Fortran and Python Homework 2

Remember to turn in all source files to Collab as well as uploading output as specified.

1. Write a program that will determine how many terms are required for the sum 1+2+3+… to exceed a value specified by the user. Print the limit, the number of terms, and the value of the sum. Upload your results for 1,000,000, 10,000,000, and 1,000,000,000.

2. Write a program to print a chessboard of size N (i.e. the board is NxN) where N is specified by the user. Instead of trying to print colored squares (far beyond what we have studied so far), just print a letter R for red and B for black. Thus a 3x3 chessboard will resemble

R B R
B R B
R B R

You will probably want to print a space after each letter to make it look a little more square.

You can tell whether the square should be red or black with the following rule: if the row number and column number are both even or are both odd, then the square is red; otherwise it is black. This should work whether you count from zero or from one.

You will need the modulus function for your language. For Python the modulus operator is %, i.e. if N%2==0 the number is even; otherwise it is odd. For Fortran it is the mod function mod(N,M), so if mod(N,2)==0 the number is even and so forth.

One suggestion for printing is to use some form of non-advancing IO. In Fortran we have already studied this. In Python you may use the write function. The print function we have used so far automatically adds a newline at the end but write does not. You can call sys.stdout.write("R") and it will not advance. To write a newline you can invoke sys.stdout.write("\n"). You will need to add the line

import sys

at the beginning of your script.

Upload your result for the standard 8x8 chessboard.

3. The Fibonacci sequence is a famous sequence of numbers in which each one is obtained by adding the previous two. That is, the sequence goes

1, 1, 2, 3, 5, 8, 13, 21

and so on. Sometimes you will see it start from zero but we are starting with 1 here.

Request that the user enter an integer. Check that the integer is greater than or equal to one. Use an appropriate loop to compute the Fibonacci number of that integer position in the sequence. That is, if the user enters 7 compute the 7^{th} Fibonacci number, which is 13. Hint: you will need to treat terms 1 and 2 as special cases. Another hint: this is an example of a loop where you will need *temporary variables* to save an older result. You will also need to *swap* values; that is, as you advance through the loop your temporary variables must change.

A formula exists for the N^{th} Fibonacci number.

$$F_n = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right)$$

Compute this number and compare to the result of your loop. Note that although mathematically this formula will always yield an integer, numerically that may not be the case, so you should make appropriate corrections. For Fortran I suggest you use double precision.

Print the value from your loop and from the formula to show they agree.

For Python you will once again import math and you will use math.sqrt to obtain a square root. In Fortran you will only need the sqrt() intrinsic. Suggestion: Don't compute the square root of five more than once; store it into a variable.

Upload your results for N=2, 7, 19, and 80.