state of the art of computing, and one of the reasons for this book.)

6.4 Now that you are alert to the perils of testing floating point numbers for equality, try fixing

```
C     FIRST ATTEMPT FOR APPROXIMATING AREA UNDER A CURVE
      1 AREA=0.0
        READ(2,10)T
     10 FORMAT(F10.4)
        H=0.1
        X=0.0
      2 XN=-X
        AREA=AREA+(6.0*(2.0**XN)+6.0*(2.0**(XN-H)))*0.1/2.0
        X=X+H
        IF(X-T)2,8,9
      8 WRITE(3,33)AREA
     33 FORMAT('AREA =',F8.5)
        GO TO 1
      9 CALL EXIT
        END
```

Do you think just changing the `IF` to

```
        IF (X-T) 2, 8, 8
```

is sufficient?

6.5 "Defensive programming" means anticipating problems in advance, and coding to avoid errors before they arise. What could you do to the following program fragments in the way of defensive programming?

(a) This is the entire body of a procedure for computing the arcsine of `X` in degrees:

```
        IF X = 1 THEN RETURN(90);
        ELSE RETURN(ATAND(X/SQRT(1-X**2)));
```

(b) This function finds the minimum element in an array `A` of `N` items.

```
        FUNCTION SMALL(A,N)
        DIMENSION A(1)
        SMALL = A(1)
        DO 1 K = 2,N
        IF(A(K) - SMALL)2,1,1
2       SMALL = A(K)
1       CONTINUE
        RETURN
        END
```

(c) This is a PL/I table-search routine.

```
        I = 1;
        DO WHILE( I <= N & KEY ¬= TABLE.KEY(I) );
             I = I + 1;
        END;
        IF I <= N
             THEN DATA = TABLE.DATA(I);
             ELSE DATA = '';
```

(Hint: In what order are compound logical expressions evaluated by your local PL/I