JCL - BASIC SORT TRICKS

The day-to-day application requirements in a corporate world that can be achieved using Utility Programs are illustrated below:

1. A file has 100 records. The first 10 records need to be written to output file.

```
//JSTEP020 EXEC PGM=ICETOOL
//TOOLMSG  DD SYSOUT=*
//DFSMSG   DD SYSOUT=*
//IN1      DD DSN=MYDATA.URMI.STOPAFT,DISP=SHR
//OUT1    DD SYSOUT=*
//TOOLIN   DD *
  COPY FROM(IN1) TO(OUT1) USING(CTL1)
/*
//CTL1CNTL DD *
  OPTION STOPAFT=10
/*
```

The option STOPAFT will stop reading the input file after 10th record and terminates the program. Hence, 10 records are written to output.

2. Input file has one or more records for same employee number. Write unique records to output.

```
//STEP010  EXEC PGM=SORT
//SYSOUT   DD SYSOUT=*
//SORTIN   DD DSN=MYDATA.URMI.DUPIN,DISP=SHR
//SORTOUT  DD SYSOUT=*
//SYSIN    DD *
  SORT FIELDS=(1,15,ZD,A)
  SUM FIELDS=NONE
/*
```

SUM FIELDS=NONE removes duplicates on fields specified in SORT FIELDS. In the above example, employee number is in the field position 1,15. The output file will contain the unique employee numbers sorted in ascending order.

3. Overwrite input record content.

```
//JSTEP010 EXEC PGM=SORT
//SORTIN   DD DSN= MYDATA.URMI.SAMPLE.MAIN,DISP=SHR
//SORTOUT  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
 OPTION COPY
  INREC OVERLAY=(47:1,6)
/*
```

In the input file, the content in position 1,6 is overwritten to the position 47,6 and then copied to the output file. INREC OVERLAY operation is used in order to rewrite data in input file before copying to output.

4. Adding a sequence number to the output file.

```
//JSTEP010 EXEC PGM=SORT
//SORTIN   DD *
  data1
  data2
  data3
/*
//SORTOUT  DD SYSOUT=*
```

```
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
 OPTION COPY
 BUILD=(1:1,5,10:SEQNUM,4,ZD,START=1000,INCR=2)
/*
```

The output will be:

```
data1    1000
data2    1002
data3    1004
```

4-digit sequence number is added in output at position 10, starting at 1000 and incremented by 2 for every record.

5. Adding Header/Trailer to output file.

```
//JSTEP010 EXEC PGM=SORT
//SORTIN   DD *
  data1
  data2
  data3
/*
//SORTOUT  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
 SORT FIELDS=COPY
  OUTFIL REMOVECC,
  HEADER1=(1:C'HDR',10:X'020110131C'),
  TRAILER1=(1:C'TRL',TOT=(10,9,PD,TO=PD,LENGTH=9))
/*
```

The output will be:

```
HDR       20110131
data1
data2
data3
TRL       000000003
```

TOT calculates the number of records in the input file. HDR and TRL are added as identifiers to header/trailer, which is user defined and can be customised as per the users' needs.

6. Conditional Processing

```
//JSTEP010 EXEC PGM=SORT
//SORTIN   DD *
  data1select
  data2
  data3select
/*
//SORTOUT  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
  INREC  IFTHEN=(WHEN=(6,1,CH,NE,C' '),BUILD=(1:1,15),
         IFTHEN=(WHEN=(6,1,CH,EQ,C' '),BUILD=(1:1,5,7:C'EMPTY    ')
  OPTION COPY
/*
```

The output will be:

```
data1select
data2 EMPTY
```

```
data3select
```

Based on the 6th position of the file, the BUILD of output file varies. If 6th position is SPACES, then text "EMPTY" is appended to input record. Else, the input record is written to output, as-is.

7. Backing up a file

```
//JSTEP001 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
//SYSOUT   DD SYSOUT=*
//SORTOUT  DD DUMMY
//SYSUT1   DD DSN=MYDATA.URMI.ORIG,DISP=SHR
//SYSUT2   DD DSN=MYDATA.URMI.BACKUP,DISP=(NEW,CATLG,DELETE),
//             DCB=*.SYSUT1,SPACE=(CYL,(50,1),RLSE)
```

IEBGENER copies the file in SYSUT1 to file in SYSUT2. Please note that file in SYSUT2 takes the same DCB as that of the SYSUT1 in the above example.

8. File Comparison

```
//STEP010  EXEC PGM=SORT
//MAIN     DD *
  1000
  1001
  1003
  1005
//LOOKUP   DD *
  1000
  1002
  1003
//MATCH    DD DSN=MYDATA.URMI.SAMPLE.MATCH,DISP=OLD
//NOMATCH1 DD DSN=MYDATA.URMI.SAMPLE.NOMATCH1,DISP=OLD
//NOMATCH2 DD DSN=MYDATA.URMI.SAMPLE.NOMATCH2,DISP=OLD
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
  JOINKEYS F1=MAIN,FIELDS=(1,4,A)
  JOINKEYS F2=LOOKUP,FIELDS=(1,4,A)
  JOIN UNPAIRED,F1,F2
  REFORMAT FIELDS=(?,F1:1,4,F2:1,4)
  OPTION COPY
  OUTFIL FNAMES=MATCH,INCLUDE=(1,1,CH,EQ,C'B'),BUILD=(1:2,4)
  OUTFIL FNAMES=NOMATCH1,INCLUDE=(1,1,CH,EQ,C'1'),BUILD=(1:2,4)
  OUTFIL FNAMES=NOMATCH2,INCLUDE=(1,1,CH,EQ,C'2'),BUILD=(1:2,4)
/*
```

- JOINKEYS specifies the field on which the two files are compared.
- REFORMAT FIELDS=? places 'B' *matchedrecords*, '1' *presentinfile1*, *butnotinfile2*, or '2' *presentinfile2butnotinfile1* in the 1st position of the output BUILD.
- JOIN UNPAIRED does a full outer join on the two files.

The output will be:

```
MATCH File
1000
1003

NOMATCH1 File
1001
1005

NOMATCH2 File
1002
```

The same functionality can be achieved using ICETOOL also.