Programming Assignment - 1

Due Date : Wednesday September 11th - 2019 @ 12:30 pm

Note about This Assignment

This assignment lets you get familiar with C++.

It is important that you follow the instructions. Do not try to improve on the design described here. Read the entire assignment before you start working on it.

This is intended to be a warmup assignment. It does not reflect the kinds of things that we will be doing. Expect later assignments to be larger, and to involve principles of data structures.

Background

Given any positive integer *n*, the ***hailstone sequence*** starting at *n* is obtained as follows. You write a sequence of numbers, one after another. Start by writing *n*. If *n* is even, then the next number is *n*/2. If *n* is odd, then the next number is 3*n* + 1. Continue in this way until you write the number 1.

For example, if you start at 7, then the next number is 22 (3 times 7 plus 1). The next number after 22 is 11. The hailstone sequence starting at 7 is (7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1), and it contains 17 numbers. The hailstone sequence starting at 6 is (6, 3, 10, 5, 16, 8, 4, 2, 1), and the hailstone sequence starting at 1 is (1).

Requirements

The requirements tell what the program is supposed to do.

Write a C++ program that reads a number *n* from the user (after giving a suitable prompt) and then tells the user four things:

1. The entire hailstone sequence starting at *n*, all on one line, with the numbers separated by spaces;
2. The length of the hailstone sequence that starts with *n*;
3. The largest number in the hailstone sequence that starts with *n*;
4. How many of the numbers in the hailstone sequence that are even .

The output needs to be sensible and easy to read, not just numbers.

For example, a session with this program might look as follows. Parts in black are printed by the program. Parts in blue are typed by the user.

```
What number shall I start with?  7
The hailstone sequence starting at 7 is
7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
The length of the sequence is 17.
11 of the numbers are even.
The largest number in the sequence is 52.
```

Here is another example.

```
What number shall I start with?  1
The hailstone sequence starting at 1 is
1
The length of the sequence is 1.
0 of the numbers are even.
The largest number in the sequence is 1.
```

The program is ***required*** to be written in accordance with the design issues and algorithmic issues discussed below.

Note :

- **Only positive numbers are accepted.**
- **The program should run until the user types n or N in order to terminate the program.**

Design Issues

The design of a program includes issues such as which functions it includes, how those functions are organized into modules, and what each function and module accomplishes.

For this assignment, each separate program should have just one module.

This is an exercise in using C++, not an attempt to design the most efficient possible solution to this problem. Your design is required to use the following functions. Do not try to fold all of these functions together into one.

1. Include a function that takes an integer *n* and prints the hailstone sequence starting at *n* on one line, separating the numbers by spaces. This function should not print a newline character. Make the return type **void**, indicating that it does not yield an answer, it just does something.
2. Include a function that takes an integer *n* and returns the length of the hailstone sequence starting at *n*.
3. Include a function that takes an integer *n* and returns the largest number in the hailstone sequence starting at *n*.
4. Include a function that takes an integer *n* and returns the number of even numbers in the hailstone sequence starting at *n*.
5. Include a main program that interacts with the user, asking for the start number *n*, and then printing the results.

Each function is required to have a clear and concise contract.

Style Guidelines :

At the beginning of your program (and **before** the #include statement), include the following :

Header comments (file documentation block) should be at the top of each file and should contain: Author / s, Due Date, Assignment Number, Course number and section, Instructor, and a brief description of the purpose of the code in the file. For example :

```
//      Author / s : (Your name here!!)
//      Serial Number / s :
//      Due Date :
//      Programming Assignment Number 1
//      Fall 2019 - CS 3358 - Section Number
//
//      Instructor: Husain Gholoom.
//
//       <Brief description of the purpose of the program>
```

Variable names :

- Must be meaningful.
- The initial letter should be lowercase, following words should be capitalized, no other caps or punctuation (i.e. `weightInPounds`).
- Each variable must be declared on a separate line with a descriptive comment.

Named constants :

- Use for most numeric literals.
- All capitals with underscores (i.e. `TX_STATE_SALES_TAX`)
- Should occur at top of function, or global (only if necessary)

Line length of source code should be no longer than 80 characters (no wrapping of lines).

