Homework 2 - Methods in Theoretical Physics (Physics 5350)

I. **DUE DATE**: February 09, 9:00 AM (PLEASE SHOW YOUR WORK)

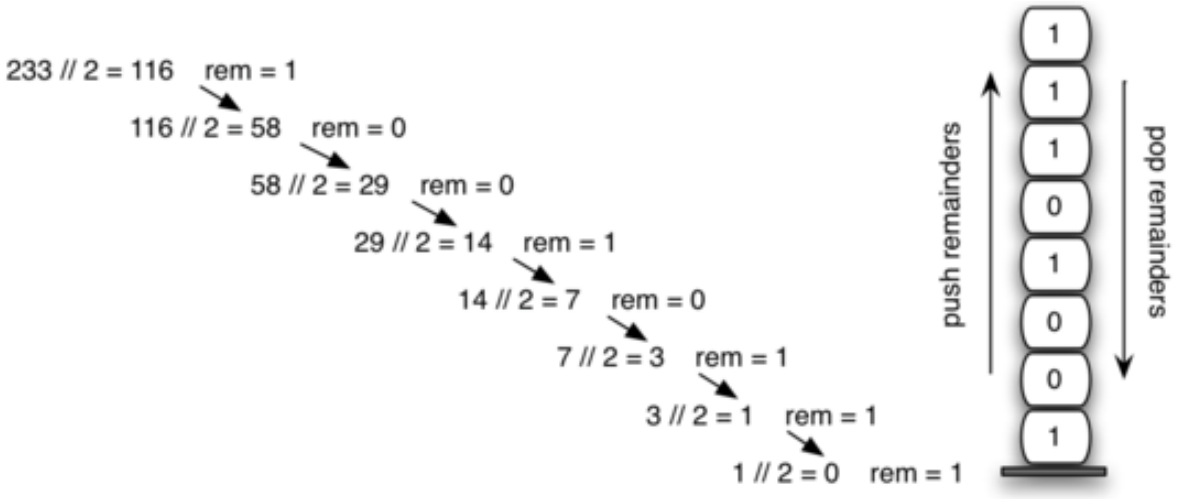
1. Following the example in class to convert hours, minutes, and seconds into seconds, write a Fortran program that does the inverse, i.e, the program should accept seconds as input and convert them into hours, minutes, and seconds. Have the program write the result to the screen.

2. Conversion from Decimal to Floating Point Representation:

(This will follow Homework #1)

Say we have the decimal number 329.390625 and we want to convert it to binary scientific notation. The first step is to convert what there is to the left of the decimal point to binary. 329 is equivalent to the binary 101001001. Below is an algorithm that shows how to do this:

The Divide-by-2 algorithm assumes that we start with an integer greater than 0. A simple iteration then continually divides the decimal number by 2 and keeps track of the remainder. The first division by 2 gives information as to whether the value is even or odd. An even value will have a remainder of 0. It will have the digit 0 in the ones place. An odd value will have a remainder of 1 and will have the digit 1 in the ones place. We think about building our binary number as a sequence of digits; the first remainder we compute will actually be the last digit in the sequence.



Next, we are left with what is to the right of the decimal point, in our example 0.390625. There is an algorithm to convert to different bases that is simple, straightforward, and largely foolproof. Here it is illustrated for base two. Our base is 2, so we multiply this number times 2. We then record whatever is to the left of the decimal place after this operation. We then take this number and discard whatever is to the left of the decimal place, and continue with this progress on the resulting number. This is how it would be done with 0.390625.

0.390625 * 2 = 0.78125	0
0.78125 * 2 = 1.5625	1
0.5625 * 2 = 1.125	1
0.125 * 2 = 0.25	0
0.25 * 2 = 0.5	0
0.5 * 2 = 1	1
0	

Since we've reached zero, we're done with that. The binary representation of the number beyond the decimal point can be read from the right column, from the top number downward. This is 0.011001.

Using the above algorithms, write a Fortran program to convert a decimal number into its binary representation. The program should read the decimal number from the keyboard and write the binary number out on the screen.

3. In Homework #1 we had looked at the "Golden Mean", which is given by

$$\phi = \frac{\sqrt{5}-1}{2} \approx 0.61803398$$

Remember that powers ϕ^n satisfy the recursion

$$\phi^{n+1} = \phi^{n-1} - \phi^n$$

Since we know that $\phi^0 = 1$ and $\phi^1 = 0.61803398$, we can easily calculate higher powers of ϕ using this recursion.

a) Starting from the program on Canvas that you already used in Homework #1, change it to double precision. For this, use the following "integer, parameter" definition:

```
integer, parameter ::  dp=kind(0.d0)
real(dp)    :: phi_0, phi_1
```

This will declare the variables phi_0 and phi_1 as 64-bit floating point numbers. Do the same

for the other variables and pay attention to constant numbers (e.g. 5.0 becomes 5.0d0 or even better 5.0_dp)

Compile and run the program.

Compare your results for double precision with those that you had obtained for single precision. What do you observe?

b) Now, let's push the accuracy of our calculation a little further.

As a first step, get the program kindfind.f90 from Canvas. Compile the program and run it. The output should provide you with an overview of the floating-point parameters that your compiler provides. For example, it might say

```
! FloatingPoint Model Parameters:                          !
! Kind   Precision       Range          Name               !
!    4           6          37          Single             !
!    8          15         307          Double             !
!   10          18        4931          Extnd1             !
!   16          33        4931          Extnd2
```

In this example the maximum precision for floating points is 33 digits and the maximum range is from 10^{-4931} to 10^{4931}. The corresponding maximum kind parameter is 16.

Document the values for your compiler and computer.

c) Next, use the maximum kind parameter for your system (16 in the above example) in the golden-mean program and compile and run it. Document what you find.

Include in the header for each program

- **Your name**
- **The number of the Homework set**
- **The number of the problem**

Please print out your Fortran programs and also upload them to Canvas.