CS5001 HW4: Due on Canvas Sat Nov 12 at 11:59pm

You must work **either on your own or with one partner**. If you work with a partner, you and your partner must first register as a group with us then submit your work as a group on Canvas. To register, please send an email to the TAs at-- "Qing Chen" chen.qing1@northeastern.edu ; "Lingyu Hu" hu.lingyu@northeastern.edu ; "Mingtianfang Li" li.mingt@northeastern.edu ; "Kejian Tong" tong.ke@northeastern.edu

NOTE: If working in pairs, you can pair up with the same person for a maximum of two assignments/projects in this course. After that, you must either find another partner or complete the assignments individually. Any team in violation of this would receive a zero for their submissions after the first two submissions.

You may discuss background issues and general solution strategies with others, but the programs you submit must be the work of just you (and your partner). We assume that you are thoroughly familiar with the discussion of academic integrity that is on the course website. Any doubts that you have about "crossing the line" should be discussed with Tas or the instructor before the deadline.

Assignment Objectives. Loops, using try-except construct, handling varying user input gracefully.

1. Shapes

In this assignment, we will implement several functions to draw various geometric shapes using asterisks "*" in a python file shapes.py.

For drawing each of the shapes listed below, write a function that implements the logic for creating the desired shape in a string, as described below, and returns the string. When printed, using the print() function, the string should display the desired shape.

For instance, consider the following pattern in the picture below. It is generated by the string s containing " *\n***\n *".

```
>>> s = "   *\n***\n   *"
>>> print(s)
   *
***
   *
```

Note that:

- "\n" is the newline character, when detected in a string, subsequent output will show on the next line or new line.
- Since there are three lines to be printed, the print string s has two newline characters (\n).
- Characters before the first "\n" are printed on the first line. Characters between the two "\n" are printed on the second line, and the characters after the second "\n" are printed on the third line.
    - First line shows three blank spaces followed by an asterisk.
    - Second line shows three asterisks.
    - Third line shows three blank spaces followed by an asterisk.
- Similar to the above, you will be using a combination of blank spaces, asterisks, and \n characters to print the desired shapes.
- You will use appropriate loops to generate the desired strings for printing

1.1 Left Triangle

For this option, implement a function called left_triangle that receives a single integer value "N" as a parameter representing the size of the left triangle to create and returns a string that contains a left triangle of the specified size. For example, a left triangle of size 6 would return a string when printed will look like the following:

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

Note that, for N=6, there are six lines in the output. Asterisks in all the lines are left aligned. Line 1 has one asterisk. Line 2 has two asterisks. Line 3 has three asterisks. Line 4 has four asterisks. Line 5 has five asterisks. Line 6 has six asterisks. Likewise, you can draw left triangles for different values of N.

Hint: You can create each line of the display one piece at a time and then use string concatenation to generate the final string.

1.2 Arrowhead

For this option, implement a function called arrowhead that receives a single integer value representing the size of the arrowhead to create and returns a string that contains an arrowhead of the specified size. For example, an arrowhead of size 5 would return a string when printed will look like the following:

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

Note that the asterisks are left-aligned and for N=5, line 1 has 1 asterisk, line 2 has 2 asterisks, line 3 has 3 asterisks, line 4 has 4 asterisks, line 5 has 5 asterisks; after this, there is a decreasing number of asterisks in the next four lines. line 6 has 4 asterisks, line 7 has 3 asterisks, line 8 has 2 asterisks, line 9 has 1 asterisk. Likewise, you can draw arrow heads for different values of N.

1.3 Right triangle

For this option, implement at function called right_triangle that receives a single integer value "N" as a parameter representing the size of the right triangle to create and returns a string that contains a right triangle of the specified size. For example, a right triangle of size 8 would return a string when printed will look like this:

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

Note that the asterisks are left-aligned and there is a decreasing pattern here. For N=8, line 1 has 8 asterisks, line 2 has 7 asterisks, line 3 has 6 asterisks, line 4 has 5 asterisks, line 5 has 4 asterisks, line 6 has 3 asterisks, line 7 has 2 asterisks, line 8 has 1 asterisk. Likewise, you can draw right triangles for different values of N.

1.4 Boomerang

For this option, implement at function called boomerang that receives a single integer value "N" as a parameter representing the size of the boomerang to create and returns a string that contains a boomerang of the specified size. For example, a boomerang head of size 5 would return a string when printed will look like this:

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

Note that for N=5, line 5 has 5 asterisks; lines 5+1 (6) and 5-1 (4) have 5+1 = 6 characters—2 blanks followed by 4 asterisks; lines 5+2 (7) and 5-2 (3) have 5+2=7 characters---4 blanks followed by 3 asterisks; lines 5+3 (8) and 5-3 (2) have 5+3=8 characters---6 blanks followed by 2 asterisks; lines 5+4 (9) and 5-4 (1) have 5+4=9 characters---8 blanks followed by 1 asterisk. Likewise, you can draw boomrangs for different values of N.

