRATCH
SYSLMOD
&GOFILE(MAIN)
DISP=(NEW, PASS)




```
//FORT    EXEC  PGM=FORTRAN,PARM='LIST,MAP'                          X
//              *                                                    X
//              * FORTRAN H COMPILE,LOAD, AND GO PROCEDURE           X
//              *
//SYSLIN   DD   DSNAME=&LNKFILE,DISP=(NEW,PASS),UNIT=DISK,           X
//               VOLUME=SER=RTOSA,SPACE=(TRK,(10,5))
//SYSPRINT DD   SYSOUT=A
//SYSABEND DD   SYSOUT=A
//LNK     EXEC  PGM=LINKEDIT,PARM='MAP,LET,LIST,REUS'
//SYSLIB DD     DSNAME=SYS1.FORTLIB,DISP=OLD
//SYSLMOD  DD   DSNAME=&GOFILE(MAIN),DISP=(,PASS),UNIT=DISK,         X
//              SPACE=(TRK,(10,5,1)),VOLUME=SER=RTOSB
//SYSUT1 DD     UNIT=DISK,SPACE=(TRK,(10,5)),VOLUME=SER=RTOSA
//SYSPRINT DD   SYSOUT=A
//SYSABEND DD   SYSOUT=A
//SYSLIN   DD   DSNAME=*.FORT.SYSLIN,LISP=(OLD,DELETE),              X
//              VOLUME=REF=*.FORT.SYSLIN
//        DD    DDNAME=SYSIN
//EXEC   EXEC   PGM=*.LNK.SYSLMOD
//SYSABEND DD   SYSOUT=A
//SYSPRINT DD SYSOUT=A
//FT05F001 DD   DDNAME=SYSIN
//FT06F001 DD   SYSOUT=A
//FT07F001   DD   UNIT=PUNCH
```

NOTE: Ref. figure on pg. 4.4.101

```
 THE PROCLIB
NAME  FORTEC
            //FORT     EXEC PGM=IEJFAAAO
            //SYSPRINT DD   SYSOUT=A
            //SYSPUNCH DD   UNIT=SYSCP
            //SYSUT1   DD   UNIT=SYSSO,SEP=SYSPUNCH,SPACE=(TRK,(30,10))
            //SYSUT2   DD    UNIT=SYSSQ,SEP=SYSUT1,SPACE=(TRK,(30,10))
            //SYSLIN DD    DSNAME=&LOADSET,DISP=(MOD,PASS),UNIT=SYSSQ,          1
            //              SEP=SYSPUNCH,SPACE=(TRK,(30,10)
```

```
 THE PROCLIB
NAME  FORTELG
            //LKFD     EXEC PGM=IEWL,PARM=(XREF,LIST,LET)
            //SYSLIB   DD   DSNAME=SYS1.FORTLIB,DISP=OLD
            //SYSLMOD  DD   DSNAME=&GOSFT(MAIN),DISP=(NEW,PASS),UNIT=SYSDA,       X
            //             SPACE=(1024,(50,20,1))
            //SYSUT1   DD   UNIT=SYSDA,SEP=(SYSLMOD,SYSLIB),SPACE=(1024,(30,20))
            //SYSPRINT DD   SYSOUT=A
            //SYSLIN   DD   DDNAME=SYSIN
            //GO       EXEC PGM=*.LKED.SYSLMOD,COND=(5,LT,LKED)
            //FT02F001 DD   UNIT=SYSCP
            //FT03F001 DD   SYSOUT=A
            //FT01F001 DD   DDNAME=SYSIN
```

```
 THE PROCLIB
NAME  FORTECLG
            //FORT     EXEC PGM=IEJFAAAO
            //SYSPRINT DD   SYSOUT=A
            //SYSPUNCH DD   UNIT=SYSCP
            //SYSUT1   DD   UNIT=SYSSQ,SEP=SYSPUNCH,SPACE=(TRK,(30,10))
            //SYSUT2   DD   UNIT=SYSSQ,SEP=SYSUT1,SPACE=(30,10))
            //SYSLIN DD    DSNAME=&LOADSET,DISP=(MOD,PASS),UNIT=SYSSQ,          1
            //              SEP=SYSPUNCH,SPACE=(YRK,(30,10))
            //LKED     EXEC PGM=IEWL,PARM=(XREF,LIST,LET),COND=(5,LT,FORT)
            //SYSLIB   DD   DSNAME=SYS1.FORTLIB,DISP=OLD
            //SYSLMOD  DD   DSNAME=&GOSFT(MAIN),DISP=(NEW,PASS),UNIT=SYSDA,       X
            //             SPACE=(1024,(50,20,1))
            //SYSUT1   DD   UNIT=SYSDA,SEP=(SYSLMOD,SYSLIB),SPACE=(1024,(30,20))
            //SYSPRINT DD   SYSOUT=A
            //SYSLIN DD    DSNAME=*.FORT.SYSLIN,DISP=(OLD,DELETE),DCB=(RECFM=F,  1
            //             BLKSIZE=80),UNIT=SYSSQ
            //          DD   DDNAME=SYSIN
            //GO       EXEC PGM=*.LKED.SYSLMOD,COND=((5,LT,FORT),(5,LT,LKED))
            //FT02F001 DD   UNIT=SYSCP
            //FT03F001 DD   SYSOUT=A
            //FT01F001 DD   DDNAME=SYSIN
DATA FOR SDS OR MEMBER
```

PROGRAM BUILD JØBLIB & EXECUTE "SECØND" MØDULE
2 LØAD MØDULES

DATE

```
//LIBBLD1   JØB MSGLEVEL=1
//JØBLIB      DD DSNAME=&GOFILE,UNIT=DISK,          C
//               DISP=(NEW,PASS),                   C
//               VØLUME=SER=RTØ5B,                  C
//               SPACE=(CYL,(2,1,2))
//          EXEC ASSEMBL1
//SYSIN       DD *
                 ASSEMBLER SØURCE DECK
/*
//ØNE       EXEC LINKEDIT
//LNK,SYSLMØD DD DSNAME=&GØFILE(FIRST),             C
//               DISP=(MOD,PASS)
//          EXEC FØRTHCL
//SYSIN       DD *
                 FØRTRAN SØURCE DECK
/*
//LNK.SYSLMØD DD DSNAME=&GØFILE(SECØND),            C
//               DISP=(MØD,PASS)
//          EXEC EXECSTEP
```

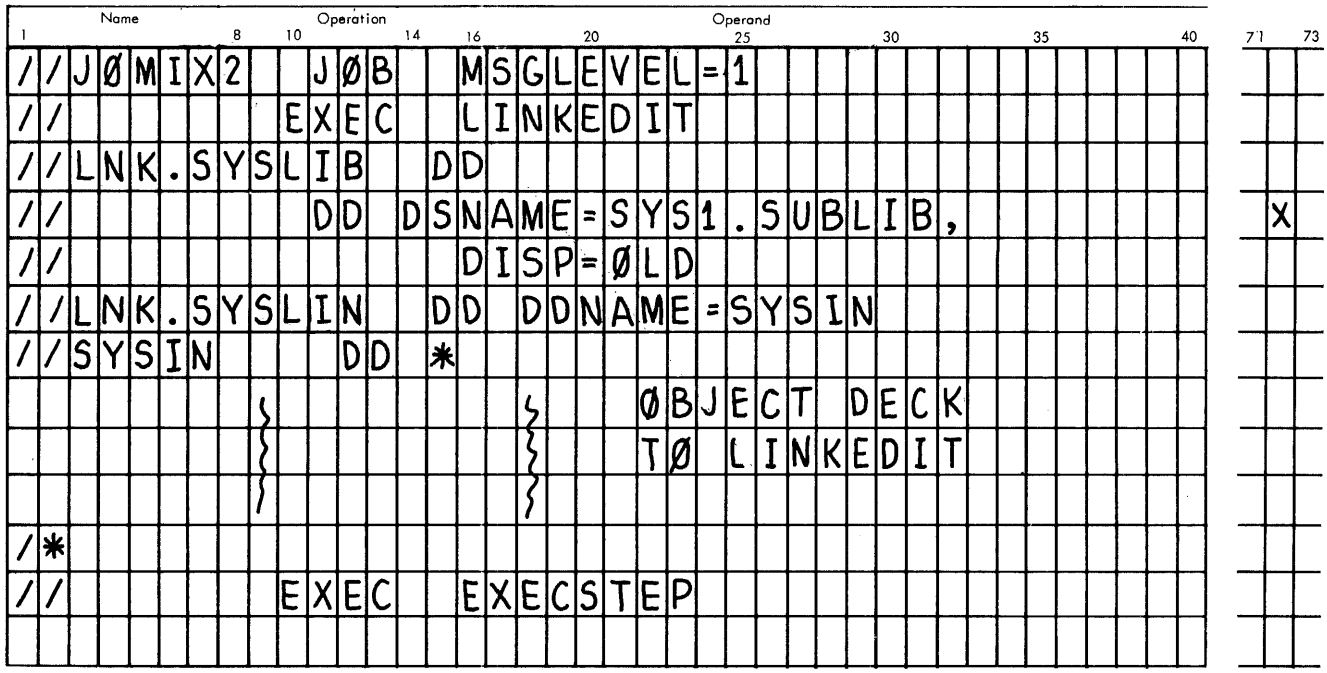
PROGRAM BUILD A LOAD MODULE AND EXECUTE USING PREVIOUSLY CREATED MODULES

DATE

```
//LIBBLD2  JØB  MSGLEVEL=1
//JØBLIB   DD   DSNAME=SMA.MISC.UTIL,                              C
//              DISP=ØLD,                                          C
//              VØLUME=SER=SMA205
//         DD   DSNAME=SMA.GØ,                                     C
//              DISP=ØLD,                                          C
//              VØLUME=SER=SMA205
//         EXEC ASSEMBL1
//SYSIN    DD   *
     }      }
     }      }     ASSEMBLER
     }      }   SØURCE MØDULE
/*
//         EXEC LINKEDIT
//         EXEC EXECSTEP
```

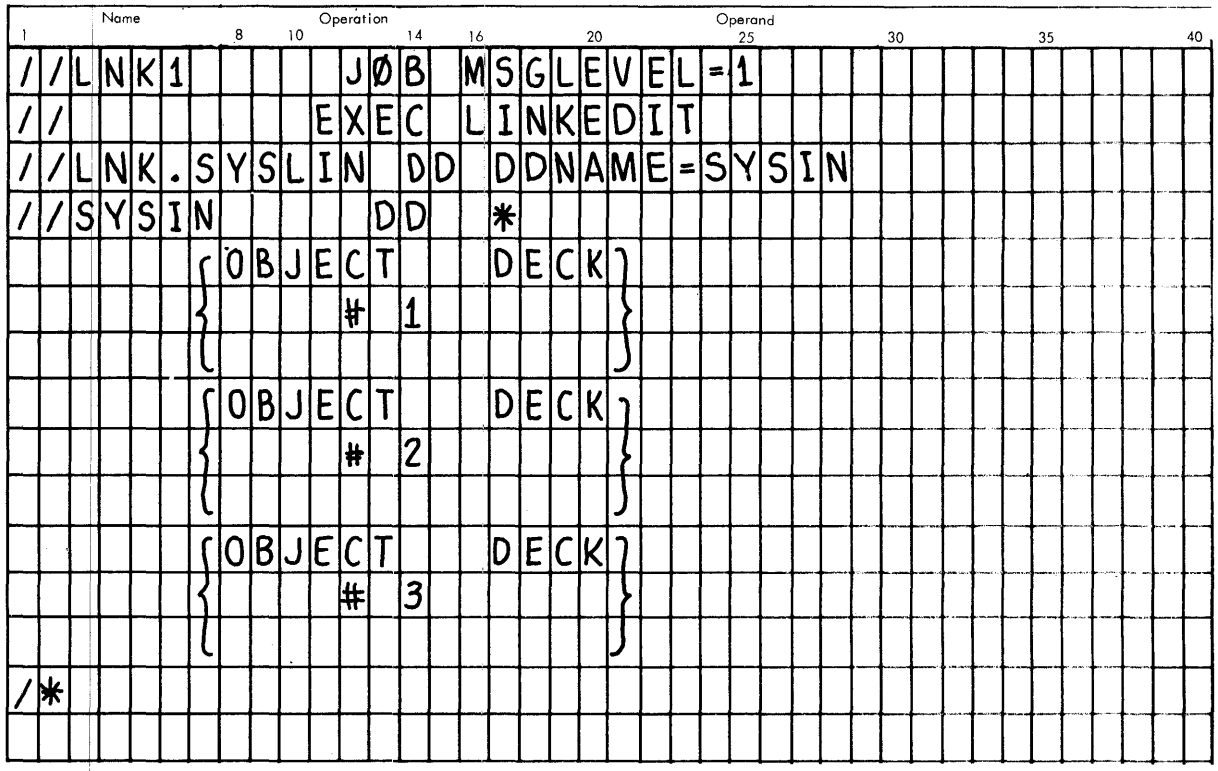
PROGRAM USE SPECIAL AUTO-CALL SUBLIB AND "DSNAME" ØPTIØN

DATE



PROGRAM LINKEDIT-MULTIPLE ØBJECT MODULES INTO ONE LOAD MODULE

DATE



PROGRAM MULTIPLE LOAD MODULES - USE OF "NAME

DATE

```
//LNK2       JØB  MSGLEVEL=1
//           EXEC LINKEDIT
//LNK.SYSLMOD DD  DSNAME=&GØFILE
//LNK.SYSLIN  DD  DSNAME=SYSIN
//SYSIN       DD  *
       { ØBJECT   }
       {   DECK   }
       {     #1   }
       { ØBJECT   }
       {   DECK   }
       {     #2   }
         NAME     FIRST
       { ØBJECT   }
       {   DECK   }
       {     #3   }
         NAME     SECØND
/*
```

PROGRAM MULTIPLE LOAD MODULES - USE OF INCLUDE

DATE

```
//LNK3      JØB   MSGLEVEL=1
//          EXEC  LINKEDIT
//LNK.SYSLMØD   DD  DSNAME=BØGSLIB,DISP=(NEW,KEEP)
//LNK.SYSIN     DD  DDNAME=SYSIN
//LNK.DUMMY1    DD  DSNAME=SMA.GØ,DISP=ØLD,              C
//                  VØLUME=SER=SMA205
//SYSIN         DD  *
          { ØBJECT }
          {   DECK }
          {    #1  }
           ALIAS  JAZZ
           NAME   FIRST
          { ØBJECT }
          {   DECK }
          {    #2  }
           INCLUDE DUMMY1(SECØND)
           NAME   SECØND
/*
```

PROGRAM EXECUTION WITH TESTRAN & POSTEDIT IN 1 JOB

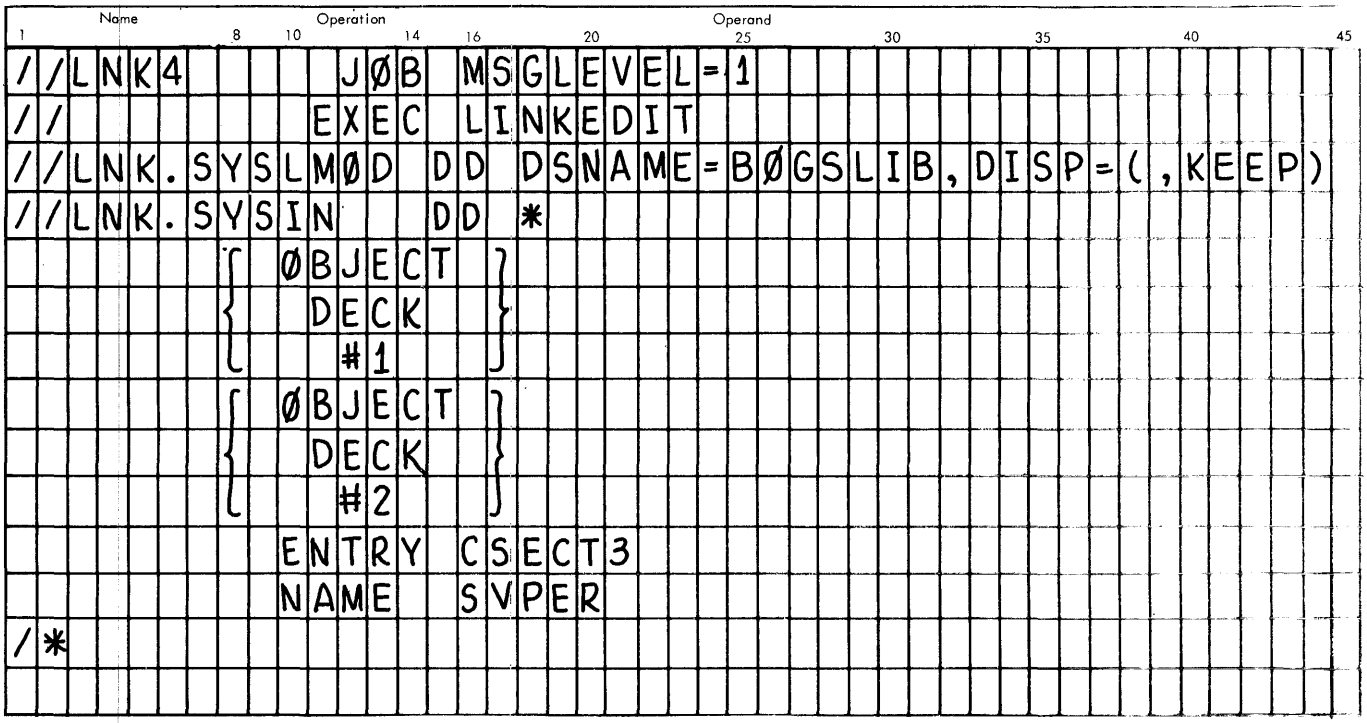
DATE [illegible]

Name | Operation | Operand

```
//TESTR1   JØB MSGLEVEL=1
//        EXEC LINKEDIT
//SYSIN     DD *
      { ØBJECT }
      {  DECK  }
      {        }
/*
//        EXEC EXECTEST
```

