INPUT-OUTPUT METHODS

Any batch program executed through a JCL requires data input, which is processed and an output is created. There are different methods of feeding input to the program and writing output received from a JCL. In batch mode, there is no user interaction required but input and output devices and required organisation are defined in JCL and submitted.

Data Input in a JCL

There are various ways to feed the data to a program using JCL and these methods have been explained below:

INSTREAM DATA

Instream data to a program can be specified using a SYSIN DD statement.

```
//CONCATEX JOB CLASS=6,NOTIFY=&SYSUID
//* Example 1:
//STEP10 EXEC PGM=MYPROG
//IN1    DD DSN=SAMPLE.INPUT1,DISP=SHR
//OUT1   DD DSN=SAMPLE.OUTPUT1,DISP=(,CATLG,DELETE),
//       LRECL=50,RECFM=FB
//SYSIN  DD *
//CUST1  1000
//CUST2  1001
/*
//*
//* Example 2:
//STEP20 EXEC PGM=MYPROG
//OUT1   DD DSN=SAMPLE.OUTPUT2,DISP=(,CATLG,DELETE),
//       LRECL=50,RECFM=FB
//SYSIN  DD DSN=SAMPLE.SYSIN.DATA,DISP=SHR
//*
```

In Example 1, input to MYPROG is passed through SYSIN. The data is provided within the JCL. Two records of data are passed to the program. Please note that /* marks the end of instream SYSIN data.

"CUST1 1000" is record1 and "CUST2 1001" is record2. End of data condition is met when the symbol /* is encountered while reading the data.

In Example 2, the SYSIN data is held within a dataset, where SAMPLE.SYSIN.DATA is a PS file, which can hold one or more records of data.

Data Input through files

As mentioned in most of the examples in previous chapters, data input to a program can be provided through PS, VSAM or GDG files, with relevant DSN name and DISP parameters along with DD statements.

In Example 1, SAMPLE.INPUT1 is the input file through which data is passed to MYPROG. It is referred as IN1 within the program.

Data Output in a JCL

The output in a JCL can be cataloged into a dataset or passed to the SYSOUT. As mentioned in DD statements chapter, **SYSOUT=*** redirects the output to the same class as that mentioned in the MSGCLASS parameter of the JOB statement.

Saving Job Logs

Specifying **MSGCLASS=Y** saves the job log in the JMR *JoblogManagementandRetrieval*. The entire JOB log can be redirected to the SPOOL and can be saved to a dataset by giving the XDC command against the job name in the SPOOL. When the XDC command is given in the SPOOL, a dataset creation screen is opened up. The job log can then be saved by giving appropriate PS or PDS definition.

Job logs can also be saved into a dataset by mentioning an already created dataset for SYSOUT and SYSPRINT. But the entire job log cannot be captured through this way $i.e., JESMSGwillnotbecataloged$ as done in JMR or XDC.

```
//SAMPINST JOB 1,CLASS=6,MSGCLASS=Y,NOTIFY=&SYSUID
//*
//STEP1    EXEC PGM=MYPROG
//IN1      DD DSN=MYDATA.URMI.INPUT,DISP=SHR
//OUT1     DD SYSOUT=*
//SYSOUT   DD DSN=MYDATA.URMI.SYSOUT,DISP=SHR
//SYSPRINT DD DSN=MYDATA.URMI.SYSPRINT,DISP=SHR
//SYSIN    DD MYDATA.BASE.LIB1(DATA1),DISP=SHR
//*
//STEP2    EXEC PGM=SORT
```

In the above example, SYSOUT is cataloged in MYDATA.URMI.SYSOUT and SYSPRINT in MYDATA.URMI.SYSPRINT.