C FORTRAN PROGRAM TO FIND MEAN OF N NUMBERS AND NUMBER OF VALUES GREATER THAN THE MEAN
DIMENSION A(99) ← array with fixed-size bounds
REAL MEAN ← variables beginning A-N are implicitly int.
READ(5),N
FORMAT(12)
READ(1,10)(A(I),I=1,N)
10 FORMAT(6F10.5)
SUM=0.0
DO 15 I=1,N
  SUM=SUM+A(I)
  MEAN=SUM/FLOAT(N)
  NUMBER=0
DO 20 I=1,N
  IF (A(I).LE. MEAN) GOTO 20
  NUMBER=NUMBER+1
20 CONTINUE
WRITE(2,25) MEAN,NUMBER
25 FORMAT(8H MEAN = ,F10.5,5X,20H NUMBER OVER MEAN = ,15)
STOP
END

begin comment this program is the ALGOL 60 version of finding the mean and the number of those greater than the mean;
integer n;
read(n);
begi real array a[1:n]; variable bounds for array
begin integer i, number, real sum, mean; implicit variable declarations, must be explicit
for i:= 1 step 1 until n do read (a[i]);
sum:= 0.0;
for i:= 1 step 1 until n do
  sum:= sum + a[i];
  mean:= sum/n;
  number:= 0;
for i:= 1 step 1 until n do
  if a[i] > mean then number:= number + 1;
  end
  write("MEAN =", mean, "NUMBER OVER MEAN =", number)
end

main()
/* this is the C version of the program to find the mean and the number of those greater than the mean */

{ float a[100], mean, sum;
/*the array a has 100 elements - a[0],.. a[99] */
int n, i, number;
scanf("%d", &n);
for(i = 0; i < n; i++) scanf("%f", &a[i]);
sum = 0.0;
for(i = 0; i < n; i++)
  sum += a[i];
mean = sum / n;
number = 0;
for(i = 0; i < n; i++)
  { if (a[i] > mean) number++;
  }
printf("MEAN = %.5f", mean);
printf("NUMBER OVER MEAN = %d", number);
}
1. IDENTIFICATION DIVISION.

PROGRAM-ID. INOUT.
*Comments are placed on a line which has an asterisk
*in the first column. It is a very good idea to put
*your NAME and the date of writing on your program.
*You should also include a brief description of the
*function of the program.
*This program reads in one file, extends each record,
*and writes out a new file.
*
2. ENVIRONMENT DIVISION.
   INPUT-OUTPUT SECTION.
   FILE-CONTROL.

       SELECT INP-FIL ASSIGN TO INFILE.
       SELECT OUT-FIL ASSIGN TO OUTFILE.

3. DATA DIVISION.
   FILE SECTION.

       FD          INP-FIL
                   LABEL RECORDS STANDARD
                   DATA RECORD IS REC-IN.
       01          REC-IN.
                   05 ALPHA-IN PIC A(4).
                   05 SP-CH-IN PIC X(4).
                   05 NUM-IN  PIC 9(4).
       FD          OUT-FIL
                   LABEL RECORDS STANDARD
                   DATA RECORD IS REC-OUT.
       01          REC-OUT.
                   05 ALPHA-OUT PIC A(4).
                   05 SP-CH-OUT PIC X(4).
                   05 NUM-OUT  PIC 9(4).
                   05 EXTRAS   PIC X(16).

   WORKING-STORAGE SECTION.

       01      EOF PIC X VALUE IS 'N'.

4. PROCEDURE DIVISION.

   AA.

       OPEN INPUT INP-FIL
       OPEN OUTPUT OUT-FIL

       PERFORM CC
       PERFORM BB THRU CC UNTIL EOF = 'Y'

       CLOSE INP-FIL, OUT-FIL
       DISPLAY "End of Run"
       STOP RUN.

   BB.

       MOVE REC-IN TO REC-OUT
       MOVE 'EXTRA CHARACTERS' TO EXTRAS
       WRITE REC-OUT.

   CC.

       READ INP-FIL
       AT END MOVE 'Y' TO EOF.

************** END OF LISTING **************
EXAMPLE: PROCEDURE OPTIONS (MAIN):
/* This is the PL/I version of the mean and the number of values greater than the mean */
GET LIST (N);
IF N > 0 THEN BEGIN;
DECLARE MEAN, A(N) DECIMAL FLOAT.
    SUM DEC FLOAT INITIAL(0), NUMBER FIXED INITIAL (0);
GET LIST (A);
    DO I = 1 TO N;
        SUM = SUM + A(I);
    END;
    MEAN = SUM/N;
    DO I = 1 TO N;
        IF A(I) > MEAN THEN
            NUMBER = NUMBER + 1;
    END: terminates inner & outer block
PUT LIST ('MEAN=', MEAN,
    'NUMBER GREATER THAN MEAN=', NUMBER);
END EXAMPLE:

10 REM THIS IS A BASIC PROGRAM FOR FINDING THE MEAN
20 DIM A(99)
30 INPUT N
40 FOR I = 1 TO N
50 INPUT A(I)
60 LET S = S + A(I)
70 NEXT I
80 LET M = S/N
90 LET K = 0
100 FOR I = 1 TO N
110 IF A(I) < M THEN 130
120 LET K = K + 1
130 NEXT I
140 PRINT "MEAN IS", MEAN
150 PRINT "NUMBER GREATER THAN MEAN IS", K
160 STOP
170 END
Functional programming example in a functional subset of Ada.
Finds difference between largest and smallest of 3 integers.

```ada
function max(a, b : integer) return integer is
begin
  if a > b then
    return a;
  else
    return b;
  end if;
end max;

function min(a, b : integer) return integer is
begin
  if a < b then
    return a;
  else
    return b;
  end if;
end min;

function difference(a, b, c : integer) return integer is
begin
  return max(a, max(b, c)) - min(a, min(b, c));
end difference;

with a possible call being:

put(difference(10, 4, 7));

result: 6
```
parent( pam, bob).
parent( tom, bob).
parent( tom, liz).
parent( bob, ann).
parent( bob, pat).
parent( pat, jim).
female( pam).
males( tom).
males( bob).
females( liz).
females( ann).
females( pat).
males( jim).

offspring( Y, X) :-
parent( X, Y).

mother( X, Y) :-
parent( X, Y),
females( X).

grandparent( X, Z) :-
parent( X, Y),
parent( Y, Z).

sister( X, Y) :-
parent( Z, X),
parent( Z, Y),
females( X),
different( X, Y).

predecessor( X, Z) :-
parent( X, Y).

predecessor( X, Z) :-
parent( Y, X),
predecessor( Y, Z).

% Pam is a parent of Bob
% Pam is female
% Tom is male
% Y is an offspring of X if
% X is a parent of Y
% X is the mother of Y if
% X is a parent of Y and
% X is female
% X is a grandparent of Z if
% X is a parent of Y and
% Y is a parent of Z
% X is a sister of Y if
% X and Y have the same parent and
% X is female and
% X and Y are different
% Rule pr1: X is a predecessor of Z
% Rule pr2: X is a predecessor of Z

Figure 1.8 The family program.

(a) ?- parent( pam, bob).
(b) ?- mother( pam, bob).
(c) ?- grandparent( pam, ann).
(d) ?- grandparent( bob, jim). ← Backtracking here

Bratko, Prolog Programming