Homework #4
Due by Friday 8/6 11:59pm

Submission instructions:

1. For this assignment, you should turn in 6 '.cpp' files, one for each question 1 to 6.
   Name your files 'YourNetID_hw4_q1.cpp', 'YourNetID_hw4_q2.cpp', etc.
   If a question contains two sections, make a `cout` statement before the implementation of each section (`cout<<"section a"<<endl;`, `cout<<"section b"<<endl;`, etc.)

2. You must type all your solutions. We will take off points for submissions that are handwritten.

3. **You should submit your homework in the Gradescope system.**

4. **You can work and submit in groups of up to 4 people. If submitting as a group, make sure to associate all group members to the submission on gradescope.**

5. Pay special attention to the style of your code. Indent your code correctly, choose meaningful names for your variables, define constants where needed, choose the most appropriate control flow statements, etc.

Question 1:

Write **two versions** of a program that reads a positive integer `n`, and prints the first `n` even numbers.

a) In the first, use a `while` loop.

b) In the second, use a `for` loop.

Each section should interact with the user **exactly** as it shows in the following example:

Please enter a positive integer: 3
2
4
6

Question 2:

In this question we will use a **simplified** version of the Roman Numerals System to represent positive integers.

The digits in this system are I, V, X, L, C, D and M. Each digit corresponds to a decimal value, as showed in the following table:

Roman digit	I	V	X	L	C	D	M
Decimal value	1	5	10	50	100	500	1000

A number in the *simplified Roman numerals system* is a sequence of Roman digits, which follow these 2 rules:

1. The digits form a monotonically non-increasing sequence. That is the value of each digit is less than or equal to the value of the digit that came before it.
   For example, DLXXVI is a monotonically non-increasing sequence of Roman digits, but XIV is **not**.

2. There is no limit on the number of times that 'M' can appear in the number.
   'D', 'L' and 'V' can each appear at most one time in the number.
   'C', 'X' and 'I' can each appear at most four times in the number.

For example: IIII, XVII and MMMMMMDCCLXXXXVII are legal numbers in our simplified Roman numeral system, but IIIII, XIV, VVI and CCXLIII are **not**.

Write a program that reads from the user a (decimal) number, and prints it's representation in the simplified Roman numerals system.

Your program should interact with the user **exactly** as it shows in the following example:

Enter decimal number:
147
147 is CXXXXVII

Question 3:
Write a program that reads from the user a positive integer (in a decimal representation), and prints its binary (base 2) representation.

Your program should interact with the user **exactly** as it shows in the following example:
Enter decimal number:
76
The binary representation of 76 is 1001100

Implementation Requirements:
1. You are supposed to implement the algorithm that converts to base 2. You should not use any `string` or special `cout` functionalities to make the conversion.
2. You are not allowed to use arrays.

Question 4:
Write two versions of a program that **reads a sequence of positive integers from the user**, calculates their **geometric mean**, and print the geometric mean.
Notes:
1. In mathematics, geometric mean of a dataset $\{a_1, a_2, a_3 \dots, a_n\}$ containing positive numbers, is given by: $\sqrt[n]{a_1 \cdot a_2 \cdot a_3 \cdots a_n}$.

   For example, the geometric mean of 2, 9 and 12 is equal to 6 ($\sqrt[3]{2 \cdot 9 \cdot 12} = 6$).
2. In order to calculate the n-th root of a number, you should call the **pow** function, located in the **cmath** library.

Your two versions should read the integer sequence in two ways:
 a) First read the length of the sequence.
For example, an execution would look like:
Please enter the length of the sequence: 3
Please enter your sequence:
1
2
3
The geometric mean is: 1.8171

b) Keep reading the numbers until -1 is entered.
For example, an execution would look like:
Please enter a non-empty sequence of positive integers, each one in a separate line. End your sequence by typing -1:
1
2
3
-1
The geometric mean is: 1.8171

Question 5:

Write a program that asks the user to input a positive integer n, and prints a textual image of an hourglass made of $2n$ lines with asterisks.

For example if n=4, the program should print:

```
* * * * * * *
 * * * * *
  * * *
   *
   *
  * * *
 * * * * *
* * * * * * *
```

Question 6:

Write a program that asks the user to input a positive integer n, and print all of the numbers from 1 to n that have more even digits than odd digits.

For example, if n=30, the program should print:

```
2
4
6
8
20
22
24
26
28
```