PROGRAM USE OF ENTRY

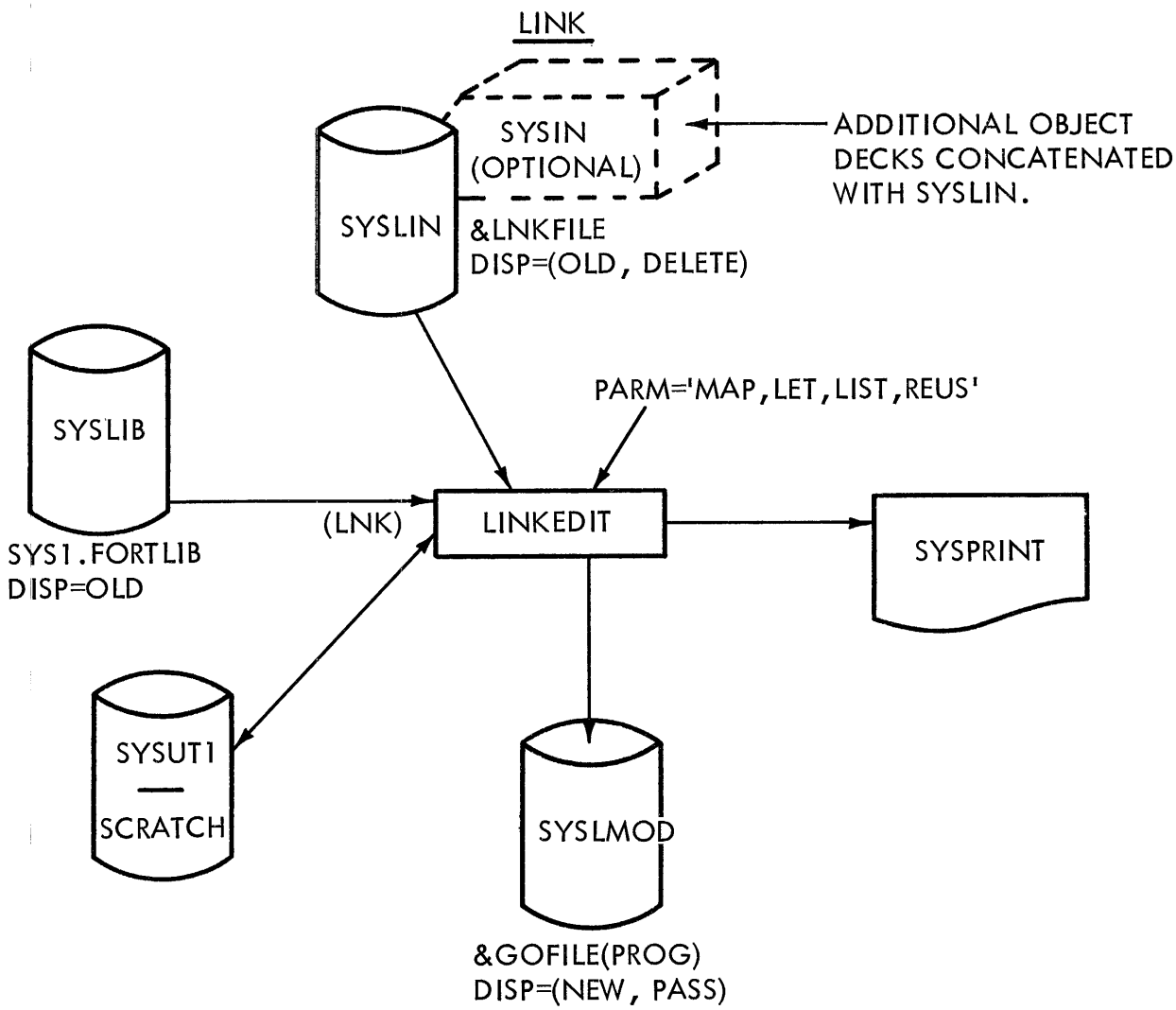
DATE



PROGRAM EXECUTION WITH TESTRAN & PØSTEDIT
IN A SECOND SEPARATE JOB

DATE

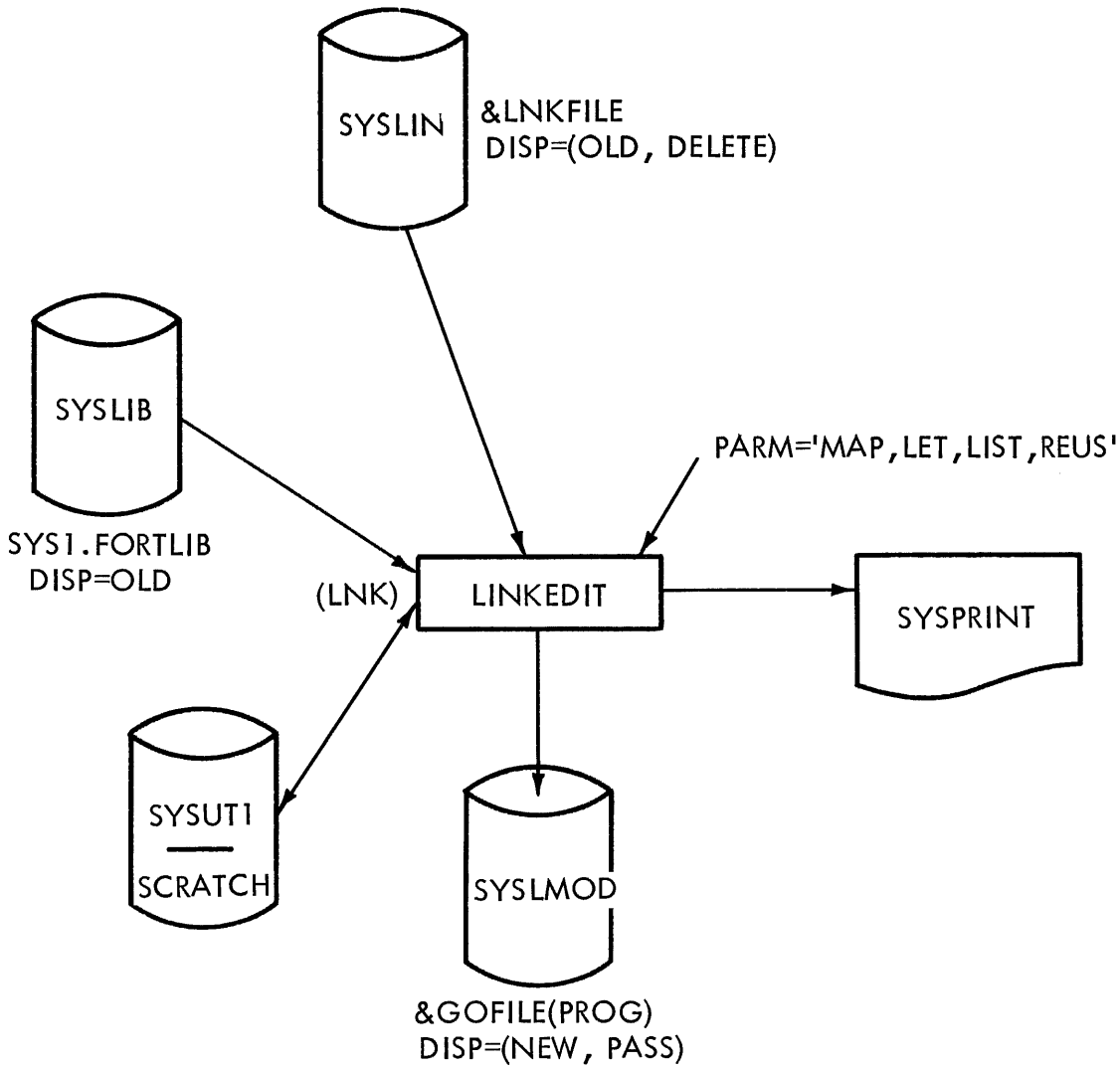
```
//TESTR2    JØB MSGLEVEL=1
//        EXEC LINKEDIT
//LNK.SYSLIN  DD  *
       {  ØBJECT  }
       {   DECK   }
       {          }
/*
//        EXEC EXECSTEP
//EXEC.SYSTEST  DD DISP=(NEW,KEEP),DSNAME=BØGSTEST
//TESTR3    JØB MSGLEVEL=1
//        EXEC TESTEDIT
//EDT.SYSTEST  DD DSNAME=BØGSTEST,VØLUME=SER=RTØSA
```




```
//LNK     EXEC    PGM=LINKEDIT,PARM='MAP,LET,LIST,REUS'                X
//                *                                                    X
//                * LINKEDIT FROM CARD READER AND DISK                 X
//                *
//SYSLMOD DD      DSNAME=&GOFILE(PROG),UNIT=DISK,DISP(,PASS),          X
//                VOLUME=SER=RTOSB,SPACE=(CYL,(2,1,2))
//SYSLIB  DD      DSNAME=SYS1.FORTLIB,DISP=OLD,UNIT=DISK

//SYSUT1 DD       UNIT=DISK,SPACE=(CYL,(4,1)),VOLUME=SER=RTOSA
//SYSPRINT DD     SYSOUT=A
//SYSLIN DD       UNIT=DISK,DISP=(OLD,DELETE),VOLUME=SER=RTOSA,        X
//                DSNAME=&LNKFILE,DCB=(,RECFM=F,BLKSIZE=80)
//        DD      DDNAME=SYSIN
```

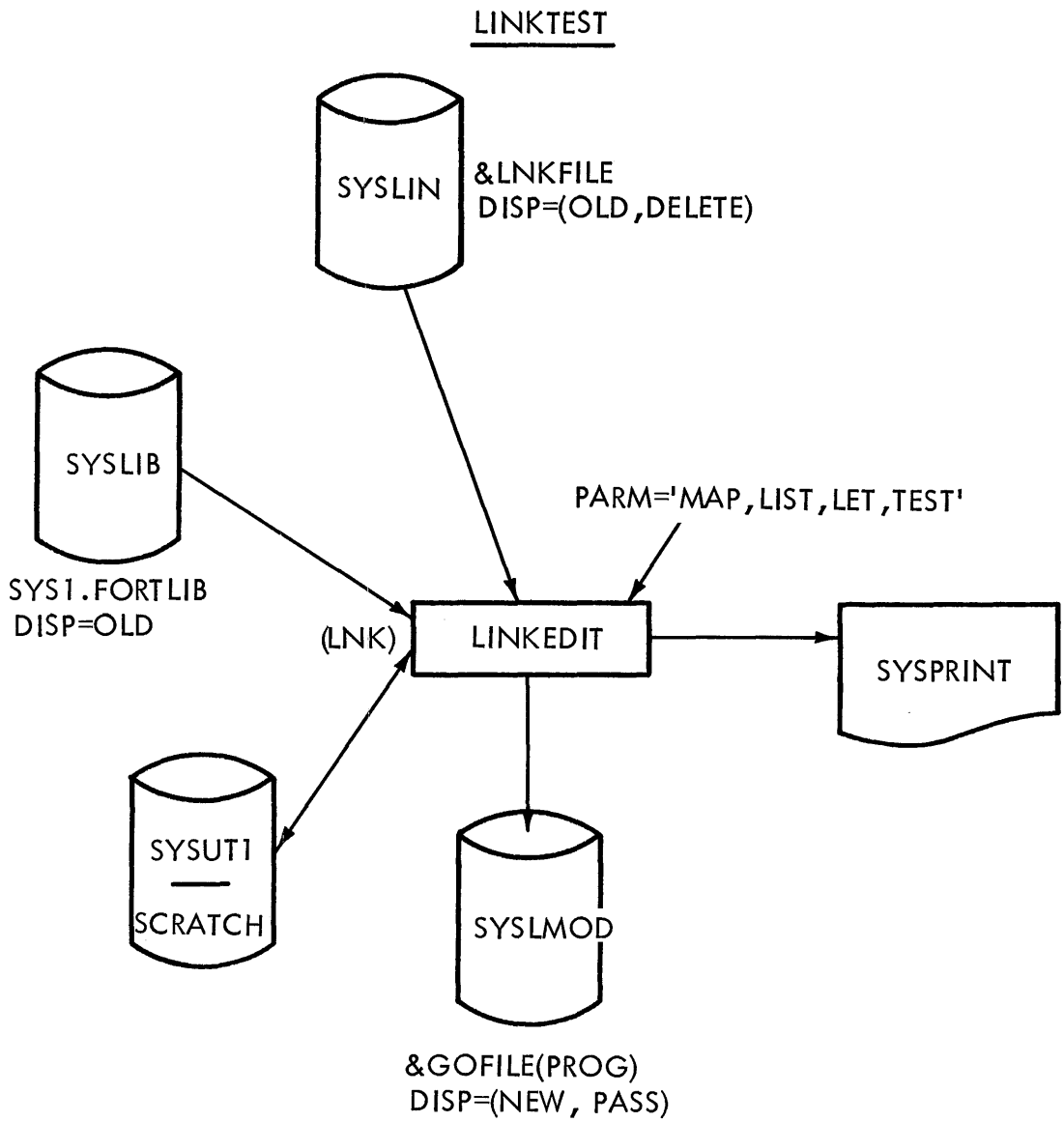
NOTE: Ref. figure on pg. 4.4.101

LINKEDIT




```
//LNK     EXEC  PGM=LINKEDIT,PARM='MAP,LET,LIST,REUS'                    X
//              *                                                         X
//              *    LINKEDIT  STEP                                       X
//              *
//SYSLMOD  DD   DSNAME=&GOFILE(PROG),UNIT=DISK,DISP=(,PASS),             X
//              VOLUME=SER=RTOSB,SPACE=(CYL,(2,1,2))
//SYSLIB   DD  DSNAME=SYS1.FORTLIB,DISP=OLD
//SYSPRINT  DD  SYSOUT=A
//SYSUTI DD     UNIT=DISK,SPACE=(CYL,(4,1)),VOLUME=SER=RTOSA
//SYSLIN DD     UNIT=DISK,DISP=(OLD,DELETE),VOLUME=SER=RTOSA,            X
//              DSNAME=&LNKFILE,DCB=(,RECFM=F,BLKSIZE=80)
```

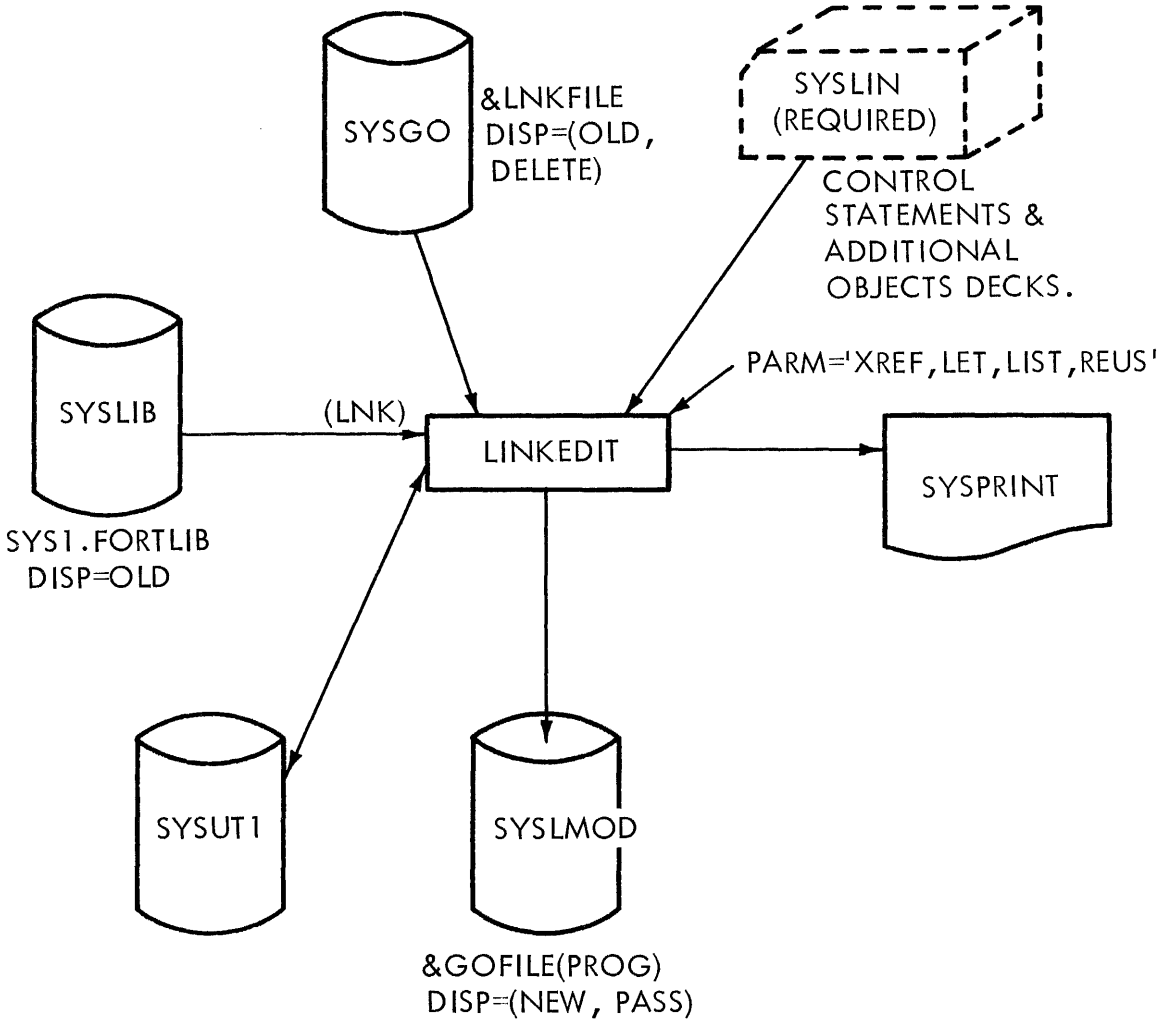
NOTE: Ref. figure on pg. 4.4.101




```
//LNK     EXEC  PGM=LINKEDIT,PARM='MAP,LIST,LET,TEST'                X
//              *                                                     X
//              *LINKEDIT STEP WITH TESTRAN                           X
//              *
//SYSLMOD DD DSNAME=&GOFILE(PROG),UNIT=DISK,DISP=(,PASS),             C
//              SPACE=(TRK,(10,5,1)),VOLUME=SER=RTOSB
//SYSLIB  DD    DSNAME=SYS1.FORTLIB,DISP=OLD,UNIT=DISK

//SYSPRINT DD   SYSOUT=A
//SYSOUT DD     UNIT=DISK,SPACE=(CYL,(4,1)),VOLUME=SER=RTOSA
//SYSLIN DD     UNIT=DISK,DISP=(OLD,DELETE),VOLUME=SER=RTOSA,       X
//              DSNAME=&LNKFILE,DCB=(,RECFM=F,BLKSIZE=80)
```

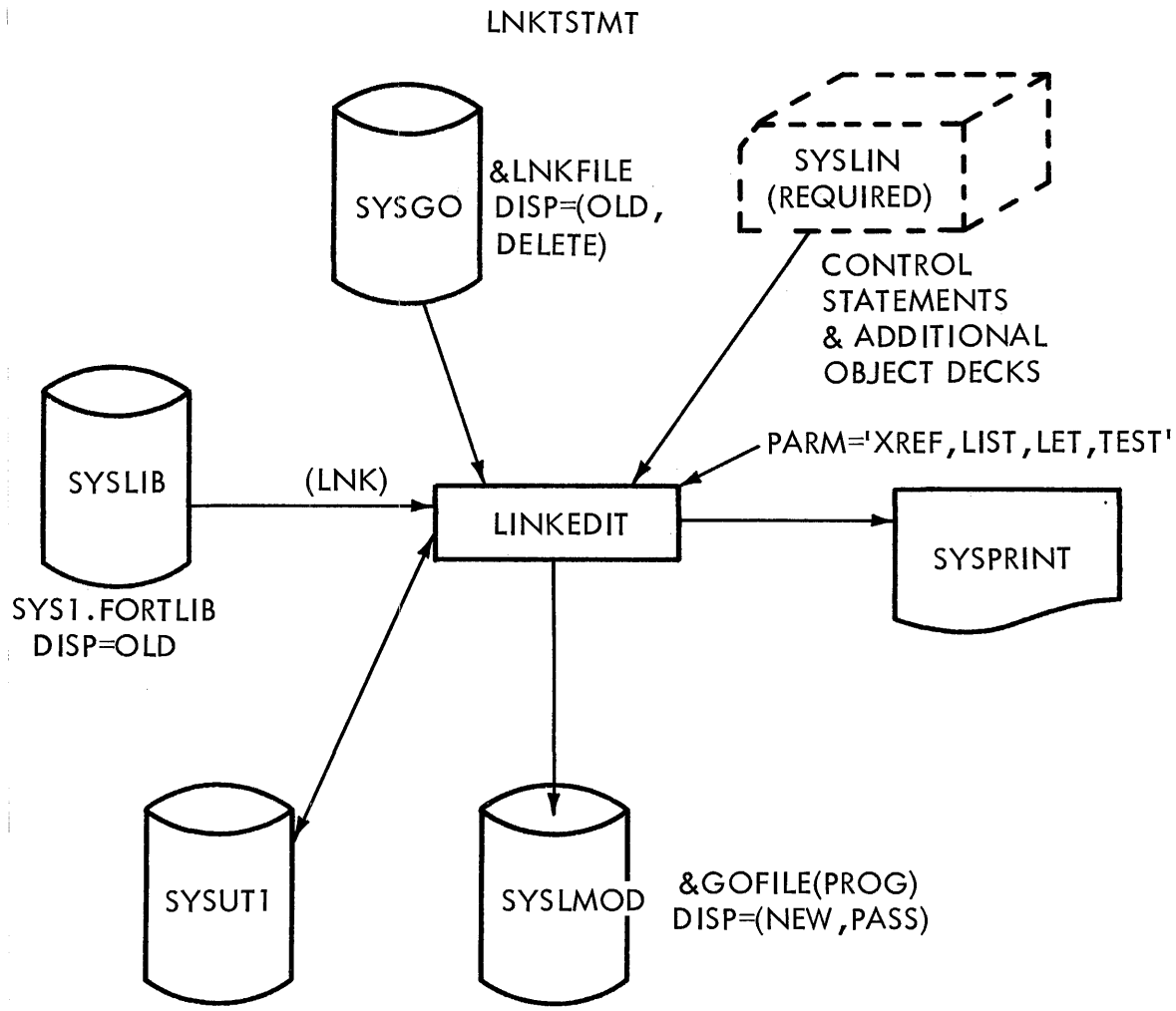
NOTE: Ref. figure on pg. 4.4.101

LINKMULT




```
//LNK   EXEC  PGM=LINKEDIT,PARM='XREF,LET,LIST,REUS'                    X
//             *                                                        X
//             *LINKEDIT STEP WITH MULTIPLE INPUTS                      X
//             *
//SYSPRINT DD  SYSOUT=A
//SYSUTI DD    UNIT=DISK,SPACE=(CYL,(4,1)),VOLUME=SER=RTOSA
//SYSLMOD DD DSNAME=&GOFILE(PROG),UNIT=DISK,DISP=(,PASS),              C
//             SPACE=(TRK,(8,1,1)),VOLUME=SER=RTOSB
//SYSLIB DD    DSNAME=SYS1.FORTLIB,DISP=OLD,                           X
//             DCB=(,RECFM=0,BLKSIZE=1024)
//SYSGO  DD    UNIT=DISK,VOLUME=SER=RTOSA,DSNAME=&LNKFILE,             X
//             DISP=(OLD,DELETE),DCB=(,RECFM=F,BLKSIZE=80)
```

NOTE: Ref. figure on pg. 4.4.101




```
//LNK     EXEC  PGM=LINKEDIT,PARM='XREF,LIST,LET,TEST'                  X
//              *
//              * LINKEDIT STEP WITH MULTIPLE INPUTS AND TESTRAN        X
//              *
//SYSLMOD DD DSNAME=&GOFILE(PROG),UNIT=DISK,DISP=(,P .SS),              C
//              SPACE=(TRK,(10,5,1)),VOLUME=SER=RTOSB
//SYSLIB DD     DSNAME=SYS1.FORTLIB,DISP=OLD

//SYSPRINT DD   SYSOUT=A
//SYSUTI DD     UNIT=DISK,SPACE=(CYL,(4,1)),VOLUME=SER=RTOSA
//SYSGO  DD     UNIT=DISK,VOLUME=SER=RTOSA,DSNAME=&LNKFILE,             X
//              DISP=(OLD,DELETE),DCB=(,RECFM=F,BLKSIZE=80)
```

