Section H : Sorting Procedures in PL/I

H.1 LINEAR SELECTION

```
LINSEL: PROCEDURE(TOSORT,SORTED,NUMBER);
DECLARE
TOSORT(*) FIXED BINARY(15,0),
SORTED(t)  FIXED  BINARY(15,0),
NUMBER FIXED BINARY(15,0),
SOURCE FIXED BINARY(15,0),
I FIXED BINARY(15,0),
J FIXED BINARY(15,0),
LOW FIXED BINARY(15,0);
GROW:  DO J=1 TO NUMBER BY 1;
       LOW=TOSORT(1);
       SOURCE=1;
       SELECT: DO I=2 TO-NUMBER BY 1;
       IF LOW>TOSORT(I) THEN DO
       LOW=TOSORT(I);
       SOURCE=I;
                              WD;
       ELSE;  END SELECT;
       PLACE: SORTED(J)=LOW;
       TOSORT(SOURCE)=99;
 END  GROW;
 END  LINSEL;
```

H.2 BASIC EXCHANGE

```
BASICEXC:PROCEDURE(TOSORT,NUMBER);
DECLARE
TOSORT(*) FIXED BINARY(15,0),
NUMBER FIXED BINARY(15,0),
EXCOUNT FIXED BINARY(15,0),
ADJUST FIXED BINARY(15,0),
ELIMIT FIXED BINARY(15,0),
OLIMIT FIXED BINARY(15,0),
ODDEVE FIXED BINARY(15,0),
LIMIT FIXED BINARY(15,0),
TEMP FIXED BINARY(15,0),
PASSW FIXED BINARY(1,0),
I FIXED BINARY(15,0),
   ADJUST=2*TRUNC(NUMBER/2);
   IF ADJUST<NUMBER THEN DO;
   ELIMIT=ADJUST;
   OLIMIT=ADJUST-1;
   END;
   ELSE DO;
   ELIMIT=NUMBER-2;
   OLIMIT=NUMBER-1;
   END;
   ODD: PASSW=1;
   LIMIT=OLIMIT;
   ODDEVE=1;
PASS: EXCOUNT=0;
   DO I=ODDEVE TO LIMIT BY 2;
   IF TOSORT(I)>TOSORT(I+1) THEN DO;
   TEMP=TOSORT(I);
   TOSORT(I)=TOSORT(I+1);
   TOSORT(I+1)=TEMP;
   EXCOUNT=1;
   END;
   END;
   IF EXCOUNT=0 THEN GO TO EXIT:
   IF PASSW=1 THEN DO;
   PASSW=0;
   ODDEVE=2;
   LIMIT=ELIMIT;
   GO TO PASS;
   END;
   ELSE GO TO ODD;
   EXIT: END BASICEXC:
```

H.3 STANDARD EXCHANGE

```
STEXCH:PROCEDURE(TOSORT,NUMBER);
DECLARE
TOSORT(*) FIXED BINARY(15,0),
NUMBER FIXED BINARY(15,0),
EXCOUNT FIXED BINARY(15,0),
TEMP FIXED BINARY(15,0),
I FIXED BINARY(15,0),
PASS: EXCOUNT=0;
PASSLOOP: DO I=1 TO NUMBER-1 BY 1;
IF TOSORT(I)>TOSORT(I+1) THEN DO;
 TEMP=TOSORT(I);
TOSORT(I)=TOSORT(I+1);
TOSORT(I+1)=TEMP;
EXCOUNT=1;
END;
END PASSLOOP;
 IF EXCOUNT=0 THEN GO TO EXIT;
GO TO PASS;
EXIT: END STEXCH;
```

H.4 SIMPLE INSERTION

```
SIMSERT: PROCEDURE (TOSORT,NEW);
DECLARE
TOSORT(*) FIXED BINARY(15,0),
NEW FIXED BINARY(15,0),
I FIXED BINARY(15,0),
J FIXED BINARY(15,0),
DO I=1 TO LENGTH BY 1;
IF NEW<=TOSORT(I) THEN DO;
DO J=LENGTH TO I BY -1;
TOSORT(J+1)=TOSORT(J);
END;
GO TO SETIN;
END;
END:
SETIN: TOSORT(I)=NEW;
END SIMSERT;
END FEED;
```