Indentation :

- Use 2-4 spaces (but be consistent throughout your program).
- Indent blocks, within blocks, etc.
- Use blank lines to separate sections.

Comments for variables :

All variable definitions should be commented as follows:

```
int  gender;    // integer value for the gender,
                // 1 = Male , 2 = Female ,
```

Rules :

1. This is a **individual** work . Group works are not accepted.
2. Your program **must compile** and run using latest version of codeblocks under windows.
3. Your program must be **documented according the style above . See the website for the sample programming style program.**
4. Must **use at least 4 functions (prototypes & definitions) other than your main function.**
5. **You may use switch statements. You are allowed to or can use local arrays / strings / variables. You are not allowed to use vector arrays , global arrays / strings /variables ….**
6. You must use the appropriate libraries in writing this program.
7. Must properly format the output as it is shown on the sample run below. Replace my name with your name

8. You must name your program as :

   - **F19_3358_S#_LastName_FirstName_PG1.cpp // S# is your section number**

   Where LastName is your Last Name and FirstName is your First Name, and S# is your section number. For example , the file name should look something like :

   F19_3358_001_Gholoom_Husain_PG1.cpp (not .cbp) or

9. You must upload your programs no later than the starting of class time on the due date. **No late assignments will be accepted. Everyone must upload their program electronically.**

   Use Tracs To upload your program.

10. You must **also** turn in hard copy of your source code no later than the starting of class time on the due date . should the hard copy consist of more than one page , then , the hard copy must be **stapled**. if you are unable to turn in a printout during class, you can take the program to the computer science department and hand it to the front desk personal (Comal 211) before the deadline. Make sure that the front office stamps the program. Make sure that include the date and time. Finally ,make sure that they place the program in my mailbox. **One hard copy per group.**

    DO NOT slide your program under my office door – It will **NOT** be accepted

The following points will be deducted if :

- Incorrect file format such as uploading.cbp instead of .cpp , missing electronic copy , missing the hardcopy , using .h and .cpp files , Group work, compilation Errors , using data structure other that arrays , using global variables, global arrays … etc. Using **(- 10 points)**

- Other **(at least 1 point each)** :

  - Logical Errors
  - Unable to read the source code due to unclear printing
  - Incorrect program file name.
  - Incorrect output format.
  - Missing function prototypes.
  - More than one hardcopy per group or Hard copy is not stapled.
  - Incorrect Style **such as but not limited to** Missing output header , output footer , comments or program documentations , missing roster number , missing section number, … etc

Sample Run

```
Welcome to the hailstone sequence program
The function of the program is to  read a number n from the user
and then tells the user four things:
        1.      The entire hailstone sequence starting at n, all on one line,
                with the numbers separated by spaces;
        2.      The length of the hailstone sequence that starts with n;
        3.      The largest number in the hailstone sequence that starts with n;
        4.      How many of the numbers in that hailstone sequence are even.

Enter a Positive integer Number  7

The sequence of hailstone starting at   7 is
7  22  11   34  17   52  26   13   40  20   10   5   16  8   4   2   1
The Length of the Sequence is   17
11   of the numbers are even
The largest number in the sequence is    52


Would like to find hailstone sequence for  another number - Enter y or Y for yes
or n | N   for  no   h

Error *** Invalid choice

Would like to find hailstone sequence for  another number - Enter y or Y for yes
or n | N   for  no   Y

Enter a Positive integer Number  4

The sequence of hailstone starting at   4 is
4  2   1
The Length of the Sequence is   3
2   of the numbers are even
The largest number in the sequence is    4

Would like to find hailstone sequence for  another number - Enter y or Y for yes
or n | N   for  no   y

Enter a Positive integer Number  -10


Error ***  Number  must be >  0

Would like to find hailstone sequence for  another number - Enter y or Y for yes
or n | N   for  no   u

Error *** Invalid choice

Would like to find hailstone sequence for  another number - Enter y or Y for yes
or n | N   for  no   N

The hailstone Algorithm is implemented By Husain Gholoom
```