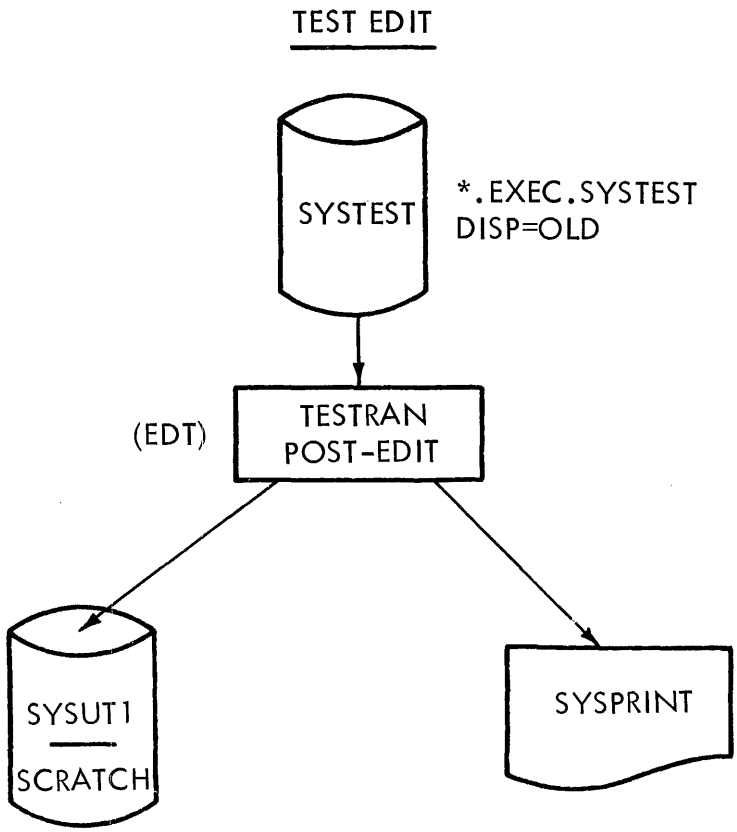
NOTE: Ref. figure on pg. 4.4.101

ADD A PROCEDURE TO SYS1.PROCLIB

```
// JOBF        JOB     1, NAME, MSGLEVEL=1
// STEP1       EXEC    PGM=IEBUPDAT
// SYSPRINT    DD      SYSOUT=A
// SYSUT2      DD      DSNAME=SYS1.PROCLIB,DISP=OLD
// SYSIN       DD      DATA

./          ADD     ASMFC,00,0,1
./          NUMBR   00000000,00000000,00000010,00000010

        { PROCEDURE  TO BE ADDED

./          ENDUP
/*
```

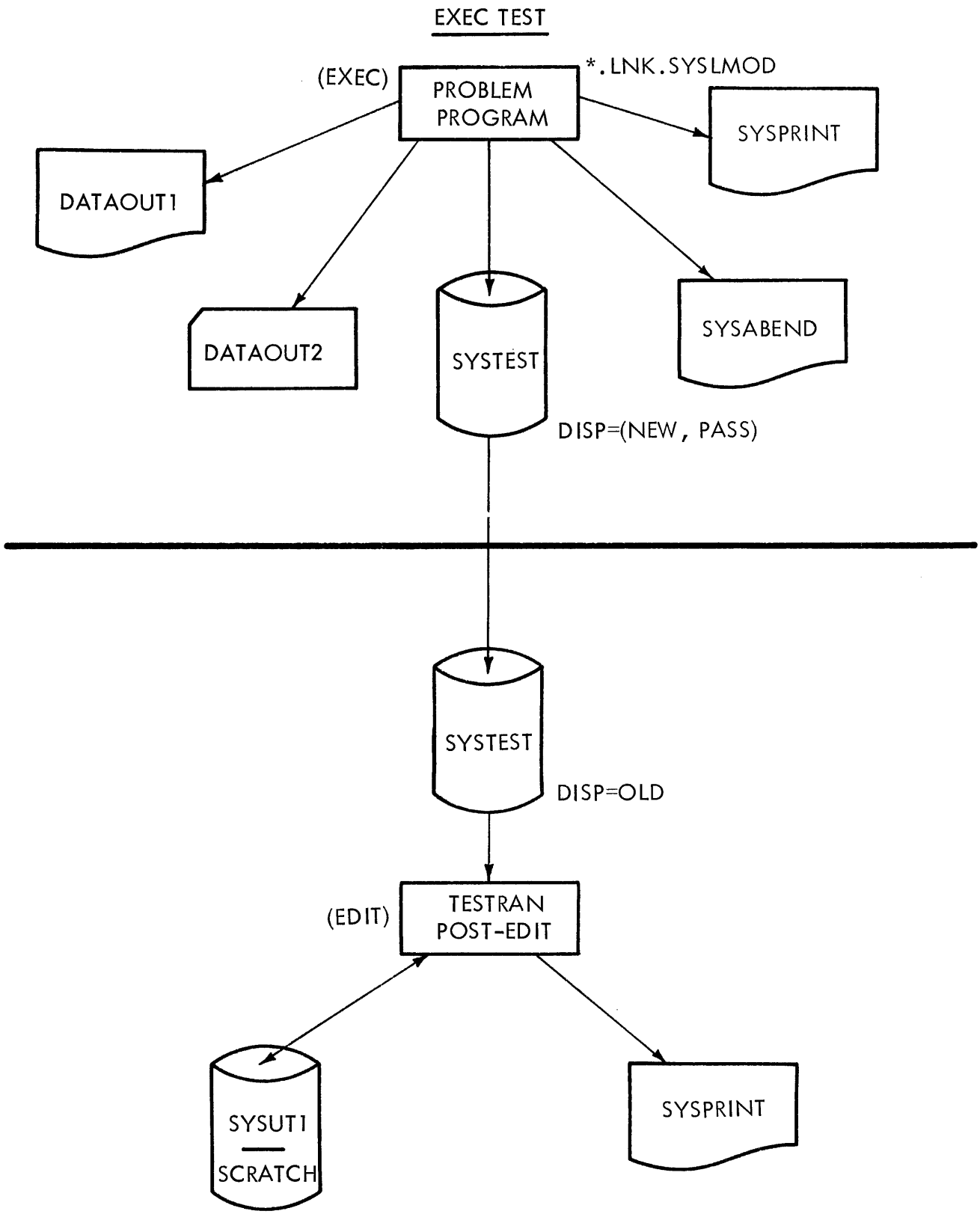
CANNOT CATALOG IN A PROCEDURE

JOB STATEMENT
DD STATEMENT WITH JOBLIB IN THE NAME FIELD
DD STATEMENT WITH * IN THE OPERAND FIELD
DD STATEMENT WITH DATA IN THE OPERAND FIELD
COMMAND, NULL, OR DELIMITER STATEMENTS




```
//EDT    EXEC    PGM=IEGTTEDT                                    X
//              *                                               X
//              *  TESTRAN POST EDITOR STEP                     X
//              *
//SYSPRINT DD     SYSOUT=A
//SYSUT1 DD       UNIT=DISK,VOLUME=SER=RTOSB,                   X
//              DISP=(NEW,DELETE),SPACE=(TRK,(100,10))
//SYSTEST DD      DSNAME=*.EXEC.SYSTEST.INPUT=DISK,             X
//              VOLUME=REF=*.EXEC.SYSTEST,DISP=OLD
```

NOTE: Ref. figure on pg. 4.4.101


EXEC TEST
(EXEC)
PROBLEM PROGRAM
*.LNK.SYSLMOD
SYSPRINT
DATAOUT1
DATAOUT2
SYSTEST
SYSABEND
DISP=(NEW, PASS)
SYSTEST
DISP=OLD
(EDIT)
TESTRAN POST-EDIT
SYSUT1
SCRATCH
SYSPRINT

```
//EXEC    EXEC  PGM=*.LNK.SYSLMOD                                          X
//              *                                                          X
//              * EXECUTION STEP WITH TESTRAN                              X
//              *
//SYSPRINT DD   SYSOUT=A
//SYSABEND DD   SYSOUT=A
//DATAOUT1  DD   SYSOUT=A
//DATAOUT2  DD   UNIT=PUNCH
//SYSTEST1 DD   UNIT=DISK,VOLUME=SEP=RTOSA,                                X
//              SPACE=(TRK,(100,10)),DISP=(NEW,PASS)
//EDT     EXEC  PGM=IEGTTEDT                                               X
//              *                                                          X
//              * TESTRAN POST EDITOR STEP                                 X
//              *
//SYSPRINT DD   SYSOUT=A
//SYSUTI DD     UNIT=DISK,SPACE=(TRK,(100,1)),VOLUME=SER=RTOSA
//SYSTEST  DD   DSNAME=*.EXEC.SYSTEST,UNIT=DISK,                           C
//              VOLUME=REF=*.EXEC.SYSTEST,DISP=OLD
```

NOTE: Ref. figure on pg. 4.4.101

PROGRAM MAKE A TEMPORARY CHANGE TO A SOURCE MODULE

ASSEMBLE/LINKEDIT AND EXECUTE | DATE

Name	Operation	Operand	Identification-Sequence
//RTUP1	JOB	MSGLEVEL=1	
//	EXEC	RTUPDATE	
//UPD.SYSUT1	DD	DSNAME=SMA225,DISP=ØLD	
//SYSIN	DD	*	
./	CHNGE	LAUNCH,01,0,1	
./	DELET	00000005,00000001	
	AR	7,6 }	0000000005
	LR	10,7 } REPLACING	0000000010
/*		STATEMENTS	
//	EXEC	RTASMB	
//	EXEC	LØADANGØ	

PROGRAM PERMANENTLY UPDATE SOURCE LIBRARY (CATALOGED)
ASSEMBLE/LINKEDIT/EXECUTE

DATE

```
//RTUP3    JØB  MSGLEVEL=1
//         EXEC RTUPDATE
//UPD.SYSUT1  DD  DSNAME=SMA205,DISP=ØLD
//UPD.SYSUT2  DD  DSNAME=SMA205,VOLUME=,                    C
//              DISP=ØLD
//SYSIN        DD *
./      CHNGE   LAUNCH,01,0,1
./      DELET   00000005,00000010
        AR      7,6                                         00000005
        LR      10,7                                        00000010
/*
//         EXEC RTASMB
//ASM.SYSIN DD   DSNAME=SMA205(LAUNCH)

//         EXEC LØADANGØ
```

PROGRAM MAKE A TEMPORARY CHANGE TO A SOURCE MODULE
COMPILE / LINKEDIT AND EXECUTE

DATE

```
//RTUP2     JØB  MSGLEVEL=1
//          EXEC RTUPDATE
//SYSUT1    DD   DSNAME=SMA225,DISP=ØLD
//SYSIN     DD   *
./        CHNGE  DRIVER,02,0,1
./        DELET  00000010,00000020
      FØRMAT (I20)                                                      00000010
    2 I=I+2                                                             00000020
/*
//          EXEC RTCOMPH
//          EXEC LØADANGØ
```

PROGRAM PERMANENTELY UPDATE

COMPILE /LINKEDIT AND EXECUTE | DATE

```
//RTUP4   JØB  MSGLEVEL=1
//        EXEC RTUPDATE
//UPD.SYSUT1  DD  DSNAME=SMA225,DISP=ØLD
//UPD.SYSUT2  DD  DSNAME=SMA225,VØLUME=2                              C
//               DISP=ØLD
//SYSIN       DD  *
./             CHNGE DRIVER,03,0,1
./             DELET 00000010,00000020
          FØRMAT (I2Ø)                                                00000010
      2   I=I+2                                                       00000020
/*
//        EXEC RTCØMPH
//FØRT.SYSIN  DD  DSNAME=SMA225(DRIVER)
//        EXEC LØADANGØ
```

PROGRAM PERMANENTY UPDATE RT-LOAD MODULE
USING AN OBJECT DECK

DATE

```
//LØAD1    JØB MSGLEVEL=1
//        EXEC RTLNKMØD
//DLNK.SYSMØDI DD DSNAME=SMA.LØAD(JAZZ),                               C
//             DISP=ØLD
//LNK.SYSLINA DD DDNAME=SYSIN2
//LNK.SYSLIN DD DSNAME=SYSIN
//          DD DSNAME=*.SYSLINB
//             VØLUME=REF=*.SYSLINB                                    C
//SYSIN       DD *
         ⎧ØBJECT   ⎫
         ⎨MØDULES  ⎬
         ⎩         ⎭
/*
//RTLK.SYSMØDØ DD DSNAME=SMA.LØAD(JAZZ)
```