H.5 SIMPLE SIFTING (SHUTTLESORT)

```
SHUTTLESORT: PROCEDURE (TOSORT,NUMBER);
/* PUBLISHED IN ALGOL AS ALGORITHM 175, COMMUNICATIONS OF ACM,
  VOL 6, NO 6, P 312 */
DECLARE
TOSORT(*) FIXED BINARY(15,0),
NUMBER FIXED BINARY(15,0),
TEMP FIXED BINARY(15,0),
I FIXED BINARY(15,0),
J FIXED BINARY(15,0);
PASS: DO I=1 TO NUMBER-1 by 1;
EXCH: DO J=I TO 1 BY -1;
IF TOSORT(J)<=TOSORT(J+1) THEN GO TO EXIT;
TEMP=TOSORT(J);
TOSORT(J)=TOSORT(J+1)
TOSORT(J+1)=TEMP;
END EXCH;
EXIT: END PASS;
END SHUTTLESORT;
```

H.6 SHELLSORT

```
BSHELLSORT:PROCEDURE(TOSORT,NUMBER);
/* ALGORITHM 201, SHELLSORT, PUBLISHED IN ALGOL PUBLICATION
  LANGUAGE, COMMUNICATIONS OF ACM, VOL 6, NO. 8, AUGUST 1963 */
DECLARE
TOSORT(*) FIXED BINARY (31,0),
DISTANCE FIXED BINARY (31,0),
LIMIT FIXED BINARY (31,0),
TEMP FIXED BINARY (31,0),
I FIXED BINARY (31,0),
J FIXED BINARY (31,0),
LOGNMBR FIXED BINARY (31,0),
NUMBER FIXED BINARY (31,0),
LOGNMBR=LOG2(NUMBER);
DISTANCE=2**LOGNMBR-1;
DIST: DO WHILE (DISTANCE>0);
LIMIT=NUMBER-DISTANCE;
SETS: DO J=1 TO LIMIT BY 1;
ELTS: DO I=J TO 1 by -DISTANCE;
IF TOSORT(I+DISTANCE)>=TOSORT(I) THEN GO TO OUT;
TEMP=TOSORT(I);
TOSORT(I)=TOSORT(I+DISTANCE);
TOSORT(I+DISTANCE)=TEMP;
END ELTS;
OUT: END SETS;
DISTANCE=DISTANCE/2;
END DIST;
END BSHELLSORT
```

H.7 TREESORT

```
TREESORT3: PROCEDURE(TOSORT,NUMBER);
/* ALGORITHM 245, PUBLISHED IN COMMUNICATIONS ACM, VOL 7, NO 12,
  P 701 */
DECLARE
TOSORT(*) FIXED BINARY (31,0),
ANCESTOR FIXED BINARY (31,0),
TEMP FIXED BINARY (31,0),
I FIXED BINARY (31,0),
LIMNODE FIXED BINARY (31,0),
FATHER FIXED BINARY (31,0),
NUMBER FIXED BINARY (31,0),

DO I=TRUNC(NUMBER/2) TO 2 BY -1;
CALL UPTREE;
END;
I=1;
DO LIMNODE=NUMBER TO 2 BY -1;
CALL UPTREE;
TEMP=TOSORT(1);
TOSORT(1)=TOSORT(LIMNODE);
TOSORT(LIMNODE)=TEMP;
END;


UPTREE: PROCEDURE;
ANCESTOR=I;
TEMP=TOSORT (ANCESTOR);
INLOOP: FATHER=2*ANCESTOR;
IF FATHER>LIMNODE THEN GO TO SETIN;
IF FATHER=LIMNODE THEN GO TO TEMPTEST;
IF TOSORT(FATHER+1)>TOSORT(FATHER) THEN FATHER=FATHER+1;
TEMPTEST: IF TOSORT(FATHER)>TEMP THEN DO;
TOSORT(ANCESTOR)=TOSORT(FATHER);
ANCESTOR=FATHER;
GO TO INLOOP;
END;
SETIN: TOSORT(ANCESTOR)=TEMP;
END UPTREE;
END TREESORT3
```

