Computer Science 117 Summer 2002, Session III
Practice Test for August 5, 2002

Read the last part of the test before working any of the problems.

1. The *root mean square* (abbreviated RMS) of a list of numbers is defined to be the square root of the average of the squares of the numbers. (The RMS concept has applications in electrical engineering, such as in the phrase, "RMS voltage.") For example, the *root mean square* of (2,4) is $\sqrt{10} \approx 3.16$, since the average of 2^2 and 4^2 is 10.

   Write a Fortran 90 subroutine that has three arguments. When the subroutine is called, the third argument is set equal to the root mean square of the first and second arguments. Thus, for example, execution of the statement `CALL RMS(2,4,X)` would cause the value of the variable `X` to become $\sqrt{10}$. Your subroutine code should make proper use of IN, INOUT, or OUT, as needed.

   If written in the way I think best, your code should be six lines long.

2. The following Fortran 90 function has a few errors in it, but you should be able to tell what its purpose is. Rewrite the function, correcting errors, and using fewer lines by thinking of LOGICAL as a data type.

```
function is_perfect_square(n)
integer::n
if n < 0 then
 .false.
else if nint(sqrt(n))**2 .eqv. n then
 .true.
else
 .false.
end if
end function is_perfect_square
```

3. If n is a non-negative integer, n *factorial* (usually written as $n!$) is defined to be the product of all the positive integers up through n. Thus, $1! = 1$, $2! = 1 \cdot 2 = 2$, $3! = 1 \cdot 2 \cdot 3 = 6$, and $7! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6 \cdot 7 = 5040$. We define $0! = 1$. A programmer was attempting to write a recursive Fortran 90 function to compute factorial. Here is that programmer's code. The code contains several errors. Fix them. (Either mark corrections or rewrite the code completely.)

```
 function factorial(n)::integer
  intention in::n
  if n < 0 then
   print 10 "Error in function factorial: negative argument not allowed"
10 format(a10)
  else if n = 0 then
   factorial = 1
  else
   factorial = n * factorial(n-1)
  end if
 end function factorial
```

Now, let us incorporate your answers into a program.

Suppose that all three of the above subprograms are inserted into the program below in the correct place:

```
program t03
 implicit none
 real::x
 integer::i
 call rms(2.0,4.0,x)
 print*,"The rms is",x
 do i = -20,20
  if (is_perfect_square(i)) print*,i,"is a perfect square"
 end do
 print*,"factoria(7) = ",factorial(7)
contains
! Insert subprograms here
end program t03
```

The output of the program, if the subprograms are written correctly, should be:

```
The rms is 3.1622777
0 is a perfect square
1 is a perfect square
4 is a perfect square
9 is a perfect square
16 is a perfect square
factoria(7) =  5040
```