PROGRAM PERMANENTLY UPDATE SØURCE AND LØAD MØDULE LIBRARIES

DATE

```
//UPDATE1   JØB MSGLEVEL=1
//         EXEC RTUPDATE
//UPD.SYSUT1 DD  DSNAME=SMA225,DISP=ØLD
//UPD.SYSUT2 DD  DSNAME=SMA225,VOLUME=,                                 C
//            DISP=ØLD
//SYSIN      DD *
./       CHNGE LAUNCH,03,0,1
./       DELET 00000005,00000010
         AR    7,6                                                      00000005
         LR    10,0(7)                                                  00000010
/*
//         EXEC RTASMB
//ASM.SYSIN    DD  DSNAME=SMA225(LAUNCH)
//         EXEC RTLNKMØD
//DLNK.SYSMØDI DD  DSNAME=SMA.LØAD(JAZZ)
//RTLK.SYSMODØ DD  DSNAME=SMA.LØAD(JAZZ)
```

```
**** JCL AND OBJECT MODULE TO LINK EDIT HPINIT
//HPINIT  JOB 1,STRAUSS,MSGLEVEL=1
//  EXEC PGM=LINKEDIT,PARM='NCAL,LIST,XREF,SCTR,LET'
//SYSPRINT DD SYSOUT=A
//SYSUT1  DD  UNIT=2311,SPACE=(CYL,(5,5))
//SYSLMOD DD DSNAME=SYS1.LINKLIB,DISP=OLD
//SYSLIN DD *
BESD        &    AHPINIT          A-                                                    0001
BTXT        8    A-(-)E-&--BA--=Q1   HC   A&/KA   AL +AQMG--YQ--BG=MO-KQTN,-B          0002
BTXT    8   8    AG&-BAW AF&-,AN A$TF&A3-CA& W$FD&-DF-D--+ L   KXOB-OQ--BQ           0003
BTXT    0   W    A )Q) QG EO-=   A&     Q(-*+GK -OO K /+OB                            0004
BTXT    Q   (    A                                                                    0005
BTXT   A&   )    A                                                                    0006
BRLD        H      A AH   '                                                          0007
BEND                                                                                 0008
  ENTRY HPINIT
  NAME HPINIT(R)

//SPRINTER  JOB  13,'PRINT SYSGEN',MSGLEVEL=1
//STEP1   EXEC  PGM=IEFPRT
//SYSPRINT   DD  SYSOUT=A
//PRINTAP   DD  DSNAME=SYSOUT,UNIT=182,LABEL=(,SL),DISP=OLD,                    XX
//              VOLUME=SER=DYSOUT
//PRINTER   DD  SYSOUT=A

**** SAMPLE JCL AND CONTROL CARDS TO SCRATCH DATA SETS
//   VARY 192,OFFLINE
//   START WTR,282
//   DISPLAY JOBNAMES
//SCRATCH  JOB 1,STRAUSS,MSGLEVEL=1
//EXSCRAT  EXEC PGM=IEHPROGM
//DD1   DD  UNIT=2311,VOLUME=SER=SPSRES,DISP=OLD
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
  SCRATCH  DSNAME=COBOLE,VOL=2311=SPSRES,PURGE
  SCRATCH  DSNAME=FORTRANE,VOL=2311=SPSRES,PURGE
  SCRATCH  DSNAME=SYS1.PROCLIB,MEMBER=CTLCDS,VOL=2311=SPSRES,PURGE
   SCRATCH  VTOC,SYS,VOL=2311=SPSRES,PURGE
/*

**** SAMPLE JCL AND CONTROL CARDS TO ADD A MEMBER TO PROCLIB
//IEBUPDAT   JOB 1,STRAUSS,MSGLEVEL=1
//UPDATE  EXEC  PGM=IEBUPDAT,PARM=NEW
//SYSPRINT DD  SYSOUT=A
//SYSUT2 DD    DSNAME=SYS1.PROCLIB,DISP=OLD
//SYSIN  DD    DATA
./       ADD   CTLCDS,01,0,1
 PUNCH TYPORG=PS,MAXFLDS=1,STOPAFT=100
    RECORD  FIELD=(80,1,,1)
./       ENDUP
/*

**** JCL TO EXECUTE HIGH COMPUTE JOB
//HIGHCOMP  JOB  1,STRAUSS,MSGLEVEL=1
//JOBLIB  DD  DSNAME=FORTRANE,DISP=(OLD,PASS),VOLUME=REF=SYS1.SVCLIB
//EXHICOMP  EXEC  PGM=HIGHCOMP
//FT06F001 DD SYSOUT=A,DCB=(RECFM=UA,BLKSIZE=136)

**** JCL TO EXECUTE HIGH IO JOB
```

```
//HIGHIO  JOB 1,STRAUSS,MSGLEVEL=1
//JOBLIB  DD  DSNAME=COBOLE,DISP=(OLD,PASS),VOLUME=REF=SYS1.SVCLIB
//EXHIIO  EXEC  PGM=HIGHIO
//WRRDONE  DD UNIT=191,SPACE=(CYL,(25,10))
//WRRDTWO  DD UNIT=191,SPACE=(CYL,(25,10))
//WRITEONE DD   UNIT=(180,,DEFER),LABEL=(,NL)
//WRITETWO DD   UNIT=(181,,DEFER),LABEL=(,NL)

**** SAMPLE JCL AND CONTROL CARDS TO PUNCH TAPE
//TPTOPCH  JOB 1,STRAUSS,MSGLEVEL=1
//PUNCH  EXEC  PGM=IEBPTPCH
//SYSPRINT  DD  SYSOUT=A
//SYSUT1  DD  UNIT=(182,,DEFER),LABEL=(,NL),VOLUME=SER=123123,          C
//              DCB=(RECFM=F,BLKSIZE=80)
//SYSUT2  DD  UNIT=00D,DCB=(RECFM=F,BLKSIZE=80)
//SYSIN  DD  DSNAME=SYS1.PROCLIB(CTLCDS),DISP=OLD

**** SAMPLE JCL AND CONTROL CARDS TO PRINT TAPE
//TPTOPRT  JOB 1,STRAUSS,MSGLEVEL=1
//PRINT  EXEC  PGM=IEBPTPCH
//SYSPRINT  DD  SYSOUT=A
//SYSUT1  DD  UNIT=(183,,DEFER),LABEL=(,NL),VOLUME=SER=ABCABC,          C
//              DCB=(RECFM=F,BLKSIZE=80)
//SYSUT2  DD  UNIT=00E,DCB=(RECFM=F,BLKSIZE=80)
//SYSIN  DD  DSNAME=SYS1.PROCLIB(CTLCDX),DISP=OLD

**** SAMPLE JCL AND DATA TO GO CARD TO TAPE
//IBGENER  JOB 1,STRAUSS,MSGLEVEL=1
//CDTOTAPE  EXEC  PGM=IEBGENER
//SYSIN  DD  DUMMY
//SYSUT2  DD  UNIT=(2400,,DEFER),LABEL=(,NL),DCB=(RECFM=F,BLKSIZE=80)
//SYSPRINT  DD  SYSOUT=A
//SYSUT1  DD  *
DATA CARDS PLACED HERE
/*

**** SAMPLE JCL AND CONTROL CARDS TO LIST VTOC
//IEHLIST  JOB 1,STRAUSS,MSGLEVEL=1
//LIST  EXEC  PGM=IEHLIST
//SYSPRINT DD  SYSOUT=A
//DD1  DD  UNIT=2311,VOLUME=SER=SPSRES,DISP=OLD
//SYSIN  DD     *
         LISTVTOC DUMP,VOL=2311=SPSRES
    LISTCTLG
  LISTPDS  DSNAME=SYS1.LINKLIB,VOL=2311=SPSRES
/*

**** JCL AND OBJECT MODULE TO LINK EDIT HIGH COMPUTE JOB
//FORDEMO  JOB 1,STRAUSS,MSGLEVEL=1
//LKED EXEC PGM=LINKEDIT,PARM='XREF,LIST'
//SYSLMOD  DD  DSNAME=FORTRANE(HIGHCOMP),DISP=(,KEEP),                  C
//              SPACE=(CYL,(5,5,1)),VOLUME=REF=SYS1.SVCLIB
//SYSUT1  DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSLIB DD DSNAME=SYS1.FORTLIB,DISP=OLD
//SYSPRINT DD SYSOUT=A
//SYSLIN  DD  *
BESD        &    AMAIN=                                                MAIN0001
BESD        &     MAIN    A  H   A                                     MAIN0002
BTXT  F(    D    A                                                     MAIN0003
```

```
BESD        &   BINVERT  B                                                    MAIN0004
BRLD        H     B A* F(                                                    MAIN0005
BTXT  G(    U   A   M       A   +   I   BA&  A-  A&                          MAIN0006
BTXT  G&    8   ABF++HFS B-(1INPUT MATRIXS   B-DOROW&CS  B-D COL&C-C TO&CMAIN0007
BTXT  GH    8   AF+++FS  B-&1INVERTED MATRIXSB-L1RE-INVERTED MATRIXS B-AOMAIN0008
BTXT  G&    0   A -/0***** INVERSIONS COMPLETED ***** -AO -AOS             MAIN0009
BTXT  H*    D   A                                                          MAIN0010
BESD        &   CIBCOM=  B                                                 MAIN0011
BTXT  H)    &   A  F-&  -  F-&  -                                          MAIN0012
BRLD        Y     C A* H* A A) H) A A) H& A A) HM A A) HQ                   MAIN0013
BTXT  H-    D   A   &                                                      MAIN0014
BTXT  HU    6   AQOH*E O Q(A- Q&-&H&&-DQ G(& F-A  A& FYA  A& FMA  A& FQ   MAIN0015
BTXT  H-    8   AQOFM$OFQQ&F8GAQ&F'G/8 GDQ-FM.- BQOH-)OF--TOF 'Q&GDG18 GHMAIN0016
BTXT  HK    6   AQ-FM.- BQOH-)OFO-TOF 'Q&GDG18 G)Q-FM.- BQOH-)OF--TOF '  MAIN0017
BTXT  HH    8   AA- AQOF-Q FQQ&G8GB& A- AQOF-Q FMQ&G4GB& Q&G-G1G QOH*E O MAIN0018
BTXT  I     8   A   E  G&A  A& FMA  A& FQQ-FM.- BQ H-) F---G E OHDF 'A- AMAIN0019
BTXT  I8    8   AQOF-Q FQQ&H GB& A- AQOF-Q FMQ&G'GB& E O&QOH*E OD   F  GQMAIN0020
BTXT  IO    8   AE O&Q GO& FUA  A& FMQOH*E OD   F  G(E OHD FME O&Q G4& F*MAIN0021
BTXT  IY    8   AQ G8& F QOF $OF-Q&G GJQ F-& F G QOH*E OD   F  G8E OHD F*MAIN0022
BTXT  I     8   AE OHD F Q F*& FQQ-FM.- BQ H-) F---G E OHDF 'A- AQOF Q FQMAIN0023
BTXT  +Q    8   AQ&HHGB& E O&QOF -OG4&OF*QOF*-OG'&OF QOF*$OF-Q&GUGAQ&GYGAMAIN0024
BTXT  +&    8   AA- AQOF-Q FMQ&HDGB& QOFU-OG4&OFUQOFU$OG Q&G&GAQ&GQG/QOH*MAIN0025
BTXT  +H    8   AE OD   F  G-E O&A&H)QOF(E G  TQ&GMG1QOH*E OD   F  GUE O&MAIN0026
BTXT  +&    U   AA&HMQOF(E G  XQ&GMG1QOH*E OD   F  G'                      MAIN0027
BRLD        8     A A) ID A A) I( A A) I- A A) IM A A) +- A A) +8 A A) +  MAIN0028
BTXT  +U    W   AE O&A- AAO EQ FYQ&GOGB& QOH*E 04E00000                    MAIN0029
BTXT        M   AMAIN   H-(-)QGOQGOHU                                      MAIN0030
BTXT  FO    8   A  HO  +Y  H+  H,  IF  HH  +&  +D  +(  I'  +M  H,  I-  IOMAIN0031
BTXT  GY    D   A  I=                                                      MAIN0032
BTXT  GO    *   A  HF  H+  HO  I)  IM  I&  I(                             MAIN0033
BRLD        8     A A( FO( F4( F8( F'( G ( GD( GH( G)( G&( GM( GQ( G*) G-MAIN0034
BRLD        )     A A( GU) GY                                              MAIN0035
BRLD        -     A A( GO( G4( G8( G'( H ( HD) HH                          MAIN0036
BTXT   -    D   A                                                          MAIN0037
BRLD        H     A A)  -                                                  MAIN0038
BEND   H        A                .+                                        MAIN0039
BESD        &   AINVERT=                                                   INVE0001
BESD        &    INVERT  A  H   A                                          INVE0002
BTXT  A*    D   A                                                          INVE0003
BESD        &   BCLOCK   B                                                 INVE0004
BRLD        H     B A* A*                                                  INVE0005
BTXT  AQ    &   A   AA&  A&                                                INVE0006
BTXT  AY    8   AB-*OSINGULARITY IN INPUT MATRIX -N SUBMATRIX ALL ZEROES  INVE0007
BTXT  B-    8   A-D I =&C-E  J =&CS  B-/ TOTAL MATRIX CALCULATION TIME =   INVE0008
BTXT  BQ    &   AHB+HB-H SECONDSS                                          INVE0009
BTXT  B*    )   A   (   &                                                  INVE0010
BESD        &   CIBCOM=  B                                                 INVE0011
BTXT  BM    H   A& AM& A&                                                  INVE0012
BRLD        Q     A A) BM C A* BU A A) BQ                                  INVE0013
BTXT  BY    &   A+      +                                                  INVE0014
BTXT  B8    8   A&&O&Q(A- Q&-&H&&-DQ-& KC -- KC M&DA&BMQOA*E G  A8 A*0 A8INVE0015
BTXT  BO    6   AA  A& A'Q A'Q&A'.& B&E MQ A'&E UQO MQ-A'.- BQOB )OA=-T   INVE0016
BTXT  CW    8   A--B*8WO OWO-AHQ A'& A Q A'& ADQ& MQOA .O BQOB )OAF-3-OB*INVE0017
BTXT  C     6   A8X& OW8 AH=FQ&A(G1Q- MQ&A .& BQOB )OAF-L-&B*8V- OWO-AH   INVE0018
BTXT  CM    8   AQ A QOA'.O B&G MQ AD&G UA- AQO -Q ADQ&BOGB& A- AQO -Q A INVE0019
BTXT  C)    8   AQ&8(GB& 8 AH2 Q&AUG/QOBUE OD   F  AYE OHD A'E OHD A'E O&INVE0020
BTXT  DD    6   AQ&A'.& BQE M& ADQOAD$OA'Q&AYGJA  A& A Q- MQ&A'.& BQOB   INVE0021
BTXT  DO    6   A)OAB-L-&B*8V- 3WO-A)QOAD.O BQOB )OAB-3-OB*8G- OE- 8 A)   INVE0022
BTXT  DO    6   AOG- A- AQO -Q A Q&B4GB& Q&A'.& BQE U& A QOA $OA'Q&AOGJ   INVE0023
```

```
BTXT   DW    8    AA  A& ADQ- MQ&AD.& BQOB )OA=-L-&B*8V- 3WO-A)QOAD.O BQOB INVE0024
BTXT   D     6    A)OAB-3-OB*8G- OE- 8 A)OG- A- AQO -Q ADQ&B8GB& A  A& A   INVE0025
BTXT   EM    6    AQOA $OA'Q&A4GAQ- MQ&A'.& BQOB )OA=-L-&B*8V- 3WQOA .O B  INVE0026
BTXT   E+    8    AQOB )OA=-3-OB*8G- 'FOG- A- AQO -Q A Q&B'GB& A  A& A A  AINVE0027
BTXT   EB    6    A& ADQOA $OA'Q&A8GAQOAD$OA'Q&A8GAQ- MQ&A .& BQOB )OA=-L  INVE0028
BTXT   E8    8    A-&B*8V- QOA'.O BQOB )OAF-3-OB*'X- Q&A .& BQOB )OAF-C-&B*INVE0029
BTXT   EO    8    AOY- OY- A- AQO -Q ADQ&BDGB& A- AQO -Q A Q&B&GB& A  A& ADINVE0030
BTXT   FY    8    AQOAD$OA'Q&A'GAQ- MQ&A'.& BQOB )OAF-L-&B*8V- QOA'.O BQOB INVE0031
BTXT   F-    6    A)OA=-3-OB*'X- OV- A- AQO -Q ADQ&BHGB& 8-A8Q- MQ&A'.& B  INVE0032
BTXT   FO    6    AQOB )OA=-L-&B*'V- O-A88-A8'V- O-A8,W8V- Y A  FOE- A- A  INVE0033
BTXT   F)    8    AQO -Q A'Q&BYGB& Q  -& A'QOA'$OAQ&OA'Q A'K Q&A-GJQ&A'.& BINVE0034
BTXT   GD    8    AQE M& A QOA $OA'Q&AMGJA  A& ADQ- MQ&AD.& BQOB )OA=-L-&B*INVE0035
BTXT   G'    8    A8E- O A)QOAD.O BQOB )OAB-3-OB*8X- 3WOV- 8 A)OG- A- AQO -INVE0036
BTXT   G4    6    AQ ADQ&B)GB& Q&A'.& BQE U& ADQOAD$OA'Q&A)GJA  A& A Q- M  INVE0037
BTXT   G&    8    AQ&A'.& BQOB )OAB-L-&B*8E- O A)QOAD.O BQOB )OAB-3-OB*8X- INVE0038
BTXT   GS    6    A3WOV- 8 A)OG- A- AQO -Q A Q&B-GB& Q&A)G1A&BQQOA*E G     INVE0039
BTXT   HQ    8    AQOA&$OAM&T&-B4Y-BY&-BOK3EOG OF1WO-AQ8 AQ2 Q&AMGAQOBUE ODINVE0040
BTXT   H&    4    A   F  B4E OHD AQE O&Q& QQ-& KC-  -Q- *Q(-*Q -)K -)G=     INVE0041
BTXT         M    AINVERT H-(-)QGOQGOB8                                     INVE0042
BRLD         &      A A) C( A A) HM                                         INVE0043
BTXT   A-    8    A  B-  HU  FH  C(  CO  C   D(  DO  ES  ES  F6  FU  H+  G&INVE0044
BTXT   AQ    8    A  GU  GY  F'  DD  DH  DS  E)  ES  E8  F2  G-  G   DW  EMINVE0045
BTXT   A-    H    A  HH  HU                                                 INVE0046
BTXT   BY    (    A  B4  C8  C   DW  D&  E&  EO  EB  FU  G   GS             INVE0047
BRLD         8      A A( A-( AU( AY( A(( AO( A4( A8( A'( A&( AD( AH( A)) A-INVE0048
BRLD         8      A A( AM( AQ( A*( A ( AU( AY( A(( AO( A4( A8( A'( A&) ADINVE0049
BRLD         M      A A( AH( A)( A-) AM                                     INVE0050
BRLD         O      A A( BY( B(( BO( B4( B8( B'( B&( BD( BH( B)) B-         INVE0051
BTXT   -     D    A                                                         INVE0052
BRLD         H      A A)  -                                                 INVE0053
BEND                             HD                                        INVE0054
BESD         -    ACLOCKSUB        &CLOCK   A       A                        0001
BTXT         8    AGOO+ECLOCK-(-)A 08&  H&  DQ A& A+.Q) DQO-QQOO & O Q(-)G=     0002
BEND                                                                        0003
 NAME HIGHCOMP(R)
/*
```

```
**** JCL AND SOURCE TO COMPILE HIGH COMPUTE JOB
//FORTSOUR  JOB  1,STRAUSS,MSGLEVEL=1
//FORT EXEC PGM=IEJFAAAO,PARM='SIZE=43COO'
//SYSPRINT DD SYSOUT=A
//SYSLIN  DD  UNIT=2540-2
//SYSUT1  DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSUT2  DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSIN DD *
      DIMENSION A(20,20)
C                                                                     INMTRX
C     DEVELOP INPUT MATRIX                                            INMTRX
C     UPPER TRIANGLE=8, LOWER=1, MAIN DIAGONAL=6                      INMTRX
C                                                                     INMTRX
      N = 20
C   CHANGE THE 5 IN THIS DO TO CONTROL THE NUMBER OF REPITITIONS * * * * * *
      DO 30 LIMIT=1,5
8     DO 12 I=1,N                                                     INMTRX
      DO 12 J=1,N                                                     INMTRX
      IF(I-J)9,10,11                                                  INMTRX
9     A(I,J)=8.                                                       INMTRX
      GO TO 12                                                        INMTRX
10    A(I,I)=6.                                                       INMTRX
```

```
      GO TO 12                                                          INMTRX
11    A(I,J)=1.                                                         INMTRX
12    CONTINUE                                                          INMTRX
      GO TO 29
C                                                                       INMTRX
C     READ INPUT MATRIX                                                 INMTRX
C                                                                       INMTRX
13    READ(5,14)((A(I,J),J=1,N),I=1,N)                                  INMTRX
14    FORMAT(10F8.6)                                                    INMTRX
C                                                                       INMTRX
C     INVERT THE MATRIX TWICE.  PRINT THE INPUT MATRIX AND BOTH         INMTRX
C       INVERTED MATRICES.                                              INMTRX
C                                                                       INMTRX
29    WRITE(6,15)                                                       INMTRX
15    FORMAT('1INPUT MATRIX')                                           INMTRX
      KNT=0                                                             INMTRX
16    DO 22 I=1,N                                                       INMTRX
      WRITE(6,17)I                                                      INMTRX
17    FORMAT('0ROW',I3)                                                 INMTRX
      J1=1                                                              INMTRX
      JN=10                                                             INMTRX
18    IF(JN-N)20,20,19                                                  INMTRX
19    JN=N                                                              INMTRX
20    WRITE(6,21)J1,JN,(A(I,J),J=J1,JN)                                 INMTRX
21    FORMAT(' COL',I3,' TO',I3,10F10.6)                                INMTRX
      J1=JN+1                                                           INMTRX
      JN=J1+9
      IF(J1-N)18,19,22                                                  INMTRX
22    CONTINUE                                                          INMTRX
      KNT=KNT+1                                                         INMTRX
      IF (KNT-2) 23,25,27
23    WRITE(6,24)                                                       INMTRX
24    FORMAT('1INVERTED MATRIX')                                        INMTRX
      CALL INVERT(N,A)                                                  INMTRX
      GO TO 16                                                          INMTRX
25    WRITE(6,26)                                                       INMTRX
26    FORMAT('1RE-INVERTED MATRIX')                                     INMTRX
      CALL INVERT(N,A)                                                  INMTRX
      GO TO 16                                                          INMTRX
27    WRITE(6,28)                                                       INMTRX
28    FORMAT('0'/'0***** INVERSIONS COMPLETED *****'/'0'/'0')           INMTRX
30    CONTINUE
      STOP
      END                                                               INMTRX
      SUBROUTINE INVERT(N,A)
      DIMENSION A(20,20),L(20),M(20)
      CALL CLOCK(ITIMES)
      D=1.0
      DO 80 K=1,N
      L(K)=K
      M(K)=K
      BIGA=ABS(A(K,K))
      DO 20 I=K,N
      DO 20 J=K,N
      IF(BIGA-ABS(A(I,J)))10,20,20
   10 BIGA=ABS(A(I,J))
      L(K)=I
      M(K)=J
   20 CONTINUE
```

```
      IF (BIGA)21,21,24
   21 WRITE(6,22)K,K
   22 FORMAT('0SINGULARITY IN INPUT MATRIX'/' SUBMATRIX ALL ZEROES'/
     $       ' I =',I3,'  J =',I3)
C
C       ROW INTERCHANGE
C
   24 J=L(K)
      IF(J-K)35,35,25
   25 DO 30 I=1,N
      HOLD=-A(K,I)
      A(K,I)=A(J,I)
   30 A(J,I)=HOLD
C
C       INTERCHANGE COLUMNS
C
   35 I=M(K)
      IF(I-K)45,45,37
   37 DO 40 J=1,N
      HOLD= -A(J,K)
      A(J,K)=A(J,I)
   40 A(J,I)=HOLD
C
C       DIVIDE COLUMN BY MINUS PIVOT
C
   45 DO 55 I=1,N
   46 IF(I-K) 50, 55, 50
   50 A(I,K)=A(I,K)/(-A(K,K))
   55 CONTINUE
C
C       REDUCE MATRIX
C
      DO 65 I=1,N
      DO 65 J=1,N
   56 IF(I-K) 57, 65, 57
   57 IF(J-K) 60, 65, 60
   60 A(I,J)=A(I,K)*A(K,J)&A(I,J)
   65 CONTINUE
C
C       DIVIDE ROW BY PIVOT ELEMENT
C
      DO 75 J=1,N
   68 IF(J-K) 70, 75, 70
   70 A(K,J)=A(K,J)/A(K,K)
   75 CONTINUE
C
C       CONTINUED PRODUCT OF PIVOTS
C
      D=D*A(K,K)
C
C       REPLACE PIVOT BY RECIPROCAL
C
      A(K,K)=1.0D0/A(K,K)
   80 CONTINUE
C
C       FINAL ROW AND COLUMN INTERCHANGE
C
      K=N
  100 K=K-1
```

```
      IF(K) 150, 150, 103
  103 I=L(K)
      IF(I-K) 120, 120, 105
  105 DO 110 J=1,N
      HOLD=A(J,K)
      A(J,K)=-A(J,I)
  110 A(J,I)=HOLD
  120 J=M(K)
      IF(J-K) 100, 100, 125
  125 DO 130 I=1,N
      HOLD=A(K,I)
      A(K,I)= -A(J,I)
  130 A(J,I)=HOLD
      GO TO 100
  150 CALL CLOCK (ITIME)
      ZDELTA = ITIME - ITIMES
      IF (ZDELTA) 77,88,77
   77 WRITE (6,999) ZDELTA
  999 FORMAT (' TOTAL MATRIX CALCULATION TIME = ',-2PF8.2,' SECONDS')
   88 RETURN
      END
/*
//ASSEM   EXEC  PGM=IEUASM
//SYSLIB    DD    DSNAME=SYS1.MACLIB,DISP=OLD
//SYSPRINT  DD    SYSOUT=A
//SYSPUNCH   DD  UNIT=2540-2
//SYSUT1    DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSUT2    DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSUT3    DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSIN     DD    *
CLOCKSUB START 0
         ENTRY CLOCK
         USING *,15
CLOCK    SAVE  (14,12),,*
         LA    14,SAVEAREA          LOAD NEW SAVE AREA ADDRESS
         ST    14,8(13)         NEW SAVE AREA ADDR TO OLD SAVE AREA
         ST    13,4(14)         OLD SAVE AREA ADDR TO NEW SAVE AREA
         LR    13,14            NEW SAVE AREA ADDR TO REGISTER 13
         TIME  BIN
         L     13,4(13)         RESTORE REGISTER 13
         L     3,24(0,13) PARAM LIST ADR IN R3
         L     3,0(0,3)
         ST    0,0(0,3)
         RETURN (14,12)
SAVEAREA DS    18F
         END
/*

**** JCL AND OBJECT MODULE TO LINK EDIT HIGH IO JOB
//COBDEMO  JOB  1,STRAUSS,MSGLEVEL=1
//LKED     EXEC PGM=LINKEDIT,PARM='XREF,LIST'
//SYSLMOD  DD  DSNAME=COBOLE(HIGHIO),DISP=(,KEEP),SPACE=(CYL,(5,5,1)), C
//               VOLUME=REF=SYS1.SVCLIB
//SYSUT1   DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSLIB    DD   DSNAME=SYS1.COBLIB,DISP=OLD
//SYSPRINT  DD   SYSOUT=A
//SYSLIN    DD   *
BESD       &    ARDWRDEMO                                          RDWR0001
BTXT    $  D    A AT                                               RDWR0002
```

```
BTXT    U   A   A                                                                  RDWR0003
BTXT    Y   8   AXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXRDWR0004
BTXT    -   Q   AXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXRDWR0005
BTXT    8   D   A0000                                                              RDWR0006
BTXT    A   )   A ---,--/,---                                                      RDWR0007
BTXT    -   A   A                                                                  RDWR0008
BTXT    )   A   A                                                                  RDWR0009
BTXT    (   F   A/--.--                                                            RDWR0010
BTXT    T   A   A                                                                  RDWR0011
BTXT    M   8   A                       A       AB   A&   WRRDONE B HH     ARDWR0012
BTXT    A)  Y   A   A   &       A   A   A   &   A         A                 RDWR0013
BTXT    A4  8   A                       A       AB   A&   WRRDTWO B HH     ARDWR0014
BTXT    A(  Y   A   A   &       A   A   A   &   A         A                 RDWR0015
BTXT    AM  8   A                       A       AB   A&   WRITEONEB HH     ARDWR0016
BTXT    A)  Y   A   A   &       A   A   A   &   A         A                 RDWR0017
BTXT    A4  8   A                       A       AB   A&   WRITETWOB HH     ARDWR0018
BTXT    B(  Y   A   A   &       A   A   A   &   A         A                 RDWR0019
BTXT    BQ  8   ABR F B  ELAPSED TIME WRITE PHASE TOTAL LOGICAL RECORDS WRDWR0020
BTXT    B-  8   ARITTEN - ELAPSED TIME READ-WRITE PHASE TOTAL LOGICAL RECRDWR0021
BTXT    BH      AORDS READ AND REWRITTEN -END OF DEMONSTRATION              RDWR0022
BESD        &   BIHD02301B                                                  RDWR0023
BESD        &   CIHD02302B                                                  RDWR0024
BESD        &   DIHD02304B                                                  RDWR0025
BESD        &   EIHD03001B                                                  RDWR0026
BTXT    CD  8   A           0& A    +    U   CY   X& AF    B   Q0&E5/ =-+   RDWR0027
BTXT    C'  D   A  C4                                                       RDWR0028
BTXT    E*  H   A                                                           RDWR0029
BTXT    EU  H   A                                                           RDWR0030
BTXT    E(  H   A                                                           RDWR0031
BTXT    E4  H   A                                                           RDWR0032
BTXT    D'  D   A  B8                                                       RDWR0033
BTXT    D   D   A                                                           RDWR0034
BTXT    DD  D   A                                                           RDWR0035
BTXT    DH  D   A                                                           RDWR0036
BTXT    D)  D   A                                                           RDWR0037
BTXT    D&  D   A                                                           RDWR0038
BTXT    DM  D   A  &                                                        RDWR0039
BTXT    DQ  D   A  -                                                        RDWR0040
BTXT    D)  D   A  0                                                        RDWR0041
BTXT    D-  D   A                                                           RDWR0042
BTXT    DM  D   A  &                                                        RDWR0043
BTXT    DQ  D   A  -                                                        RDWR0044
BTXT    D*  D   A  0                                                        RDWR0045
BTXT    D   D   A  &                                                        RDWR0046