H.8 QUICKERSORT

```
QUICKERSORT: PROCEDURE(TOSORT,NUMBER);
/* ALGORITHM 271, COMMUNICATIONS ACM, VOL 8, NO 11, p 669 */;
DECLARE
TOSORT(*) FIXED BINARY (31),
NUMBER FIXED BINARY (31,0),
ORIGIN FIXED BINARY (31,0),
LOWLIM FIXED BINARY (31,0),
HIGHLIM FIXED BINARY (31,0),
PARTIND FIXED BINARY (31,0),
PIVOT FIXED BINARY (31,0),
TEMP FIXED BINARY (31,0),
EXCH FIXED BINARY (31,0);
LIMIT=20;
SORT: BEGIN;
DECLARE
PARTTABLOW(LIMIT) FIXED BINARY (31,0),
PARTTABHIGH(LIMIT) FIXED BINARY (31,0);

ORIGIN=1;
PARTIND=1;
TESTSIZE: IF NUMBER-ORIGIN>1 THEN DO;
PIVOT=TRUNC ((NUMBER+ORIGIN)/2);
TEMP=TOSORT(PIVOT);
TOSORT(PIVOT)=TOSORT(ORIGIN);
HIGHLIM=NUMBER;
FINDHIGH: DO LOWLIM=ORIGIN+1 TO HIGHLIM BY 1;
IF TOSORT(LOWLIM)>TEMP THEN DO;
FINDLOW: DO HIGHLIM=HIGHLIM TO LOWLIM BY -1;
IF TOSORT(HIGHLIM)<TEMP THEN DO;
EXCH=TOSORT(LOWLIM);
TOSORT(LOWLIM)=TOSORT(HIGHLIM);
TOSORT(HIGHLIM)=EXCH;
HIGHLIM=HIGHLIM-1;
GO TO ENDHIGH:
END;
END FINDLOW;
HIGHLIM=LOWLIM-1;
GO TO LIMSMET;
END;
ENDHIGH: END FINDHIGH;
```

```
LIMSMET: TOSORT(ORIGIN)=TOSORT(HIGHLIM);
TOSORT(HIGHLIM)=TEMP;
IF 2*HIGHLIM>ORIGIN+NUMBER THEN DO;
PARTTABLOW(PARTIND)=ORIGIN;
PARTTABHIGH(PARTIND)=HIGHLIM-1;
ORIGIN=HIGHLIM+1;
END;
ELSE DO;
PARTTABLOW(PARTIND)=HIGHLIM+1;
PARTTABHIGH(PARTIND)=NUMBER;
NUMBER=HIGHLIM-1;
END;
PARTIND=PARTIND+1;
GO TO TESTSIZE;
END:
IF ORIGIN=NUMBER THEN GO TO SETPART;
IF TOSORT(ORIGIN)>TOSORT(NUMBER) THEN DO;
EXCH=TOSORT(ORIGIN)
TOSORT(ORIGIN)=TOSORT(NUMBER);
TOSORT(NUMBER)=EXCH;
END;
SETPART: PARTIND=PARTIND-1;
IF PARTIND>0 THEN DO;
ORIGIN=PARTTABLOW(PARTIND);
NUMBER=PARTTABHIGH(PARTIND);
GO TO TESTSIZE;
END;
END SORT;
END QUICKERSORT;
```

