NYU, Tandon School of Engineering
CS-UY 1114 Introduction to Programming and Problem Solving — Fall 2015

Homework #5
Due by Monday, 10/26, 11:55pm

Submission instructions:

1. You should submit your homework in the NYU Classes system.
2. Put your code in 9 '.py' files, each containing a script for each question. Name your files 'hw5q1.py','hw5q2.py', 'hw5q3a.py', 'hw5q3b.py', 'hw5q4.py', etc.
   Zip your 9 '.py' files to one file in the following format: '.rar' or '.zip'. Name your zipped file 'Your_name_NetID_hw5.rar' or 'Your_name_NetID_hw5.zip'.
   Make sure your zipped file is not corrupt before submitting to NYU Classes. **Corrupt file will not be graded.**

Key points in this homework:

- 'for' and 'while' loops.
- string and its methods.
- string iteration.

Note: do **NOT** use syntax that was not covered in class.

Question 1:

Ask user to input a positive integer *n.* Your program should print the first *n* odd numbers.

For example, one execution would look like this:

```
Please enter a positive integer: 3

1

3

5
```

Question 2:

Ask user to input a positive integer *n*, and print a textual image of an hourglass made of 2*n* lines with asterisks.

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

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

Question 3:

Ask user to input a string containing only lower case letters. Determine if the input is ordered in a lexicographical increasing order.

For example, an execution would look like:

```
Please enter a string of lowercase letters: abgkp
abgkp is increasing.
```

Another execution would look like:

```
Please enter a string of lowercase letters: abgcp
abgcp is not increasing.
```

Question 4:

Write a program that asks user to input a positive integer n and print out the Roman numeral for n. Your program should be general enough to work with any positive integer n.

Recall from Lab 3, the rules for Roman numeral are:

Number is a combination of the special characters.

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

In this question, use the non-standard form of Roman numerals. For example, if n = 4, print out: IIII (and not IV). If n = 44, print out: XXXXIIII (and not XLIV). Etc.

Question 5:

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.

Note: In mathematics, geometric mean of a dataset $\{a_1, a_2, a_3 \dots, a_n\}$ 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 ($\sqrt[3]{2 \cdot 9 \cdot 12} = 6$).

Your two versions of the program 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 a positive integer: 1

Please enter a positive integer: 2

Please enter a positive integer: 3

The geometric mean is: 1.8171
```

b) Keep reading the numbers until 'Done' is entered.
For example, an execution would look like:

```
Please enter a positive integer: 1

Please enter a positive integer: 2

Please enter a positive integer: 3

Please enter a positive integer: Done

The geometric mean is: 1.8171
```

Question 6

Ask user to input a line of text, and a character *ch*. Your program should:

1. **Create a string variable** that contains the text after removing all occurrences of the character *ch* from the input text.
2. Print that string.

For example, an execution would look like:

```
Please enter a line of text: This is a line of text.

Please enter the character you want to remove: s

Thi i a line of text.
```

Question 7:

Ask 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
```

Question 8:

Ask user to input a positive integer *n*, and print a triangle of numbers **aligned to the right**, where the first line contains the number 1. The second line contains the numbers 1,2. The third line contains 1,2,3. And so on.

For example if *n=5*, the program should print:

```
    1
   12
  123
 1234
12345
```