BTXT    DU  D   A  -                                                        RDWR0047
BTXT    DY  D   A                                                           RDWR0048
BTXT    D(  D   A  0                                                        RDWR0049
BTXT    D0  D   A  &                                                        RDWR0050
BTXT    D4  D   A  -                                                        RDWR0051
BTXT    D8  D   A                                                           RDWR0052
BTXT    D'  D   A  0                                                        RDWR0053
BTXT    D&  D   A A                                                         RDWR0054
BTXT    DD  D   A A&                                                        RDWR0055
BTXT    DH  D   A A-                                                        RDWR0056
BTXT    D)  D   A A0                                                        RDWR0057
BTXT    D*  D   A  EY                                                       RDWR0058
BTXT    E4  8   A-(-)Q   Q&   D&   HQ G   *G   C4Q' HQ/AL )++&L HQKQ02YE A& MRDWR0059
BTXT    E(  8   A&&0&K 0&A&0&+LA& MQ0&0E QJA&A4&&0&K 0&A&0&+LA&A4Q0&0E Q/RDWR0060
BTXT    DM  D   A  FF                                                       RDWR0061
```

```
BTXT  D     D   A   FQ                                                         RDWR0062
BTXT  EU    8   APC & &K) DQ02UE Q 1*Q 1QQ 1MQ01MG=Q02(E 3D 8 -Q02YE K & RDWR0063
BTXT  F*    8   A YK -  YA& MQ0&0E QJA&A4Q0&0E Q/0  &B KD0= &8B0MBQ9D0=0MRDWR0064
BTXT  DQ    D   A   F-                                                         RDWR0065
BTXT  DU    D   A   F(                                                         RDWR0066
BTXT  D*    D   A   FD                                                         RDWR0067
BTXT  FM    8   AQ&1 G*Q01(G=Q&1YG'Q02YE Q02UE Q 1UQ 1QQ 1MQ01MG=Q02(E A&RDWR0068
BTXT  D     D   A   F(                                                         RDWR0069
BTXT  F)    8   A M&&0&K 0&J &,Q&1YG)Q-&).-&K&-&)A&A4&&0MK&0MJ &,Q01(G.Q-RDWR0070
BTXT  DU    D   A   F+                                                         RDWR0071
BTXT  FD    8   A&).-&K&-&)A&0&+MA& MQ020E A&A4Q020E Q02UE Q 10Q 1MQ 1-Q&RDWR0072
BTXT  DY    D   A   G                                                          RDWR0073
BTXT  D(    D   A   G&                                                         RDWR0074
BTXT  F'    8   A1-G'Q02(E KH Y 83D 8 -Q02UE Q 14Q 1MQ 1-Q01-G=Q02(E Q02URDWR0075
BTXT  D0    D   A   GH                                                         RDWR0076
BTXT  D4    D   A   GW                                                         RDWR0077
BTXT  G4    8   AE Q 18Q 1-Q 1)Q&1)G'Q02(E Q02UE Q 1'Q 1MQ 1-Q01-G=Q02(E RDWR0078
BTXT  D'    D   A   G4                                                         RDWR0079
BTXT  E     D   A   G8                                                         RDWR0080
BTXT  G(    8   AQ&2 G'    )  0C) )0E)     0E) -  0 UAWAITING REPLY0IUIEP00RDWR0081
BTXT  D8    D   A   G&                                                         RDWR0082
BTXT  GU    8   A0D 0IUIEP990D E &HIHD02901G0Q&2HA-&H&-0 KQ& B-A&&RK.&  QRDWR0083
BTXT  G*    8   AA&&)Q1Q&2D$1 0& +TKD0& &K.08 A .080&E-N 080 Q02)G=K 0 0 RDWR0084
BTXT  ED    D   A   HM                                                         RDWR0085
BTXT  HM    8   AD-K+ H09Q&2HA-&H&-0 K & B9A&& K+&  HA&&.Q1Q&2D$1 0& +TQ0RDWR0086
BTXT  DY    D   A   H&                                                         RDWR0087
BTXT  H)    8   A2YE A& M&&0&K&0&A&0&+LA&A4&&0&K&0&A&0&+LA&AM&&0&K 0&A&0&RDWR0088
BTXT  HD    8   A+LA&AMQ0&0E Q1A&A4&&0&K 0&A&0&+LA&A4Q0&0E QAPC & &K) DQ0RDWR0089
BTXT  EH    D   A   HK                                                         RDWR0090
BTXT  D(    D   A   HU                                                         RDWR0091
BTXT  D&    D   AQ0 I8                                                         RDWR0092
BTXT  D0    D   A   H0                                                         RDWR0093
BTXT  H'    8   A2UE Q 2&Q 1QQ 1MQ&1MG'Q02(E 3D 8 -Q02YE D 1HG=Q02YE A& MRDWR0094
BTXT  E)    D   A   I&                                                         RDWR0095
BTXT  E&    D   A   IW                                                         RDWR0096
BTXT  H4    8   AQ-2M&-0&KB&/0JQ0&0E QJQ02QG=K11KK'1LK11+KH1.Q&1'G'K 0 & RDWR0097
BTXT  D4    D   A   ID                                                         RDWR0098
BTXT  DD    D   AQ0 I&                                                         RDWR0099
BTXT  D8    D   A   I&                                                         RDWR0100
BTXT  I(    8   AA&AMQ0&0E Q10  &BQQ02YE D 1)G=Q02YE A&A4Q-2*&-0&KB&/0JQ0RDWR0101
BTXT  EM    D   A   I0                                                         RDWR0102
BTXT  EQ    D   A   I=                                                         RDWR0103
BTXT  D'    D   A   I*                                                         RDWR0104
BTXT  IU    8   A&0E Q/Q02-G=K11+K41 Q&14G'K & - A&A4Q0&0E QA0  &BQQ02YE RDWR0105
BTXT  DH    D   AQ0 I8                                                         RDWR0106
BTXT  D&    D   A   IY                                                         RDWR0107
BTXT  E*    D   A   I&                                                         RDWR0108
BTXT  I*    8   AD 1-G=Q02YE Q02UE Q 2UQ 1QQ 1MQ01MG=Q02(E A& M&&0&K 0&J RDWR0109
BTXT  E-    D   A   IY                                                         RDWR0110
BTXT  EU    D   A   ++                                                         RDWR0111
BTXT  IM    8   A&,Q&2YG)Q-&).-&K&-&)A&A4&&0MK 0MJ &,Q02(G.Q-&).-&K&-&)A&RDWR0112
BTXT  EY    D   A   +(                                                         RDWR0113
BTXT  +)    8   AAM&&0QK 0QJ &,Q&20G)Q-&).-&K&-&)A&A4&&0*K&0*J &,Q024G.Q-RDWR0114
BTXT  E(    D   A   ++                                                         RDWR0115
BTXT  +D    8   A&).-&K&-&)A&0&+MA& MQ020E A&A4Q020E A&AMQ020E A&A4Q020E RDWR0116
BTXT  E0    D   A   +M                                                         RDWR0117
BTXT  +'    8   AQ02UE Q 28Q 1MQ 1-Q&1-G'Q02(E KH Y 83D 8 -Q02UE Q 2'Q 1MRDWR0118
BTXT  E4    D   A   +=                                                         RDWR0119
BTXT  E8    D   A   +*                                                         RDWR0120
```

```
BTXT  +4   8   AQ 1-Q01-G=Q02(E Q02UE Q 2 Q 1-Q 1)Q&1)G'Q02(E Q02UE Q 2DRDWR0121
BTXT  E'   D   A  +0                                                  RDWR0122
BTXT  +(   8   AQ 1MQ 1-Q01-G=Q02(E Q&2HA-&H&-0 K)& BRA&& K.&  QA&&)Q1Q&RDWR0123
BTXT  E    D   A  .M                                                  RDWR0124
BTXT  .U   8   A2D$1 0& +TKD0& &K.08 A .080&E-N 080 Q&2HG'K 0 0 D-K+ H09RDWR0125
BTXT  .*   8   AQ&2HA-&H&-0 KZ& B7A&&&K+&  HA&&.Q1Q&2D$1 0& +TQ&2HA-&H&-RDWR0126
BTXT  DD   D   A  .&                                                  RDWR0127
BTXT  .M   8   A0 KL& B/A&&MQ1Q&2D$1 0& +TQ--DQ(-)$ G=Q02YE 2T0= (2T0M 'RDWR0128
BTXT  ED   D   A  .0                                                  RDWR0129
BTXT  .)   8   A9S0=0MQ02)G,231Y '230  (= 1Y0 3  '1Y00  Q&2&G'8118B$P 0 RDWR0130
BTXT  )D   8   A0 K+0 19M00+M 0 J 0 1GK 180 230  '0 180 K 1Y18230  (= 1YRDWR0131
BTXT  EH   D   A  )                                                   RDWR0132
BTXT  )'   8   A0 3  '1Y00  211Y 0=01YBQ3  01Y00 =2J0= &2J0M 09J0=0MQ02MRDWR0133
BTXT  E)   D   A  )Q                                                  RDWR0134
BTXT  )4   8   AG,211Y 0210  &= 1Y0 3  01Y00 =Q&2QG'8118B$210  00 180 K RDWR0135
BTXT  E&   D   A  )*                                                  RDWR0136
BTXT  )(   8   A1Y18210  &= 1Y0 3  01Y00 =211Y 8=01YBQ3  81Y00 92J0= Y2JRDWR0137
BTXT  EM   D   A  (0                                                  RDWR0138
BTXT  )U   8   A0M 89J0=0MQ02*G,211Y 8210  Y= 1Y0 3  81Y00 9Q&2-G'8118B)RDWR0139
BTXT  EQ   D   A  (D                                                  RDWR0140
BTXT  DH   D   A  (+                                                  RDWR0141
BTXT  (*   8   A210  80 180 K 1Y18210  Y= 1Y0 3  81Y00 9Q02YE KA Q 8KA $RDWR0142
BTXT  D)   D   A  (4                                                  RDWR0143
BESD       &   FGETTIME B                                             RDWR0144
BTXT  (M   8   A 02T0& 'KE08 ( E080&KD  09Q02YE A& -&&0&K&0&A&0&Q024E Q0RDWR0145
BTXT  D-   D   A  (-                                                  RDWR0146
BTXT  ()   D   A2YE                                                   RDWR0147
BRLD       8     A AI C'I DAI DEI DII D(I DJI DNI DRI D)I D/I DVI DZH D RDWR0148
BRLD       8     A AI D1I D5I D9I D'I DAI DEI DII D(I DJI DNI DRI E() D)RDWR0149
BRLD       8     A A( D-( DM( DQ( D*( D ( DU( DY( D(( D0( D4( D8( D') D&RDWR0150
BRLD       8     A A( DD( DH( D)( D-( DM( DQ( D*( D ( DU( DY( D(( D0) D4RDWR0151
BRLD       8     A A( D8( D'( E ( ED( EH( E)( E&( EM( EQ( E*( E-( EU) EYRDWR0152
BRLD       4     A A( E(( E0( E4( E8( E'( E ( ED( EH( E)( E&( EM) EQ    RDWR0153
BRLD       Y      B A* E* C A* E- D A* EU E A* EY F A* E(             RDWR0154
BEND  E4       A             (-                                       RDWR0155
BESD       &   AGETTIME          8                                        0001
BTXT       ,   A-T-)E0A 0&&  H&  DQ Q/A& B+.Q&- & & Q) DQT-)G=            0002
BEND           A                                                          0003
 NAME HIGHIO(R)
/*

****  CLIP DECK AND SAMPLE CONTROL CARDS
  D    (8B (8-  &B +H-  &
E-Q&-- & K$=H& BQA &&M &&-H& HA-A H- A.&&UG &SB-&Q&0 8A&&,&& 'A0 &&0 HQ &MQ-&-A0
 H) & G0&K& &Q* & G&&&B &-B &QJ  EG&&WR & G&&+KG- & -TG0&KQ&&DG5   &BEND B   (8
  D+   +       DS    I                             E0             0        0000
  D0   DS                         E+                        &0 ME0Q05+J    0001
5&G&0 -85&KC5& MA-0DH6 BI0 0G&1'K7G 1+AW )G00UPE5Y5Y-C50B 5HM 5&NA5652G&   0002
0&QC50KA5652G10A- KG 03-G00HI05,G01)KA5,52KG5 2YQ&G1G01&G01MM 5&KG 85QQ8   0003
5&B  8G01&) 0 G&1=G01,J= DG01,KC5*5DJB- G&1KKA5Y-BG03) 8M 5&* 0 G&0WG-0D   0004
K350KA5PG&2-0A- 0&5&G02 G01SJB DG&3YJA DG&3UG01QJ( DG&1-JD DG&1BA&1BM0-    0005
NA-B5YG01&KA5Y52G8J&5&G&0)J 5&G&1 M' 9G01SJ& DG&0DK150KD5PJ&5&G&3 K050M    0006
5&E&1(G02BG02FK 2F2FH& 0M&2)I&0KG&2FD 2&A&4SA-5NA0 CA-0&E 2MD 24D 28D 2=   0007
K 2 2 B 5&     G -85(K 2Q5-KL2D 8KA5,-BKG2Y5 PG5 5 G1Q85(KL 82DK 5-2Q     0008
                                  D 2&A&40KA2=-BE 2MD 24D 28D 2=G8         0009
NA-B2=G010G014NA-B2=G026KA2=52A&1BG010J  EG01&   0 000 0G016KG Q3YKG 03-   0010
KG 830P  8 8G01Q        AW      AD&0 ME0Q02-J 5&G&3*-85&KC5& MKM50KW5PG010 0011
     C8G030G01SNA-B06G03(JD- G&1YE&20J75YG03 J 5YG&3 JH5YG03WG01SG03(H&    0012
50AH A &50.&48G-1SD 2&D 44J 4 G&3+A&4+E 2MD 24D 28D 2=Q85&G1G03(G04 G030   0013
```

```
G034G038G03'G03YKB5)5NG01Y&0 MG0&QKB5DKB5YA-0DG05S&0 MG0&&KB5DKI5YA-0&G0      0014
5S&0 MG0&YKB5DG04+A0 FA-5YAJ QG05&KD5YG05S&0 MG0&&KB5DA-5*A0 AAJ MG05&G0      0015
4,0B5DK 4 G04Q0B5DK 4 A-04G05S  K 49A-  A0 AAJ DG05&Q00   &0 MG0&YK+49KB      0016
5YG04+&0 MG0&MG04 KX5YG04BE0Q004& 55&& 8J 5&G&5&D 0=-85&KC5& MG058KA5052      0017
G05(K-5(KC H5 G1 H&F DG06HM'- 0 - 5DK 5DQ& 8G01=  EY     F     A)AB  0I0A      0018
 D             AW       AD-T5YNA5Y52                                          0019
            KG Q5-KG 85YD 0&KC5&50KE0&54KA5652G05&          &                  0020
              &05 M06-0 5&G05+H&5 KHG&60 F BH6 BG05&       & &Q$3QCA-&+          0021
N0& G &6N9& G2M & C0& I  D-CAN AF &)Q-&QG2NA& GBNF& G2C0& .0&0G0&S&0 *B      0022
&* 7    AF   LKDE0A& B &5MKC5+5=KG Y50E0$J&&5D0-5'Q&54QJ  A-40Q058KE50A      0023
4)EA$WA-4MA0 AA-0+Q&40R&5HG00*N 46G00+0-5'G00 A  FA&46Q&50EAQ-N 5 G00&KK      0024
50J-5'G&4)$ M 5'--&-58G'A-4)KU50E9A 04H &FI  HC &HG=R&5)G000KK5ON 5HG&4)      0025
A  DA&5HQ&50EAQ-Q'EZQ Q&58R-G&0SJG5=G&0S-$&&58G00-B* HQAH& H * F-F&&58$W      0026
G=R&5-G012A-10M 5HN 5HG&1UJ&5'G)KA5ONC5HG 4)NB5HG 12H05+E 42B-5HE,Q2-05H      0027
A-10E9E 0+KD5OROG04)G'B' HH0 H ' FG'M 5'A-1SE9E 0+Q&H-4=BB5-B-5HQ-5HRKG&      0028
1-$KB* IH& H * +G'$/LKG01)B-5)Q-5)A 1,G00FA-4)KU50E9H-4=BB5-AS A -4=E 0+      0029
Q&G01)A- &H&40$KG&0+K 5 5& &40G01BR&5MG02+B-44H040A-2&J&5'G&2M$&G02-H04=      0030
E 42E,QS- 44& 44Q--PK050R 5(G 2 R&5'G-4)J&5'G&2UH-5=A0 )Q&54KSG&2**S$CR*      0031
G-4)J 5'G 2=A  &R&G&2=M 5'&  5DG9$=B02GK   5 G'R&5QG03DA  FA&46Q&50EA& 44      0032
A  BA&4=Q&50EA  4=A&5 A0 BJH5'G&2'E-2FA  DQ&50EAQ&44  & M75'N,& G'0H5'AH      0033
 B&&44-$G02=A-2+G02+R&5*G03WJ&5'G)A 5 H0  KA50K3G&4)E 42E,QBH0 BK3G&4)E      0034
42E,QKH&36H-3-E 3,-VQ1QBJ)  G 3F$$B&35B&3(-6-F 036  3-Q0  -9K 5&0 Q 5&JB      0035
  G&3&-H& 5&K 0 5& &36 -3-E 3,$WJA  G&3+-VG 30A& DH 40$EG(  40G=$HG03H-,      0036
$$JH  G&3Q-,$$JD  G&3Q-,$$G03QR&5 G04,$SN 45G&4SB-44J&5'G&4SM 5'J&5'G&4-      0037
H04=E 42E,--44&-5DP 5-5-0&5'G'R&5UG'N 5 G04UR-5DG&4MQ 5D& 5'E-4SB 58A0 A      0038
E 42E 0+G04HA-4)KU50E9G04*$SQ&54CT5-A0 )*S$CE,H( +I- HC( IG=H 5=A&5 $DA&      0039
 )Q&54K G&4'$E-.NE& & GCF 4($E-.  5=KE& & G2KG Q5HA-5NA0 HA 4YQ&50QJ  EA      0040
B 5&                                                                          0041
        D &    AF   LPAAD    G AF   LKD WAIT                          FU       0042
       &        BSLCBICSBESDBTXTBREPBRLDBENDBLDT  )   )    )   &   H7        0043
E0Q00+$,&  8Q&'QN &&G&'HA  DAJ &A&6UEAQ& &'QQ-'QN -MG&'6A  DA) MA&6UEAQ&      0044
'M&A &G0'BG0'6KA'4='KC=2'2A0'8&0 H&&'MA0'&&0 ') & G0'0& ' * & B '&I& 0G&      0045
'6B  8JD DG&'WNF'A'IG&'Q &'QA  0- 'M& 'MA  QQ&'M$&KP& '*Q0'4G      &  GH      0046
D (&   H                F        (8LOAD2  GH   RESUME G        D (&   H      0047
BEND  )       01    001F
BESD      &   A        D &   D*                                               0001
BTXT  &   8   AE0KG 81WKG -30+.KG Q1,A- BK+1,K(15A-2F -10K 4-46K 2,Q-1&      0002
BTXT  &8  8   AA 1,+&NC2 2FG&04NC2S2FG&0 ND2W2FG&0KQ-1WA&2( &10KI1,KQ15      0003
BTXT  &0  8   A+&B 1 KB2H2.2T2Q2.Q-1WA 1 +&F-02G00YKE1=2)F-02G00YK 2,G0      0004
BTXT  &Y  8   A04Q-20J 2,G&10A 16Q-20KB10KG1+10+-KAIU2DKG1+1 KC10+-KE2S      0005
BTXT  &   8   A4 KI1,A-2R -10K&15Q-1WA 1,+&KE4 1=KA1(2DKG1+1WQ-20A 16+-      0006
BTXT  JQ  8   AKE221=KI1,K&15Q-1WA 1,A-2Z -10+&B 1 KE4 1=Q-20A 1F+&A 1+      0007
BTXT  J&  J   A+&A 16+&G010                                                   0008
BTXT  J-  8   A B    J8 B    J8      &0G J0   F1 J8   EH J&                  0009
BTXT  JQ  8   AK K&   HF M*    E M*    B KH-  (   -  A   -  AG  -  A         0010
BTXT  J-  8   A,   -  AA M*-  &I KH   I       )          B                   0011
BTXT  KH  &   A                                                               0012
BTXT  KQ  8   A        DISKTAPELABEL CONTROL STATEMENT ERROR ERR               0013
BTXT  K&  ,   A           OLD ID =        NEW ID =                            0014
BTXT  KH  8   A-)32Q)3,NA4, SG&3&NA40 SG&2FNA42 SG&3FNA44 SG&32B 3 -)3,       0015
BTXT  K&  8   AQ)32B  -&  H&*  A) D&(  A) DKG&   A) HJ& DG&24A, HG02-JH       0016
BTXT  K8  8   A  A, HG02-Q  HG8&  H) - G03+* - G-3K) - G03-G 3YGHJA DG&      0017
BTXT  L0  8   A3+GH&  H) - &  H* - G03=G&3+E-3Q) - G03+GHJB DGIG03SA 3W      0018
BTXT  LY  0   AQHE&2FE&3FKG4W -KG -3 QDG8P)3232A&20KG Q30A&46G8               0019
BTXT  LQ  *   A      KH B   IODD L0   D                                       0020
BTXT  L4  8   A                                                               0021
BTXT  L(  D   A                                                               0022
BTXT  L0  8   A                                                               0023
BTXT  M0  H   A . ) & -                                                       0024
```

```
BTXT   M8    8    AVOL1       0              JACK BLOMQUIST                          0025
BTXT   MO    ( ^  A                                                                  0026
BRLD         8      A A( JU( J(( J4I J9I JAI JII JRI J/I JZI J1I JRI J/) L*         0027
BRLD         H      A AH LZ                                                         0028
BEND                                                                                0029
BLDT
DISK=191
LABEL=SER=222222
TAPE=282
LABEL=SER=ABCABC

//WRITETWO DD   UNIT=(2400,,DEFER),LABEL=(,NL)
//WRITEONE DD   UNIT=(2400,,DEFER),LABEL=(,NL)
//WRRDONE  DD UNIT=2311,SPACE=(CYL,(25,10))
//WRRDTWO  DD UNIT=2311,SPACE=(CYL,(25,10))
//SYSUT2   DD  UNIT=2540-2,DCB=(RECFM=F,BLKSIZE=80)
//SYSIN    DD  *
//SYSUT1   DD  UNIT=(2400,,DEFER),LABEL=(,NL),VOLUME=SER=123123,          C
//SYSPUNCH  DD  UNIT=2540-2
//SYSPUNCH  DD  UNIT=2540-2
//SYSUT2   DD  SYSOUT=A,DCB=(RECFM=F,BLKSIZE=80)
//SYSUT1   DD  UNIT=(2400,,DEFER),LABEL=(,NL),VOLUME=SER=ABCABC,          C
//SYSIN    DD  *
//PRINTER  DD  SYSOUT=A
 PUNCH TYPORG=PS,MAXFLDS=1,STOPAFT=100
    RECORD  FIELD=(80,1,,1)
/*

//  SHIFT 1
//PRINTAP  DD UNIT=(282,,DEFER),DISP=OLD,VOLUME=SER=ABCABC,               X
**** SAMPLE JCL TO PRINT SYSOUT TAPE
//IEFPRT   JOB 1,STRAUSS,MSGLEVEL=1
//PRINT   EXEC  PGM=IEFPRT
//PRINTAP  DD UNIT=(282,,DEFER),DISP=OLD,VOLUME=SER=ABCABC
//PRINTER  DD  UNIT=00E,DCB=(RECFM=F,BLKSIZE=80)
//  START RDR,00C

./       ADD   CTLCDX,01,0,1
 PRINT TYPORG=PS,MAXFLDS=1,STOPAFT=100
    RECORD  FIELD=(80,1,,1)
/*

//   VARY 192,ONLINE
**** SAMPLE OPERATOR COMMANDS
//  START RDR,182


**** JCL AND SOURCE TO COMPILE HIGH IO JOB
//COBSOUR  JOB  1,STRAUSS,MSGLEVEL=1
//COBOL    EXEC PGM=IEPCBL00
//SYSPRINT DD SYSOUT=A
//SYSUT1   DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSPUNCH DD UNIT=2311,SPACE=(CYL,(5,1)),DISP=(NEW,KEEP),DSNAME=A
//SYSUT2   DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSUT3   DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSIN    DD   *
       IDENTIFICATION DIVISION.
        PROGRAM-ID. 'RDWRDEMO'.
       ENVIRONMENT DIVISION.
```

```
 CONFIGURATION SECTION.
  SOURCE-COMPUTER. IBM-360 E30.
  OBJECT-COMPUTER. IBM-360 E30.
 INPUT-OUTPUT SECTION.
  FILE-CONTROL.
     SELECT WR-RD-ONE ASSIGN 'WRRDONE' UTILITY.
     SELECT WR-RD-TWO ASSIGN 'WRRDTWO' UTILITY.
     SELECT WRITEONE ASSIGN 'WRITEONE' UTILITY.
     SELECT WRITETWO ASSIGN 'WRITETWO' UTILITY.
     SELECT CARDIN ASSIGN 'CARDIN' UTILITY.
DATA DIVISION.
 FILE SECTION.
  FD WR-RD-ONE
     RECORDING MODE IS F
     LABEL RECORD OMITTED
     DATA RECORDS ARE RECORDA.
   01 RECORDA PICTURE X(80).
  FD WR-RD-TWO
     RECORDING MODE IS F
     LABEL RECORD OMITTED
     DATA RECORDS ARE RECORDB.
   01 RECORDB PICTURE X(80).
  FD WRITEONE
     RECORDING MODE IS F
     LABEL RECORD OMITTED
     DATA RECORDS ARE RECORDC.
   01 RECORDC PICTURE X(80).
  FD WRITETWO
     RECORDING MODE IS F
     LABEL RECORD OMITTED
     DATA RECORDS ARE RECORDD.
   01 RECORDD PICTURE X(80).
  FD CARDIN
     RECORDING MODE IS F
     LABEL RECORD OMITTED
     DATA RECORDS ARE RECORDE.
   01 RECORDE PICTURE X(80).
 WORKING-STORAGE SECTION.
  01 PERFORM-TIME.
   02 PERFORM-TYPE.
     03 FILLER PICTURE X(13).
     03 PERFORM-COUNT PICTURE 9.
     03 FILLER PICTURE X(6).
   02 PERFORM-TERMINAL PICTURE X(7).
   02 FILLER PICTURE X(4) VALUE ' AT '.
  01 DAY-TIME.
   02 DEC-TIME PICTURE X(4).
   02 FILLER PICTURE X VALUE ' '.
  01 FILLER REDEFINES DAY-TIME.
   02 PACK-TIME PICTURE S9(9) COMPUTATIONAL-3.
  01 HOLD-CARD PICTURE X(80) VALUE ALL 'X'.
  01 CARDS-READ PICTURE 9(4) VALUE ZERO.
  01 RECORD-COUNT PICTURE S9(9) COMPUTATIONAL-3.
  01 RDWRCOUNT PICTURE ZZZ,ZZZ,999.
  01 FORMAT-TIME.
   02 FOR-HOURS PICTURE 99.
   02 FILLER PICTURE X VALUE ' '.
   02 FOR-MINUTES PICTURE 99.
   02 FILLER PICTURE X VALUE ' '.
```

```
    02 FOR-SECONDS PICTURE 99.99.
    02 FILLER PICTURE X VALUE ' '.
  01 COMP-TIME.
    02 COMP-HOURS PICTURE 99.
    02 COMP-MINUTES PICTURE 99.
    02 COMP-SECONDS PICTURE 99V99.
    02 FILLER PICTURE X.
  01 DISPLAY-TIME PICTURE S9(9).
    01 FILLER REDEFINES DISPLAY-TIME.
    02 HOURS PICTURE 99.
    02 MINUTES PICTURE 99.
    02 SECONDS PICTURE 99V99.
    02 FILLER PICTURE X.
PROCEDURE DIVISION.
 START.
     OPEN OUTPUT WR-RD-ONE.
     OPEN OUTPUT WR-RD-TWO.
     MOVE ZEROES TO RECORD-COUNT.
     PERFORM GET-TIME. MOVE PACK-TIME TO DISPLAY-TIME.
 WRITE-RECORD.
     MOVE HOLD-CARD TO RECORDA, RECORDB.
     WRITE RECORDA.
     WRITE RECORDB.
     ADD 2 TO RECORD-COUNT.
     IF RECORD-COUNT , 04999 GO TO FINISH-WRITE-PHASE.
     GO TO WRITE-RECORD.
 FINISH-WRITE-PHASE.
     PERFORM GET-TIME.
     CLOSE WR-RD-ONE, WR-RD-TWO.
     PERFORM TIME-FORMAT.
     MOVE DISPLAY-TIME TO COMP-TIME.
     MOVE PACK-TIME TO DISPLAY-TIME.
     PERFORM TIME-FORMAT.
     PERFORM ELAPSED-TIME.
     PERFORM TIME-FORMAT.
     DISPLAY 'ELAPSED TIME WRITE PHASE ' FORMAT-TIME UPON CONSOLE.
     MOVE RECORD-COUNT TO RDWRCOUNT.
     DISPLAY 'TOTAL LOGICAL RECORDS WRITTEN -' RDWRCOUNT UPON
      CONSOLE.
 RD-WR-PHASE.
     OPEN INPUT WR-RD-ONE.
     OPEN INPUT WR-RD-TWO.
     OPEN OUTPUT WRITEONE.
     OPEN OUTPUT WRITETWO.
     MOVE ZEROES TO RECORD-COUNT.
     PERFORM GET-TIME. MOVE PACK-TIME TO DISPLAY-TIME.
 BRANCH-2.
     GO TO RD-WR-ONE.
 RD-WR-ONE.
     READ WR-RD-ONE AT END ALTER BRANCH-4 TO PROCEED TO BRANCH-3,
      ALTER BRANCH-2 TO PROCEED TO FINISH-RD-WR-PHASE, GO TO
      BRANCH-3.
     MOVE RECORDA TO RECORDC.
     WRITE RECORDC.
     ADD 1 TO RECORD-COUNT.
 BRANCH-3.
     GO TO RD-WR-TWO.
 RD-WR-TWO.
     READ WR-RD-TWO AT END ALTER BRANCH-3 TO PROCEED TO BRANCH-2,
```

```
            GO TO BRANCH-2.
           MOVE RECORDB TO RECORDD.
           WRITE RECORDD.
           ADD 1 TO RECORD-COUNT.
         BRANCH-4.
           GO TO RD-WR-ONE.
         FINISH-RD-WR-PHASE.
           PERFORM GET-TIME.
           CLOSE WR-RD-ONE, WR-RD-TWO, WRITEONE, WRITETWO.
           PERFORM TIME-FORMAT.
           MOVE DISPLAY-TIME TO COMP-TIME.
           MOVE PACK-TIME TO DISPLAY-TIME.
           PERFORM TIME-FORMAT.
           PERFORM ELAPSED-TIME.
           PERFORM TIME-FORMAT.
           DISPLAY 'ELAPSED TIME READ-WRITE PHASE ' FORMAT-TIME UPON
            CONSOLE.
           MOVE RECORD-COUNT TO RDWRCOUNT.
           DISPLAY 'TOTAL LOGICAL RECORDS READ AND REWRITTEN -'
            RDWRCOUNT UPON CONSOLE.
           DISPLAY 'END OF DEMONSTRATION' UPON CONSOLE.
           STOP RUN
       ELAPSED-TIME.
           IF COMP-SECONDS NOT , SECONDS, SUBTRACT COMP-SECONDS FROM
            SECONDS, ELSE COMPUTE SECONDS = 60 + SECONDS - COMP-SECONDS,
            SUBTRACT 1 FROM MINUTES.
           IF COMP-MINUTES NOT , MINUTES, SUBTRACT COMP-MINUTES FROM
            MINUTES, ELSE COMPUTE MINUTES = 60 + MINUTES - COMP-MINUTES,
            SUBTRACT 1 FROM HOURS.
           IF COMP-HOURS NOT , HOURS, SUBTRACT COMP-HOURS FROM HOURS,
            ELSE COMPUTE HOURS = 24 + HOURS - COMP-HOURS.
         TIME-FORMAT.
           MOVE HOURS TO FOR-HOURS.
           MOVE MINUTES TO FOR-MINUTES.
           MOVE SECONDS TO FOR-SECONDS.
         GET-TIME.
           ENTER LINKAGE.
           CALL 'GETTIME' USING DEC-TIME.
           ENTER COBOL.
/*
//ASSEM   EXEC  PGM=IEUASM
//SYSLIB    DD   DSNAME=SYS1.MACLIB,DISP=OLD
//SYSPRINT  DD   SYSOUT=A
//SYSPUNCH  DD  UNIT=2311,SPACE=(CYL,(5,1)),DISP=(NEW,KEEP),DSNAME=B
//SYSUT1    DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSUT2    DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSUT3    DD  UNIT=2311,SPACE=(CYL,(5,1))
//SYSIN     DD   *
GETTIME  START                                                  TIME0001
         STM    14,3,12(13)                                     TIME0002
         BALR   3,0                                             TIME0003
         USING  *,3                                             TIME0004
         LA     14,SAVEAREA                                     TIME0005
         ST     14,8(13)                                        TIME0006
         ST     13,4(14)                                        TIME0007
         LR     13,14                                           TIME0008
         LR     2,1                                             TIME0009
         TIME   DEC                                             TIME0010
       L      1,0(0,2)                                          TIME0011
```

```
         ST      0,0(0,1)                                              TIME0012
         L       13,4(13)                                              TIME0013
         LM      14,3,12(13)                                           TIME0014
         BR      14                                                    TIME0015
SAVEAREA DS      18F                                                   TIME0016
         END     GETTIME                                               TIME0017
/*

IEB400I END OF DATA FOR SDS OR MEMBER
```