H.9 SINGSORT

```
SINGSORT: PROCEDURE(TOSORT,NUMBER);
/* ALGORITHM 347, COMMUNICATIONS OF ACM, VOL 12, NO 3, P 185 */
DECLARE
TOSORT(*) FIXED BINARY (31,0),
PIVOT FIXED BINARY (31,0),
TEMP2 FIXED BINARY (31,0),
LIMDEX FIXED BINARY (31,0),
INITIAL FIXED BINARY (31,0),
MEDIAN FIXED BINARY (31,0),
BOTIND FIXED BINARY (31,0),
TOPIND FIXED BINARY (31,0),
LIMITS FIXED BINARY (31,0),
I FIXED BINARY (31,0),
NUMBER FIXED BINARY (31,0),
PARTOP FIXED BINARY (31,0) INITIAL (1);

LIMITS=20;
SORT: BEGIN;
DECLARE
TOPS(LIMITS) FIXED BINARY (31,0),
BOTTOMS(LIMITS) FIXED BINARY (31,0),

LIMDEX=1;
INITIAL=PARTOP;
GO TO SINKTEST;
SPLIT: MEDIAN=TRUNC(PARTOP+NUMBER)/2);
PIVOT=TOSORT(MEDIAN);
TOPIND=PARTOP;
BOTIND=NUMBER;
IF TOSORT(PARTOP)>PIVOT THEN DO;
TOSORT(MEDIAN)=TOSORT(PARTOP);
TOSORT(PARTOP)=PIVOT;
PIVOT=TOSORT(MEDIAN);
END;
IF TOSORT(NUMBER)<PIVOT THEN DO;
TOSORT(MEDIAN)=TOSORT(NUMBER);
TOSORT(NUMBER)=PIVOT;
PIVOT=TOSORT(MEDIAN);
IF TOSORT(PARTOP)>PIVOT THEN DO;
TOSORT(MEDIAN)=TOSORT(PARTOP);
TOSORT(PARTOP)=PIVOT;
PIVOT=TOSORT(MEDIAN);
END;
END;
```

```
FINDSMALL: BOTIND=BOTIND-1;
IF TOSORT(BOTIND)>PIVOT THEN GO TO FINDSMALL;
TEMP2=TOSORT(BOTIND);
FINDLARGE: TOPIND=TOPIND+1;
IF TOSORT(TOPIND)<PIVOT THEN GO TO FINDLARGE;
IF TOPIND<=BOTIND THEN DO;
TOSORT(BOTIND)=TOSORT(TOPIND);
TOSORT(TOPIND)=TEMP2;
GO TO FINDSMALL;
END;
IF BOTIND-PARTOP<NUMBER-TOPIND THEN DO;
TOPS(LIMDEX)=PARTOP;
BOTTOMS(LIMDEX)=BOTIND;
PARTOP=TOPIND;
END;
ELSE DO;
TOPS(LIMDEX)=TOPIND;
BOTTOMS(LIMDEX)=NUMBER;
NUMBER=BOTIND;
END;
LIMDEX=LIMDEX+1;
SINKTEST: IF NUMBER-PARTOP>10 THEN GO TO SPLIT;
IF INITIAL=PARTOP THEN DO;
IF PARTOP<NUMBER THEN GO TO SPLIT;
END;
DO I=PARTOP+1 TO NUMBER BY 1;
PIVOT=TOSORT(I);
TOPIND=I-1;
IF TOSORT (TOPIND)>PIVOT THEN DO;
SINK: TOSORT(TOPIND+1)=TOSORT(TOPIND);
TOPIND=TOPIND-1;
IF TOSORT(TOPIND)>PIVOT THEN GO TO SINK;
TOSORT(TOPIND+1)=PIVOT;
END;
END;
LIMDEX=LIMDEX-1;
IF LIMDEX>=1 THEN DO;
PARTOP=TOPS(LIMDEX);
NUMBER=BOTTOMS(LIMDEX);
GO TO SINKTEST;
END;
END SORT;

END SINGSORT;
END FEED;
```