Putting it all together

Once you have all of the shape functions working, you should write a main script in your Python file shapes.py (don't forget to use the conditional if __name__ == '__main__':). The purpose of the main script is to serve as the driver for your functions. Your program should do the following:

1. Print a menu of options to the user and read in the user's choice:

    What shape shall we draw?
    0 - nothing
    1 - left triangle
    2 - arrowhead
    3 - right triangle
    4 - boomerang Enter choice:

2. Prompt the user for the size of the shape to draw (sizes must be greater than 3)

3. Call the appropriate function
4. Print the results to the screen
5. Repeat until the user selects "0 - nothing"

To help clarify what the program functionality, consider the following sample runs of the program:

Let's draw some shapes together:
0 - nothing (terminate program)
1 - left triangle
2 - arrowhead
3 - right triangle
4 - boomerang Enter choice: 1
Enter size (>= 3): 5

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

0 - nothing (terminate program)
1 - left triangle
2 - arrowhead
3 - right triangle
4 - boomerang Enter choice: 3
Enter size (>= 3): 6

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

0 - nothing (terminate program)
1 - left triangle
2 - arrowhead
3 - right triangle
4 - boomerang
Enter choice: 2 Enter size (>= 3): 7

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

0 - nothing (terminate program)
1 - left triangle
2 - arrowhead
3 - right triangle
4 - boomerang Enter choice: 4
Enter size (>= 3): 6

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

0 - nothing (terminate program)
1 - left triangle
2 - arrowhead
3 - right triangle

4 - boomerang
Enter choice: 9

Invalid choice, try again

0 - nothing (terminate program)
1 - left triangle
2 - arrowhead
3 - right triangle
4 - boomerang
Enter choice: x9y

Invalid choice, try again

0 - nothing (terminate program)
1 - left triangle
2 - arrowhead
3 - right triangle
4 - boomerang Enter choice: 1
Enter size (>= 3): abc

Size should be a positive integer!
Enter size (>= 3): -10

Size should be a positive integer!
Enter size (>= 3): 10.99

Size should be a positive integer!
Enter size (>= 3): abx

Size should be a positive integer!
Enter size (>= 3): 5

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

0 – nothing (terminate program)
1 - left triangle
2 - arrowhead
3 - right triangle
4 - boomerang

Enter choice: 0

Thank you for drawing shapes with me!

Rubrics

1. The program should correctly display the menu options as indicated above in the sample outputs. **(10 points)**
2. The program should not terminate until the user explicitly terminates the program by entering a choice of "0". **(10 points)**
3. The program should correctly use try-except construct to handle unexpected inputs like a string for menu choice. An unexpected string/invalid input should not terminate/crash your program. In such cases, continue the program by printing an error message and asking for another input as shown in the sample outputs. **(10 points)**
    a. Similarly, your program should use try-except to handle invalid sizes entered by the user. An unexpected string/invalid input should not terminate/crash your program. In such cases, continue the program by printing an error message and asking the user to enter a valid size as shown in the sample outputs.
4. The program should correctly handle invalid integer menu choices entered. For instance, instead of 0 to 4, if the user enters another integer or float value, continue the program by printing an error message and asking for another input as shown in the sample outputs. **(10 points)**
    a. Similarly, your program should correctly handle invalid sizes (e.g., negative or float sizes) entered for shapes. If the user enters an invalid integer or float value, continue the program by printing an error message and asking for another input as shown in the sample outputs.
5. The program should correctly draw the shapes as described above.
    a. Left triangle **(15 points)**
    b. Arrowhead **(15 points)**
    c. Right triangle **(15 points)**
    d. Boomerang **(15 points)**

What to submit?

1. Prepare a README.txt file containing the following:

a) Include a quick summary of how you run your program shapes.py.
b) If any of the programs is not working, include what is the issue in your opinion and how would you fix it if you had more time?

2. Submit your files shapes.py, README.txt inside a single zip file on Canvas.

Additional directions

Following directions apply to all the problems above. Negative numbers indicate the penalty to be applied on your final score for a problem/direction if the directions are not followed.

1. Start your programs with a docstring (comment) summarizing what it is doing, what are the inputs, and the expected output. (-5 points)
2. At the beginning, as three comments, include your name (both teammates if applicable), date, and your email ids. (-5 points)
3. For every variable defined in your program, include a brief comment alongside explaining what it stores. (-5 points)
4. README.txt file needs to be submitted as described above. (-10 points)
5. Submit all the files (.py and .txt files) as a single zip folder/file on Canvas. (-5 points)
6. No new submissions, code files can be accepted after the deadline. Please double check and ensure that you are submitting everything needed to run your programs, and the code files are the best versions you want evaluated. Only the material submitted in the zip file uploaded on Canvas before the deadline shall be graded.