STUDENT PROBLEM 1

SYSTEM UTILITIES

This exercise is designed as a work project. It is to be coded on the normal coding form but not punched up to be executed.

A. Using "IEHPROGM" perform the following:

1. Construct a catalog on WORK01 with the following specifications:

   a. Build three levels of index (i.e., A. B. C)

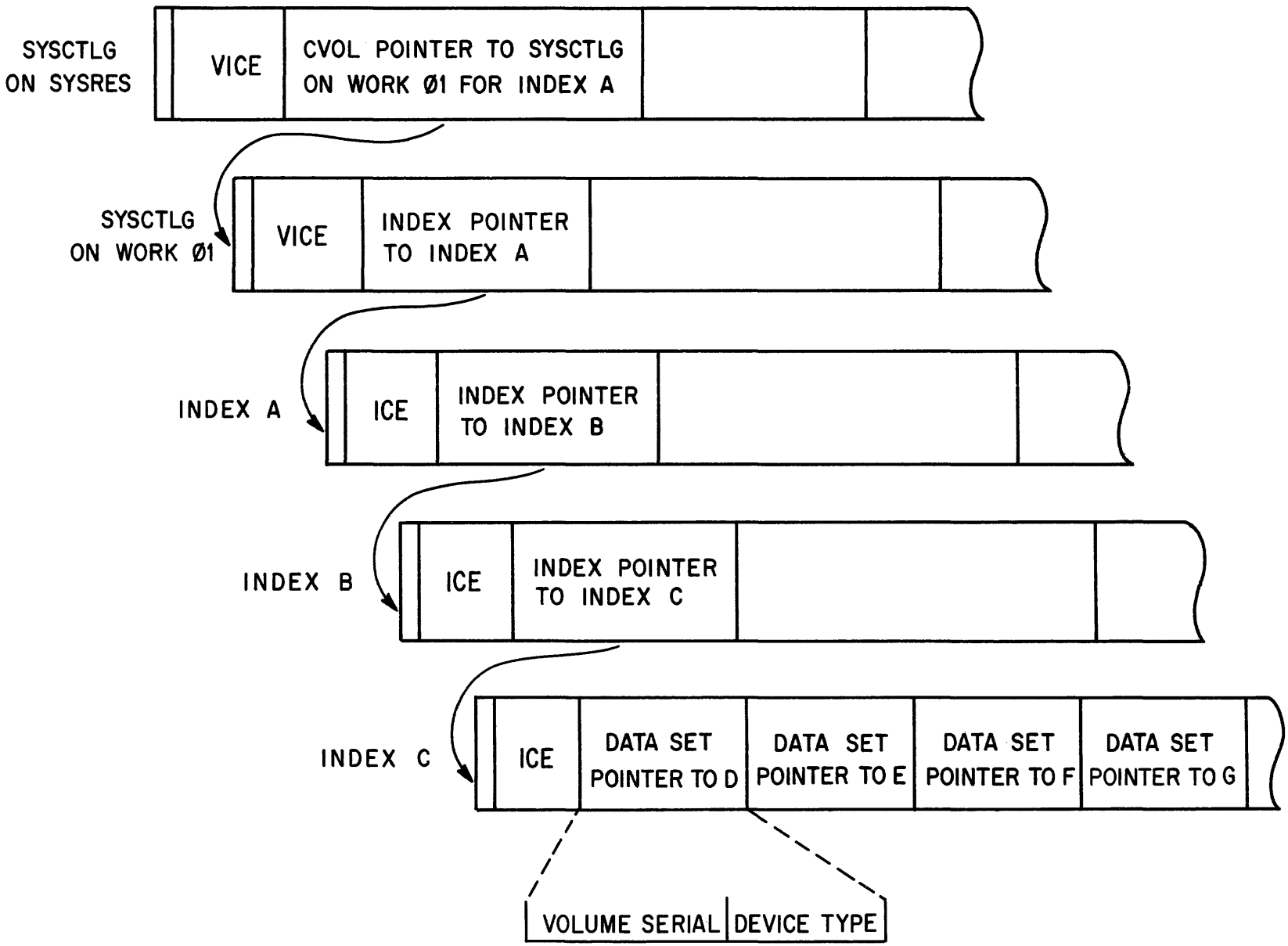
   b. Catalog four data sets using four unique serial numbers

   c. Connect your catalog to the one residing on SYSRES

2. Draw a "picture" of what your catalog structure should look like. Include in your "picture" of all the necessary catalog control entries.

IEHPROGM - STRUCTURING A CATALOG ON WORK01

```
//REK9       JOB      'CLASS 6016',PRGNM,MSGLEVELS=1
//STPA       EXEC     PGM=IEHPROGM
//DD1        DD       DSNAME=SYSCTLG,VOLUME=SER=WORK01,UNIT=2311,          c
//                    DISP=OLD,SPACE=(TRK,(5,5))
//SYSPRINT   DD       SYSOUT=A
//SYSIN      DD       *
           CATLG      DSNAME=A.B.C.D,VOL=2311=WORK01
           CATLG      DSNAME=A.B.C.E,VOL=2311=WORK01
           CATLG      DSNAME=A.B.C.F,VOL=2311=WORK01
           CATLG      DSNAME=A.B.C.G,VOL=2311=WORK01
           CONNECT    INDEX=A,VOL=2311=WORK01
/*
```


SYSCTLG
ON SYSRES
VICE
CVOL POINTER TO SYSCTLG
ON WORK Ø1 FOR INDEX A
SYSCTLG
ON WORK Ø1
VICE
INDEX POINTER
TO INDEX A
INDEX A
ICE
INDEX POINTER
TO INDEX B
INDEX B
ICE
INDEX POINTER
TO INDEX C
INDEX C
ICE
DATA SET
POINTER TO D
DATA SET
POINTER TO E
DATA SET
POINTER TO F
DATA SET
POINTER TO G
VOLUME SERIAL
DEVICE TYPE

STUDENT PROBLEM 2

DATA SET UTILITIES

A. Using "IEBPTPCH" perform the following:

1. Read data cards from the reader (00C) and print them on the printer (00E)

   NOTE: This operation may require another job step prior to the execution of IEBPTPCH !