H.10 STRINGSORT

```
STRINGSORT: PROCEDURE(TOSORT,NUMBER);
/* ALGORITHM 207, COMMUNICATIONS ACM, VOL 5, NO 10, P 215 */
DECLARE
TOSORT(*) FIXED BINARY (31,0),
NUMBER FIXED BINARY (31,0),

SORT: BEGIN;
DECLARE
WORK(2*NUMBER) FIXED BINARY (31,0),
TOPST FIXED BINARY (31,0),
BOTST FIXED BINARY (31,0),
LIMITS (2) FIXED BINARY (31,0),
ADVANCE FIXED BINARY (31,0),
NEXT FIXED BINARY (31,0),
LAST FIXED BINARY (31,0),
K FIXED BINARY (31,0),
PASSW FIXED BINARY (1,0),
EXTEND LABEL;
INITIAL: DO I=1 TO NUMBER BY 1;
WORK(I)=TOSORT(I);
END INITIAL;
ODDPASS: TOPST=1;
BOTST=NUMBER;
LIMITS(1)=NUMBER+1;
LIMITS(2)=2*NUMBER;
K=1;
ADVANCE=1;
PASSW=1;
FIRSTST: EXTEND=NONDOWN;
NEXT=LIMITS(K);
IF WORK(TOPST)>=WORK(BOTST) THEN GO TO BOTTOM;
ELSE GO TO TOP;
TOP: WORK(NEXT)=WORK(TOPST);
TOPST=TOPST+1;
GO TO NEWNEXT;
BOTTOM: WORK(NEXT)=WORK(BOTST);
BOTST=BOTST-1;
```

```
NEWNEXT: LAST=NEXT;
NEXT=NEXT+ADVANCE;
IF BOTST>=TOPST THEN GO TO EXTEND;
IF PASSW=0 THEN IF NEXT=NUMBER+1 THEN GO TO EXIT;
ELSE GO TO ODDPASS;
ELSE IF NEXT=2*NUMBER+1 THEN GO TO EXIT;
ELSE GO TO EVENPASS;
JDOWN: IF WORK(TOPST)>=WORK(LAST) THEN GO TO TOP;
ELSE GO TO BOTHDOWN;
IDOWN: IF WORK(BOTST)>=WORK(LAST) THEN GO TO BOTTOM;
ELSE GO TO BOTHDOWN;
NONDOWN: IF WORK(TOPST)>=WORK(LAST) THEN
IF WORK(BOTST)>=WORK(LAST) THEN
IF WORK(BOTST)>=WORK(TOPST) THEN GO TO TOP;
ELSE GO TO BOTTOM;
ELSE DO;
EXTEND=JDOWN;
GO TO TOP;
END;
ELSE DO;
EXTEND=IDOWN;
GO TO IDOWN;
END;
BOTHDOWN:
LIMITS(K)=NEXT;
IF K=1 THEN K=2;
ELSE K=1;
ADVANCE= -ADVANCE;
GO TO FIRSTST;
EVENPASS: TOPST=NUMBER+1;
BOTST=2*NUMBER;
LIMITS(1)=1;
LIMITS(2)=NUMBER;
ADVANCE=1;
K=1;
PASSW=0;
GO TO FIRSTST;
EXIT: DO I=1 TO NUMBER BY 1;
TOSORT(I)=WORK(LIMITS (1) -1+I);
END;
END SORT;
END STRINGSORT;
```

H.11 DIGIT COUNTING (MATHSORT)

```
RMATHSORT: PROCEDURE(TOSORT,SORTED,NUMBER,RANGE);
/* ALGORITHM 23 PUBLISHED IN CACM, VOL 3, NO 11, NOV, 1960.
THIS VERSION DERIVED FROM MODIFICATION OF RANSHAW. CACM VOL 4 NO
  5, MAY 1961. CODING HERE ASSUMES KEY DEVELOPMENT FUNCTION TO BE
  NULL. THAT IS, KEYS PRESENTED IN TOSORT CAN BE USED DIRECTLY AS
  THEY FALL WITHIN RANGE. */
DECLARE
TOSORT(*) FIXED BINARY (15,0),
SORTED(*) FIXED BINARY (15,0),
NUMBER FIXED BINARY (15,0),
RANGE FIXED BINARY (15,0),
I FIXED BINARY (15,0),
J FIXED BINARY (15,0),
K FIXED BINARY (15,0),
SORT: BEGIN;
DECLARE
TOTALS (RANGE) FIXED BINARY INITIAL (RANGE)0);
COUNT: DO I=1 to NUMBER BY 1;
TOTALS(TOSORT(I))=TOTALS(TOSORT(I))+1;
END COUNT;
/* COUNT DEVELOPS IN TOSORT THE NUMBER OF OCCURRENCES OF ALL KEYS
  IN TOSORT */
CUMULATE: DO J=RANGE-1 TO 1 BY -1;
TOTALS(J)=TOTALS(J)+TOTALS(J+1);
END CUMULATE;
/* CUMULATE DEVELOPS THE DESCENDING CUMULATIVE COUNT
OF KEYS IN TOSORT GREATER THAN J */
PLACE: DO K=1 TO NUMBER BY 1;
SORTED(NUMBER+1-TOTALS(TOSORT(K)))=TOSORT(K);
TOTALS(TOSORT(K))=TOTALS(TOSORT(K))-1;
END PLACE;
END SORT;
END RMATHSORT;
```