B. Using "IEBGENER" perform the following:

1. Read data cards from the reader (00C) and put them on tape.

C. Using "IEBCOPY, IEBUPDAT", and "IEBCOMPR" perform the following:

1. Copy any member from SYS1.PROCLIB to a data-set for which you have allocated space.

2. Change a record within your copied member.

3. Compare the original member against your changed member.

4. Use the unequal compare exit and type on the console some indicative message.

5. Make provisions for deleting your data-set at the completion of the compare step.

```
IEF403I REKUT1   STARTED
IEF236I ALLOC. FOR REKUT1   STPA
IEF237I SYSIN    ON OOC
IEF236I ALLOC. FOR REKUT1   STPA2
IEF237I SYSUT1   ON OOC
OO IEC107D E 191,SYSRES,SYS1.PROCLIB
REPLY OO,'U'
IEF236I ALLOC. FOR REKUT1   STPC1
IEF237I SYSIN    ON OOC
IEF236I ALLOC. FOR REKUT1   STPA3
IEF237I SYSUT1   ON OOC
IEF233A M 182,SCRTCH,REKUT1
IEF236I ALLOC. FOR REKUT1   STPA4
IEF237I SYSUT1   ON OOC
IEF236I ALLOC. FOR REKUT1   STPB
IEF237I SYSIN    ON OOC
IEF236I ALLOC. FOR REKUT1   STPB1
IEF237I SYSIN    ON OOC
IEF236I ALLOC. FOR REKUT1   STPB
IEF237I SYSIN    ON OOC
IEF236I ALLOC. FOR REKUT1   STPC
IEF237I SYSIN    ON OOC
```

THIS MESSAGE IS THE RESULT OF AN UNEQUAL COMPARE BETWEEN SYS1.PROCLIB(DICK) AND KEATON(DICK).

```
IEF236I ALLOC. FOR REKUT1   STPC2
IEF237I SYSIN    ON OOC
00 IEC107D E 191,SYSRES,SYS1.PROCLIB
REPLY 00,'U'
IEF236I ALLOC. FOR REKUT1   STPC3
IEF237I SYSIN    ON OOC
IEF404I REKUT1   ENDED
```

```
//REKUT1 JOB    'CLASS 6016','KEATON''S UTLTY PRJT',MSGLEVEL=1
//JOBLIB DD     DSNAME=PRILIB,UNIT=2311,SPACE=(TRK,(1,1,1)),              C
//              VOLUME=SER=WORK01,DISP=(NEW,PASS)
//STPA   EXEC   PGM=ASMBLR
//SYSUT1 DD     DSNAME=UT1,DISP=OLD
//SYSUT2 DD     DSNAME=UT2,DISP=OLD
//SYSUT3 DD     DSNAME=UT3,DISP=OLD
//SYSLIB DD     DSNAME=SYS1.MACLIB,DISP=OLD
//SYSPRINT DD   SYSOUT=A
//SYSPUNCH DD   UNIT=2311,VOLUME=SER=WORK01,DISP=(,PASS),                 C
//              SPACE=(TRK,(1,1))
//SYSABEND DD    SYSOUT=A
//SYSIN  DD     *
IEF236I ALLOC. FOR REKUT1   STPA
IEF237I JOBLIB   ON 192
IEF237I SYSUT1   ON 19C
IEF237I SYSUT2   ON 192
IEF237I SYSUT3   ON 192
IEF237I SYSLIB   ON 19C
IEF237I SYSPUNCH ON 192
IEF237I SYSIN    ON 00C
```

SYMBOL	TYPE	ID	ADDR	LENGTH	LD ID
ERRTNE	SD	01	000000	0000E4	

```
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                F24JUNE66 10/26/66

000000                             1 ERRTNE   CSECT
                                   2          SAVE  (14,12)
000000                             3+         DS    0H
000000 90EC D00C             0000C 4+         STM   14,12,12(13) SAVE REGISTERS
000004 0520                        5          BALR  2,0
000006                             6          USING *,2
000006 4180 2096             0009C 7          LA    8,SAVEAREA
00000A 508D 0008             00008 8          ST    8,8(13)  STORE BKWRD POINTER
00000E 50D0 209A             000A0 9          ST    13,SAVEAREA+4  STORE FWD POINTER
000012 4100 2096             0009C 10         LA    13,SAVEAREA
                                   11         WTO   'THIS MESSAGE IS THE RESULT OF AN UNEQUAL COMPARE'
000016 0700                        12+        CNOP  0,4
000018 4510 204A             00050 13+        BAL   1,IHB0002A BRANCH AROUND MESSAGE
00001C 0034                        14+        DC    AL2(IHB0002-*) MESSAGE LENGTH
00001E 0000                        15+        DC    AL2(0)
                                   16+        DC    C'THIS MESSAGE IS THE RESULT OF AN UNEQUAL COMPARE' MESSX
000020 E3C8C9E240D4C5E2              +              AGE
000050                             17+IHB0002 EQU   *
000050                             18+IHB0002A DS   0H
000050 0A23                        19+        SVC   35 ISSUE SVC
                                   20         WTO   'BETWEEN SYS1.PROCLIB(DICK) AND KEATON(DICK).'
000052 0700                        21+        CNOP  0,4
000054 4510 2082             00088 22+        BAL   1,IHB0003A BRANCH AROUND MESSAGE
000058 0030                        23+        DC    AL2(IHB0003-*) MESSAGE LENGTH
00005A 0000                        24+        DC    AL2(0)
00005C C2C5E3E6C5C5D540            25+        DC    C'BETWEEN SYS1.PROCLIB(DICK) AND KEATON(DICK).' MESSAGE
000088                             26+IHB0003 EQU   *
000088                             27+IHB0003A DS   0H
000088 0A23                        28+        SVC   35 ISSUE SVC
00008A 58D0 209A             000A0 29         L     13,SAVEAREA+4
                                   30         RETURN (14,12),T,RC=(15)
00008E 58EE 000C             0000C 31+        L     14,12(13,0) RESTORE REGISTER 14
000092 980C D014             00014 32+        LM    0,12,20(13) RESTORE THE REGISTERS
000096 92FF D00C       0000C       33+        MVI   12(13),X'FF' SET RETURN INDICATION
00009A 07FE                        34+        BR    14 RETURN
00009C 0000000000000000            35 SAVEAREA DC   18F'0'
000000                             36         END   ERRTNE
```

SYMBOL	LEN	VALUE	DEFN	REFERENCES			
ERRTNE	00001	000000	0001	0036			
IHB0002	00001	000050	0017	0014			
IHB0002A	00002	000050	0018	0013			
IHB0003	00001	000088	0026	0023			
IHB0003A	00002	000088	0027	0022			
SAVEAREA	00004	00009C	0035	0007	0009	0010	0029

NO STATEMENTS FLAGGED IN THIS ASSEMBLY
53 PRINTED LINES

```
IEF285I   PRILIB                                          PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   UT1                                             KEPT
IEF285I   VOL SER NOS= DLIB02.
IEF285I   UT2                                             KEPT
IEF285I   VOL SER NOS= WORK01.
IEF285I   UT3                                             KEPT
IEF285I   VOL SER NOS= WORK01.
IEF285I   SYS1.MACLIB                                     KEPT
IEF285I   VOL SER NOS= DLIB02.
IEF285I   SYSOUT                                          SYSOUT
IEF285I   VOL SER NOS=       .
IEF285I   AAAAAAAA.AAAAAAAA.AAAAAAAA.AAAAAAAA.00000532 PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   SYSOUT                                          SYSOUT
IEF285I   VOL SER NOS=       .
//STPA1  EXEC  PGM=IEWL,PARM='XREF'
//SYSUT1 DD    DSNAME=UT1,DISP=OLD
//SYSLIN DD    DSNAME=*.STPA.SYSPUNCH,DISP=(OLD,DELETE)
//SYSLMOD DD   DSNAME=PRILIB(ERRTNE),DISP=(MOD,PASS),UNIT=2311,        C
//             VOLUME=SER=WORK01
//SYSPRINT DD  SYSOUT=A
//SYSABEND DD  SYSOUT=A
IEF236I ALLOC. FOR REKUT1   STPA1
IEF237I JOBLIB   ON 192
IEF237I SYSUT1   ON 190
IEF237I SYSLIN   ON 192
IEF237I SYSLMOD  ON 192
```

```
   F-LEVEL LINKAGE EDITOR OPTIONS SPECIFIED--XREF
IEW0000   ERRTNE    NOW ADDED TO DATA SET


                                        ----  CROSS REFERENCE TABLE  ----

  CONTROL SECTION                       ENTRY

    NAME     ORIGIN  LENGTH             NAME   LOCATION     NAME   LOCATION     NAME   LOCATION     NAME   LOCATION

  ERRTNE        00      E4

  ENTRY ADDRESS        00
  TOTAL LENGTH         E4
```

```
IEF285I   PRILIB                                                  PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   UT1                                                     KEPT
IEF285I   VOL SER NOS= DLIB02.
IEF285I   AAAAAAAA.AAAAAAAA.AAAAAAAA.AAAAAAAA.00000532 DELETED
IEF285I   VOL SER NOS= WORK01.
IEF285I   PRILIB                                                  PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   SYSOUT                                                  SYSOUT
IEF285I   VOL SER NOS=       .
IEF285I   SYSOUT                                                  SYSOUT
IEF285I   VOL SER NOS=       .
//STPA2  EXEC  PGM=IEBGENER
//SYSPRINT DD  SYSOUT=A
//SYSUT2 DD    DSNAME=SYS1.PROCLIB(DICK),DISP=OLD
//SYSIN  DD    DUMMY
//SYSUT1 DD    *
IEF236I ALLOC. FOR REKUT1   STPA2
IEF237I JOBLIB   ON 192
IEF237I SYSUT2   ON 191
IEF237I SYSUT1   ON 00C
```

DATA SET UTILITY - GENERATE PAGE 0001

PROCESSING ENDED AT EOD

```
IEF285I   PRILIB                                                  PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   SYSOUT                                                  SYSOUT
IEF285I   VOL SER NOS=       .
IEF285I   SYS1.PROCLIB                                            KEPT
IEF285I   VOL SER NOS= SYSRES.
IEF285I   NULLFILE                                                KEPT
IEF285I   VOL SER NOS=       .
//STPC1  EXEC  PGM=IEHLIST
//DD1    DD    UNIT=2311,VOLUME=SER=SYSRES,DISP=OLD
//SYSPRINT DD  SYSOUT=A
//SYSIN  DD    *
IEF236I ALLOC. FOR REKUT1   STPC1
IEF237I JOBLIB   ON 192
IEF237I DD1      ON 191
IEF237I SYSIN    ON 00C
```

DIRECTORY INFO FOR SPECIFIED PDS ON VOL SYSRES
SYS1.PROCLIB

MEMBERS	TTRC	VARIABLE USER DATA ---(USER DATA AND TTRC ARE IN HEX)
ADDPROCS	00000702	00063066
ASMEC	00000C02	00017033
ASMECL	00010202	00013040
ASMECLG	00011302	00013040
ASMFC	00020C02	01010059
ASMFCL	00021502	01010059
ASMFCLG	00030D02	01010059
ASM1	00040702	00062966
ASM1E	00050402	00629660
ASM1LINK	00060302	00052466
ASM3	00060700	
ASM3COGO	00070900	
COBEC	00080B02	01017005
COBECLG	00081202	01011015
COBELG	00090B02	01011015
COLOGO	00091602	00063000
DICK	00130C00	
EDCTLAM2	000A0F00	
FORTEC	000A1402	00019041
FORTECLG	000A1C02	00019047
FORTELG	000B1602	00013068
LIST	000C0802	00013077
LKED	000C0C02	00016043
LKEDG	000C1402	00013068
MOD	000D0202	00013068
PL1DFC	000D0602	00018018
PL1LFC	000D0D02	00018018
PL1LFCL	000D1502	00018018
PL1LFCLG	000E0B02	00018018
PL1LG	000F0402	00018018
PUNCH	000F0D00	
SCRATCH	000F1002	00063066
SORT	000F1702	00019035
SYSLIST	00100502	00063066
TASME	00100C02	00017032
TASMEG	00110502	00015065
TASMEGED	00111902	00016043
TTED	00121A02	00015065

```
IEF285I   PRILIB                                                PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   SYS1.PROCLIB                                          KEPT
IEF285I   VOL SER NOS= SYSRES.
IEF285I   SYSOUT                                                SYSOUT
IEF285I   VOL SER NOS=       .
//STPA3  EXEC  PGM=IEBPTPCH
//SYSIN  DD    DSNAME=SYS1.PROCLIB(DICK),DISP=OLD
//SYSUT2 DD    SYSOUT=A
//SYSPRINT DD  SYSOUT=A
//SYSUT1 DD    *
IEF236I ALLOC. FOR REKUT1   STPA3
IEF237I JOBLIB   ON 192
IEF237I SYSIN    ON 191
IEF237I SYSUT1   ON 00C
```

```
PRINT/PUNCH DATA SET UTILITY                                                    PAGE 0001
         PRINT TYPORG=PS,MAXFLDS=2                                      00000010
         TITLE ITEM=('80-80 LIST OF DATA CARDS BY KEATON',33)           00000020
         RECORD FIELD=(80,,,10)                                         00000040
 EOF ON SYSIN
```

80-80 LIST OF DATA CARDS BY KEATON
IN ORDER TO PRINT THESE DATA CARDS FROM THE INPUT JOB STREAM IT WAS NECE-
SSARY TO PUT THE IEBPTPCH CONTROL CARDS OUT ON A DATA SET OTHER THAN SYSIN.
THIS IS THE LAST DATA CARD.
END OF DATA FOR SDS OR MEMBER

```
IEF285I   PRILIB                                          PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   SYS1.PROCLIB                                    KEPT
IEF285I   VOL SER NOS= SYSRES.
IEF285I   SYSOUT                                          SYSOUT
IEF285I   VOL SER NOS=       .
IEF285I   SYSOUT                                          SYSOUT
IEF285I   VOL SER NOS=       .
//STPA4  EXEC  PGM=IEBGENER
//SYSPRINT DD  SYSOUT=A
//SYSIN  DD    DUMMY
//SYSUT2 DD    UNIT=2400,LABEL=(,NL),DISP=(,PASS),                    C
//             DCB=(,RECFM=F,BLKSIZE=80)
//SYSUT1 DD    *
IEF236I ALLOC. FOR REKUT1   STPA4
IEF237I JOBLIB   ON 192
IEF237I SYSUT2   ON 182
IEF237I SYSUT1   ON 00C
```

PROCESSING ENDED AT EOD

```
IEF285I   PRILIB                                                  PASSED
IEF285I   VOL SER NOS= WORKD1.
IEF285I   SYSOUT                                                  SYSOUT
IEF285I   VOL SER NOS=       .
IEF285I   NULLFILE                                                KEPT
IEF285I   VOL SER NOS=       .
IEF285I   AAAAAAAA.AAAAAAAA.AAAAAAAA.AAAAAAAA.00000546 PASSED
IEF285I   VOL SER NOS=       .
//STPB   EXEC  PGM=IEBPTPCH
//SYSUT1  DD   DSNAME=*.STPA4.SYSUT2,DISP=(OLD,DELETE),                C
//             DCB=(,RECFM=F,BLKSIZE=80)
//SYSUT2 DD SYSOUT=A
//SYSPRINT DD  SYSOUT=A
//SYSIN DD *
IEF236I ALLOC. FOR REKUT1   STPB
IEF237I JOBLIB   ON 192
IEF237I SYSUT1   ON 182
IEF237I SYSIN    ON 00C
```

```
PRINT/PUNCH DATA SET UTILITY
         PRINT     MAXFLDS=2
         TITLE ITEM=('LISTING OF TAPE RECORDS BY KEATON',33)
         RECORD FIELD=(80,,,10)
 EOF ON SYSIN
```

LISTING OF TAPE RECORDS BY KEATON
THESE RECORDS WERE FIRST PUT OUT ON TAPE FROM DATA CARDS IN THE INPUT STREAM
BY USING IEBGENER. THEY ARE BEING PRINTED AS A CHECK.
THIS IS THE LAST RECORD.
END OF DATA FOR SDS OR MEMBER

```
IEF285I   PRILIB                                                PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   AAAAAAAA.AAAAAAAA.AAAAAAAA.AAAAAAAA.00000546 DELETED
IEF285I   VOL SER NOS=       .
IEF285I   SYSOUT                                                SYSOUT
IEF285I   VOL SER NOS=       .
IEF285I   SYSOUT                                                SYSOUT
IEF285I   VOL SER NOS=       .
//STPB1  EXEC  PGM=IEBCOPY
//SYSPRINT DD  SYSOUT=A
//SYSUT1 DD    DSNAME=SYS1.PROCLIB(DICK),DISP=OLD
//SYSUT2 DD    UNIT=2311,SPACE=(TRK,(1,1,1)),DISP=(NEW,PASS),          C
//            DSNAME=KEATON(DICK),VOLUME=SER=WORK01
//SYSIN  DD    *
IEF236I ALLOC. FOR REKUT1   STPB1
IEF237I JOBLIB   ON 192
IEF237I SYSUT1   ON 191
IEF237I SYSUT2   ON 192
IEF237I SYSIN    ON 00C
```

```
 COPY  TYPCOPY=I,MAXNAME=1
 MEMBER NAME=DICK
INCLUSIVE COPY INVOKED
DICK      SUCCESSFULLY COPIED
END OF TASK
```

```
IEF285I   PRILIB                                          PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   SYSOUT                                          SYSOUT
IEF285I   VOL SER NOS=       .
IEF285I   SYS1.PROCLIB                                    KEPT
IEF285I   VOL SER NOS= SYSRES.
IEF285I   KEATON                                          PASSED
IEF285I   VOL SER NOS= WORK01.
//STPB   EXEC  PGM=IEBUPDAT,PARM=MOD
//SYSPRINT DD  SYSOUT=A
//SYSUT2 DD    DSNAME=KEATON(DICK),UNIT=2311,VOLUME=SER=WORK01,        C
//             DISP=(MOD,PASS)
//SYSUT1 DD    DSNAME=KEATON(DICK),UNIT=2311,VOLUME=SER=WORK01,        C
//             DISP=(MOD,PASS)
//SYSIN  DD    *
IEF236I ALLOC. FOR REKUT1   STPB
IEF237I JOBLIB   ON 192
IEF237I SYSUT2   ON 192
IEF237I SYSUT1   ON 192
IEF237I SYSIN    ON 00C
```

```
                    ./      CHNGE DICK,01,0,1
                            PRINT TYPORG=PS,MAXFLDS=2                                00000010
                    ./      DELET 00000020,00000020
SOURCE LINE DELETED         TITLE ITEM=('80-80 LIST OF DATA CARDS BY KEATON',33)      00000020
SOURCE LINE INSERTED        TITLE ITEM=('REPLACED RECORD IN PDS DICK',40)             00000020
                            RECORD FIELD=(80,,,10)                                   00000040

ABOVE NAME(DICK     )FOUND IN    NM DIRECTORY,TTR IS NOW ALTERED
                    ./      ENDUP
HIGHEST CONCODE WAS          00000000
```

```
IEF285I   PRILIB                                          PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   SYSOUT                                          SYSOUT
IEF285I   VOL SER NOS=       .
IEF285I   KEATON                                          PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   KEATON                                          PASSED
IEF285I   VOL SER NOS= WORK01.
//STPC   EXEC  PGM=IEBCOMPR
//SYSUT1 DD    DSNAME=SYS1.PROCLIB(DICK),DISP=OLD
//SYSUT2 DD    DSNAME=KEATON(DICK),UNIT=2311,VOLUME=SER=WORK01,        C
//            DISP=(OLD,DELETE)
//SYSPRINT DD  SYSOUT=A
//SYSABEND DD  SYSOUT=A
//SYSIN  DD    *
IEF236I ALLOC. FOR REKUT1   STPC
IEF237I JOBLIB   ON 192
IEF237I SYSUT1   ON 191
IEF237I SYSUT2   ON 192
IEF237I SYSIN    ON 00C
```

```
         COMPARE
         EXITS ERROR=ERRTNE
IEB2211    RECORDS ARE NOT EQUAL
    DDNAME = SYSUT1
    PHYSICAL RECORD NUMBER = 00000002  LOGICAL RECORD NUMBER WITHIN PHYSICAL RECORD = 00000001
4040404040404040E3C9E3D3C540C9E3C5D47E4D7DF8F060F8F040D3C9E2E340D6C640C4C1E3C140C3C1D9C4E240C2E840D2C5C1E3D6D57D6BF3F3
5D4040404040404040404040F0F0F0F0F0F0F2F0
    DDNAME = SYSUT2
    PHYSICAL RECORD NUMBER = 00000002  LOGICAL RECORD NUMBER WITHIN PHYSICAL RECORD = 00000001
4040404040404040E3C9E3D3C540C9E3C5D47E4D7DD9C5D7D3C1C3C5C440D9C5C3D6D9C440C9D540D7C4E240C4C9C3D27D6BF4F05D4040404040404040
4040404040404040404040F0F0F0F0F0F0F2F0
IEB225I JOB TERMINATED AFTER EXIT
    END OF JOB-TOTAL NUMBER OF RECORDS COMPARED = 00000002
```

```
IEF285I   PRILIB                                          PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   SYS1.PROCLIB                                    KEPT
IEF285I   VOL SER NOS= SYSRES.
IEF285I   KEATON                                          DELETED
IEF285I   VOL SER NOS= WORK01.
IEF285I   SYSOUT                                          SYSOUT
IEF285I   VOL SER NOS=       .
IEF285I   SYSOUT                                          SYSOUT
IEF285I   VOL SER NOS=       .
//STPC2  EXEC  PGM=IEHPROGM
//DD1    DD    UNIT=2311,VOLUME=SER=SYSRES,DISP=OLD
//SYSPRINT DD  SYSOUT=A
//SYSIN  DD    *
IEF236I ALLOC. FOR REKUT1   STPC2
IEF237I JOBLIB   ON 192
IEF237I DD1      ON 191
IEF237I SYSIN    ON 00C
```

```
        SCRATCH DSNAME=SYS1.PROCLIB,VOL=2311=SYSRES,MEMBER=DICK,PURGE
NORMAL END OF TASK RETURNED FROM SCRATCH


UTILITY END
```

```
IEF285I   PRILIB                                                PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   SYS1.PROCLIB                                          KEPT
IEF285I   VOL SER NOS= SYSRES.
IEF285I   SYSOUT                                                SYSOUT
IEF285I   VOL SER NOS=       .
//STPC3  EXEC  PGM=IEHLIST      .
//DD1    DD    UNIT=2311,VOLUME=SER=SYSRES,DISP=OLD
//SYSPRINT DD  SYSOUT=A
//SYSIN  DD    *
IEF236I ALLOC. FOR REKUT1   STPC3
IEF237I JOBLIB   ON 192
IEF237I DD1      ON 191
IEF237I SYSIN    ON 00C
```

DIRECTORY INFO FOR SPECIFIED PDS ON VOL SYSRES
SYS1.PROCLIB

MEMBERS	TTRC	VARIABLE USER DATA ---(USER DATA AND TTRC ARE IN HEX)
ADDPROCS	00000702	00063066
ASMEC	00000C02	00017033
ASMECL	00010202	00013040
ASMECLG	00011302	00013040
ASMFC	00020C02	01010059
ASMFCL	00021502	01010059
ASMFCLG	00030D02	01010059
ASM1	00040702	00062966
ASM1E	00050402	00629660
ASM1LINK	00060302	00052466
ASM3	00060700	
ASM3COGO	00070900	
COBEC	00080B02	01017005
COBECLG	00C81202	01011015
COBELG	00090B02	01011015
COLOGO	00091602	00063000
EDCTLAM2	000A0F00	
FORTEC	000A1402	00019041
FORTECLG	000A1C02	00019047
FORTELG	000B1602	00013068
LIST	000C0802	00013077
LKED	000C0C02	00016043
LKEDG	000C1402	00013068
MOD	000D0202	00013068
PL1DFC	000D0602	00018018
PL1LFC	000D0D02	00018018
PL1LFCL	000D1502	00018018
PL1LFCLG	000E0B02	00018018
PL1LG	000F0402	00018018
PUNCH	000F0D00	
SCRATCH	000F1002	00063066
SORT	000F1702	00019035
SYSLIST	00100502	00063066
TASME	00100C02	00017032
TASMEG	00110502	00015065
TASMEGED	00111902	00016043
TTED	00121A02	00015065

```
IEF285I   PRILIB                                  PASSED
IEF285I   VOL SER NOS= WORK01.
IEF285I   SYS1.PROCLIB                            KEPT
IEF285I   VOL SER NOS= SYSRES.
IEF285I   SYSOUT                                  SYSOUT
IEF285I   VOL SER NOS=       .
IEF285I   PRILIB                                  DELETED
IEF285I   VOL SER NOS= WORK01.
```

STUDENT PROBLEM - LINKAGE Editor

A program has been divided into five segments and assigned to several programmers. Each programmer has written and compiled his segment. It is now necessary to combine these segments into an executable program. It has been decided that a minimum amount of storage will be utilized for this program.

Construct an overlay program using the linkage editor and run this program.

MAIN	Read cards; if there is a B in Column 1, call PGMB; if no B, call the error routine PGMA. (Coded in assembler language)
PGMA	Error routine; prints error message and error card on console; returns to MAIN. (Coded in assembler language)
PGMB	Table Lookup Routine; uses columns 2 through 6 to find a message in a table; if message is found, call PGMB2 to print the message; if there is no message, call PRGB1 to print an error message. (Coded in COBOL)
PGMB1	Error routine; prints error message about missing entry in table; returns. (Coded in assembler language)
PGMB2	Print messages found in the table; return. (Coded in assembler language)

QUESTIONS TO ANSWER:

1. How much storage will the overlay program require?
2. How much storage would the program have taken without overlaying?
3. How much storage to the SEGTAB and ENTAB's require?

```
//*** SUPPLY JOB CONTROL AND LINK EDIT STATEMENTS TO BUILD OVERLAY (STEPA)****
//*** DATA SETS CALLED SYSUT1 (SEQ), SYSLMOD (PART.), AND SYS1.OBJMODS (PART.),
//*** HAVE BEEN CATALOGED.  SYS1.OBJMODS CONTAINS INPUT MODULES.  ************
//STEPB EXEC PGM=*.STEPA.SYSLMOD
//SYSABEND DD SYSOUT=A
//OUTPUT DD SYSOUT=A
//INPUT DD *
A1      THIS CARD SHOULD BE IN ERROR - PGMA HAS PRINTED IT ON THE CONSOLE.
B11111            CARD 1
B22222            CARD 2
B33333            CARD 3
B44444            CARD 4
A2      THIS CARD SHOULD BE IN ERROR - PGMA HAS PRINTED IT ON THE CONSOLE.
B55555            CARD 5  - THIS CARD IS AN ERROR - PGMB1 PRINTED IT.
B66666            CARD 6
/*
```

EXTERNAL SYMBOL DICTION.

SYMBOL	TYPE	ID	ADDR	LENGTH	LD ID
MAIN	SD	01	000000	0001A0	
ENTRYA	ER	02			
ENTRYB	ER	03			

RELOCATION DICTIONARY

POS.ID	REL.ID	FLAGS	ADDRESS
01	01	08	000019
01	01	08	0000ID
01	01	08	000081
01	01	08	000085
01	01	08	000101
01	02	1C	00003C
01	03	1C	000058

CROSS-REFERENCE

SYMBOL	LEN	VALUE	DEFN	REFERENCES		
CALLB	00004	OOOO4A	0033	0025		
CARDEND	00004	000068	0044	0095		
CARDIN	00004	0000E0	0074	0015	0020	0055
CON	00004	0000DC	0067	0033	0034	
IHB0005B	00004	00003C	0029	0030		
IHB0006B	00004	000058	0038	0039		
IHB0007	00001	00007A	0048	0045		
IHB0007A	00002	00007A	0049	0044		
LOOP	00004	000022	0020	0032	0041	
MAIN	00001	000000	0001			
PRINT	00004	000140	0L30	0017	0023	0057
SAVEM	00004	000094	0066	0007	0008	0059

NO STATEMENTS FLAGGED IN THIS ASSEMBLY

```
MAIN     START 0                                                        MAINP
         SAVE  (14,12)    SAVE REGISTERS                                MAINP
         BALR  12,0                                                     MAINP
         USING *,12       STANDARD ENTRY CODE                           MAINP
         ST    13,SAVEM+4                                               MAINP
         LA    11,SAVEM                                                 MAINP
         ST    11,8(0,13)                                               MAINP
         LR    13,11                                                    MAINP
         OPEN  (CARDIN,,PRINT,(OUTPUT))                                 MAINP
LOOP     GET   CARDIN             GET A CARD                            MAINP
         LA    11,PRINT                                                 MAINP
         CLI   0(1),C'B'                                                MAINP
         BE    CALLB                                                    MAINP
         CALL  ENTRYA          NOT A B CARD                             MAINP
         B     LOOP                                                     MAINP
CALLB    ST    1,CON
         LA    1,CON
         CALL  ENTRYB
         B     LOOP                                                     MAINP
CARDEND  WTO   'END OF JOB'                                             MAINP
         CLOSE (CARDIN,,PRINT)                                          MAINP
         L     13,SAVEM+4      STANDARD EXIT CODE                       MAINP
         RETURN (14,12)                                                 MAINP
*                                                                       MAINP
*   CONSTANTS AND DCB'S                                                 MAINP
*                                                                       MAINP
SAVEM    DS    18F                                                      MAINP
CON      DS    1F
CARDIN   DCB   DSORG=PS,MACRF=(GL),DDNAME=INPUT,RECFM=F,LRECL=80,      CMAINP
               BLKSIZE=80,BFTEK=S,EODAD=CARDEND,EROPT=ACC               MAINP
PRINT    DCB   DSORG=PS,MACRF=(PM),DDNAME=OUTPUT,RECFM=F,LRECL=120,    CMAINP
               BLKSIZE=120,BFTEK=S,EROPT=ACC                            MAINP
         END                                                            MAINP
         END                                                            MAINP
```

```
                                                    EXTERNAL SYMBOL DICTIONARY
 SYMBOL   TYPE ID   ADDR   LINGTH LD ID

           PC   01 000000 000079
ENTRYA     LD      000000          01



   LOC  OBJECT CODE     ADDR1 ADDR2  STM1     SOURCE STATEMENT

000000                                    1            START 0
                                          2 *       THIS IS TO BE PGMA
                                          3 *
                                          4 ENTRYA    SAVE  (14,12)
000000                                    5+ENTRYA    DS    OH
000000  90EC D00C              0000C      6+          ST,   14,12,12(13) SAVE
000004  05C0                               7          BALR  12,0
000006                                     8          USING *,12
000006  D24F C023 1000 00029 00000        9          MVC   OUTREC,0(1)
00000C  4110 C012              00018     10          LA    1,REC
                                         11          WTO   MF=(E,(1))
000010  0A23                            12+          SVC   35 ISSUE SVC
                                         13          RETURN (14,12)
000012  98EC D00C              0000C     14+          LM    14,12,12(13) REST
000016  07FE                            15+          BR    14 RETURN
000018                                   16          DS    OF
000018  0061                            17 REC       DC    H'97'
00001A  0000                            18           DC    H'0'
00001C  C5D9D9D6D940C3C1                19           DC    C'ERROR CARD******
000029                                   20 OUTREC    DS    CL80
                                         21          ENTRY ENTRYA
                                         22          END



                                                    CROSS-REFERENCE

SYMBOL     LEN   VALUE  DEFN    REFERENCES

ENTRYA    00002 000000 0005    0021
OUTREC    00080 000029 0020    0009
REC       00002 000018 0017    0010


NO STATEMENTS FLAGGED IN THIS ASSEMBLY
```

```
IDENTIFICATION DIVISION                                                 PGMB....
PROGRAM-ID. 'ENTRYB'.                                                   PGMB....
REMARKS. THIS IS A COBOL PROGRAM TO DO A TABLE LOOKUP AND PASS          PGMB....
        FINDINGS TO A SUBPROGRAM.                                       PGMB....
ENVIRONMENT DIVISION.                                                   PGMB....
CONFIGURATION SECTION.                                                  PGMB....
SOURCE-COMPUTER. IBM-360.                                               PGMB....
OBJECT-COMPUTER. IBM-360.                                               PGMB....
DATA DIVISION.                                                          PGMB....
WORKING-STORAGE SECTION.                                                PGMB....
01  TABLE-1.                                                           PGMB....
02 A1      PICTURE X(5) VALUE IS '11111'.                               PGMB....
02 F1 PICTURE A(35) VALUE 'CONGRATULATIONS --- YOU HAVE GOTTEN'.        PGMB....
02 A2  PICTURE X(5) VALUE IS '22222'.                                   PGMB....
02 F2 PICTURE A(35) VALUE '  THE OBERLAY PROGRAM TO EXECUTE TO'.        PGMB....
02 A3 PICTURE X(5) VALUE IS '33333'.                                    PGMB....
02 F3 PICTURE A(35) VALUE  '  COMPLETION IF YOU HAVE PRINTED 5 '.       PGMB....
02 A4 PICTURE X(5) VALUE IS '44444'.                                    PGMB....
02 F4 PICTURE A(35) VALUE ' CARDS ON THE PRINTER AND THREE ON'.         PGMB....
02 A5 PICTURE X(5) VALUE IS '66666'.                                    PGMB....
02 F5 PICTURE A(35) VALUE '  THE CONSOLE.                               PGMB....
01 TABLE-2 REDEFINES TABLE-1.                                           PGMB....
  02 ITEM OCCURS 5 TIMES.                                               PGMB....
  03 FINDER   PICTURE X(5).                                             PGMB....
  03 FOUND     PICTURE A(35).                                           PGMB....
  77  COUNT       PICTURE 9.                                            PGMB....
LINKAGE SECTION.                                                        PGMB....
01  PASSED-CARD.                                                       PGMB....
  02  CARD-CODE         PICTURE A.                                     PGMB....
  02  CONTROL-FIELD      PICTURE S(5).                                 PGMB....
  02  REST-OF-CARD      PICTURE X(74).                                 PGMB....
PROCEDURE DIVISION.                                                     PGMB....
     ENTER LINKAGE.                                                     PGMB....
     ENTRY 'ENTRYB' USING PASSED-CARD.                                  PGMB....
     ENTER COBOL.                                                       PGMB....
     MOVE 1 TO COUNT.                                                   PGMB....
IN. OF CONTROL-FIELD = FINDER ( COUNT ), GO TO CALLB2.                  PGMB....
     ADD L TO COUNT.                                                    PGMB....
     IF COUNT = 6 GO TO NO-FIND.                                        PGMB....
     GO TO IN.                                                          PGMB....
CALLB2.                                                                 PGMB...
     ENTER LINKAGE.                                                     PGMB....
     CALL 'ENTRYB2' USING FOUND ( COUNT ) PASSED-CARD.                  PGMB....
     ENTER COBOL.  GO TO EXIT-ROUTINE.                                  PGMB....
NO-FIND.                                                                PGMB....
     ENTER LINKAGE.                                                     PGMB....
     CALL 'ENTRYB1' USING PASSED-CARD.                                  PGMB....
     ENTER COBOL.                                                       PGMB....
EXIT-ROUTINE.                                                           PGMB....
     ENTER LINKAGE.                                                     PGMB....
     RETURN.                                                            PGMB....
     ENTER COBOL.                                                       PGMB....
```

```
                                                    EXTERNAL SYMBOL DICTIONARY
 SYMBOL   TYPE ID   ADDR   LENGTH LD ID

ENTRYB1    SD  01 000000 000086


  LOC   OBJECT CODE     ADDR1 ADDR2  STMT    SOURCE STATEMENT

000000                                    1 ENTRYB1   START 0
                                          2           SAVE  (14,12)
000000                                    3+          DS    0H
000000   90EC D00C                 0000C  4+          STM   14,12,12(13) SAVE
000004   05C0                             5           BALR  12,0
000006                                    6           USING *,12
000006 5820 1000                 00000    7           L     2,0(0,1)
00000A D249 C036 2000 0003C      00000    8           MVC   OUTPUT(74),0(2)
000010 4110 C016                 0001C    9           LA    1,OUT
                                         10           WTO   MF=(E,(1))
000014 0A23                              11+          SVC   35 ISSUE SVC
                                         12           RETURN (14,12)
000016   98EC D00C                 0000C  13+          LM    14,12,12(13) REST
00001A   07FE                            14+          BR    14 RETURN
00001C                                   15           DS    0F
00001C   006A                            16 OUT       DC    H'106'
00001E   0000                            17           DC    H'0'
000020 C3D6D5E3D9D6D340                  18           DC    C'CONTROL FIELD
00003C                                   19 OUTPUT    DS    CL74
                                         20           END



                                                    CROSS-REFERENCE

SYMBOL     LEN    VALUE    DEFN     REFERENCES

ENTRYB1   00001   000000   0001
OUT       00002   00001C   0016      0009
OUTPUT    00074   00003C   0019      0008


NO STATEMENTS FLAGGED IN THIS ASSEMBLY
```

```
                                                    EXTERNAL SYMBOL DICTIONARY
 SYMBOL   TYPE  ID   ADDR   LENGTH  LD ID

ENTRYB2    SD   01  000000  000108


 LOC   OBJECT CODE    ADDR1 ADDR2  STMT    SOURCE STATEMENT

000000                                    1 ENTRYB2   START 0
                                          2           SAVE  (14,12)
000000                                    3+          DS    OH
000000 90EC D00C                  0000C   4+          STM   14,12,12(13) SAVE REGIST
000004 05C0                               5           BALR  12,0
000006                                    6           USING *,12
000006 50D0 C046                  0004C   7           ST    13,SAVEB+4
00000A 41B0 C042                  00048   8           LA    11,SAVEB
00000E 50B0 D008                  00008   9           ST    11,8(0,13)
000012 18DB                               10          LR    13,11
000014 58B0 D004                  00004   11          L     11,4(0,13)
000018 58B0 B004                  00004   12          L     11,4(0,11)
00001C 58A0 B040                  00040   13          L     10,64(0,11)
000020 9823 1000                  00000   14          LM    2,3,0(1)
000024 D222 C08A 2000 00090 00000 15          MVC   WORK(35),0(2)
00002A D249 C0AD 3000 000B3 00000 16          MVC   WORK+35(74),0(3)
                                          17          PUT   (10),WORK
000030 181A                               18+         LR    1,10 LOAD PARAMETER REG
000032 4100 C08A                  00090   19+         LA    0.WORK LOAD PARAMETER REG
000036 58F0 1030                  00030   20+         L     15,48(0,1) LOAD PUT ROUTE
00003A 05EF                               21+         BALR  14,15 LINK TO PUT ROUTINE
00003C 58D0 C046                  0004C   22          L     13,SAVEB+4
                                          23          RETURN (14,12)
000040 98EC D00C                  0000C   24+         LM    14,12,12(13) RESTORE THE
000044 07FE                               25+         BR    14 RETURN
000048                                    26 SAVEB    DS    18F
000090                                    27 WORK     DS    CL110
0000FE 4040404040404040                   28          DC    CL10' '
                                          29          END
```

```
//BLDOVLY JOB 0,FPV,MSGLEVEL=1
//STEPA EXEC PGM=IEWL,PARM=(OBLY,XREF,LIST)
//SYSPRINT DD SYSOUT=A
//SYSLIB   DD  DSNAME=SYS1.OBJMODS,DISP=OLD
//SYSUT1 DD DSNAME=SYSUT1,DISP=OLD
//SYSLMOD DD DSNAME=SYSLMOD(RUN),DISP=OLD
//SYSLIN DD *
                  INCLUDE SYSLIB(MAIN)
                  OVERLAY POINT1
                  INCLUDE SYSLIB(PGMA)
                  OBERLAY POINT1
                  INCLUDE SYSLIB(PGMB)
                  OVERLAY POINT2
                  INCLUDE SYSLIB(PGMB1)
                  OVERLAY POINT2
                  INCLUDE SYSLIB(PGMB2)
                  ENTRY MAIN
                  NAME RUN(R)
/*
//STEPB EXEC PGM=*.STEPA.SYSLMOD
//SYSABEND DD SYSOUT=A
//OUTPUT DD SYSOUT=A
//INPUT DD *
A1      THIS CARD SHOULD BE IN ERROR - PGMA HAS PRINTED IT ON THE CONSOLE.
B11111            CARD 1
B22222            CARD 2
B33333            CARD 3
B44444            CARD 4
A2      THIS CARD SHOULD BE IN ERROR - PGMA HAS PRINTED IT ON THE CONSOLE.
B55555            CARD 5 - THIS CARD IS AN ERROR - PGMB1 PRINTED IT.
B66666            CARD 6
```

```
   LINKAGE EDITOR OPTIONS SPECIFIED OVLY,XREF,LIST
IEW0000       INCLUDE  SYSLIB(MAIN)
IEW0000       OVERLAY POINT1
IEW0000       INCLUDE  SYSLIB(PGMA)
IEW0000       OVERLAY POINT1
IEW0000       INCLUDE  SYSLIB(PGMB)
IEW0000       OVERLAY POINT2
IEW0000       INCLUDE  SYSLIB(PGMB1)
IEW0000       OVERLAY POINT2
IEW0000       INCLUDE  SYSLIB(PGMB2)
IEW0000       ENTRY MAIN
****RUN         NOW REPLACED IN DATA SET


                                                        CROSS REFERENCE TABLE


 CONTROL SECTION                                  ENTRY
    NAME     ORIGIN   LENGTY   SEG. NO.          NAME    LOCATION     NAME   LOCATION

$SEGTAB          00       2C       1
 MAIN            30      1A0       1
$ENTAB           IDO      24       1


 LOCATION  REFERS TO SYMBOL  IN CONTROL SECTION  SEG. NO.

      6C           ENTRYA                          2
      88           ENTRYB          ENTRYB          3

 CONTROL SECTION                              ENTRY

   NAME     ORIGIN   LENGTH   SEG. NO.          NAME    LOCATION     NAME   LOCATION

&PRIVATE       1F8      79       2
                                                ENTRYA       1F8

 LOCATION  REFERS TO SYMBOL  IN CONTROL SECTION  SEG. NO.


 CONTROL SECTION                              ENTRY

   NAME     ORIGIN    LENGTH   SEG. NO.          NAME    LOCATION      NAME LOCATION

 ENTRYB         1F8      3B6       3
                                                ENTRYB       4B0
$ENTAB          5B0       24       3

 LOCATION  REGERS TO SYMBOL  IN CONTROL SECTION  SEG. NO.

      470          ENTRYB2          ENTRYB2          5
```

```
 LOCATION  REFERS TO SYMBOL  IN CONTROL SECTION  SEG. NO.

    474            ENTRYB1          ENTRYB1          4

 CONTROL SECTION                          ENTRY

   NAME     ORIGIN  LENGTH  SEG. NO.       NAME   LOCATION     NAME   LOCATION

 ENTRYB1      5D8      86      4

 LOCATION  REFERS TO SYMBOL  IN CONTROL SECTION  SEG. NO.

 CONTROL SECTION                          ENTRY

   NAME     ORIGIN  LENGTH  SEG. NO.       NAME   LOCATION     NAME   LOCATION

 ENTRYB2      5D8     104      5

 LOCATION  REFERS TO SYMBOL  IN CONTROL SECTION  SEG. NO.

ENTRY ADDRESS       30
TOTAL LENGTH       6DC
```