Published

Week 7: Functions and Values 6 of 6

7 Week 7 Assignment

Computer SCIENCE

You will complete two tasks in this assignment based on the new things you learned in this week. Make sure to use proper naming conventions for your project and name the files for each task as specified below.

Objectives

You will complete three tasks to give you experience with

1. Function design
2. Functions
3. Positional, keyword, default parameters

Task 1 - Number Pyramid

When doing a System Design for functions, you should design a function separately from the design of the main part of the program. After a function is designed, it can simply be called in another part of the design. You will write a program based on a given System Design. Note that the design does have an error in it. You will need to identify the error and fix it.

You will not create a full Software Development Plan for this task. Your program will be created in a file called task1.py. You will create a file called plan1fix.txt, where you will describe in a few sentences what the problem is with the given System Design, and how you fixed it.

Requirements Specification

The user will enter the number of rows to have in a pyramid of numbers. Based on the user's input the program will display a pyramid where each row contains the number *x*, which is printed *x* times, where *x* is the row number. There is a space between each number on the rows. The top row's value is always 1, and the bottom row's value is the number given from the user. Each row is centered in a space equal to the length of the bottom row.

Example Output

```
Enter the number of rows: 9
        1
      2 2
     3 3 3
    4 4 4 4
   5 5 5 5 5
  6 6 6 6 6 6
 7 7 7 7 7 7 7
8 8 8 8 8 8 8 8
9 9 9 9 9 9 9 9 9
```

```
Enter the number of rows: 15
                1
              2 2
             3 3 3
            4 4 4 4
           5 5 5 5 5
          6 6 6 6 6 6
         7 7 7 7 7 7 7
        8 8 8 8 8 8 8 8
       9 9 9 9 9 9 9 9 9
     10 10 10 10 10 10 10 10 10 10
    11 11 11 11 11 11 11 11 11 11 11
  12 12 12 12 12 12 12 12 12 12 12 12
 13 13 13 13 13 13 13 13 13 13 13 13 13
14 14 14 14 14 14 14 14 14 14 14 14 14 14
15 15 15 15 15 15 15 15 15 15 15 15 15 15 15
```

Notes

1. On the bottom row there shouldn't be any space before the first number or after the last.
2. Your centering doesn't have to be exactly the same as the picture above. Notice how the first digit of the first 14 is above the 5, but the second digit of the last 14 is above a space. Your implementation may result in the 14th row (and similar rows) being shifted by 1 space as compared to the picture.
3. If the number of rows goes past 10 there is a ledge in the pyramid. You don't need to space the single digit numbers to span the width. Just one space between each number.
4. The len() function will probably come in handy.
5. Note how in the System Design each function is defined separately, and one function design can call another function
6. This is a similar concept to the multiplication table demo. Check the demo code for a review

Starter File

Follow this file to build your implementation. Once built, you'll notice an error in the design. Create a file called plan1fix.txt to describe the error and how you fixed it. Your code should be in a file called task1.py.

pyramidsystemdesig.txt
(https://usu.instructure.com/courses/681553/files/82331297/download?download_frd=1)

Make sure to review the rubric.

Task 2 - Chessmate

Starter: **assn7-task2-starter.py**
(https://usu.instructure.com/courses/681553/files/81729321/download?download_frd=1)

Video: **Chessboard.mp4**
(https://usu.instructure.com/courses/681553/files/81729316/download?download_frd=1)

Requirements

You will be submitting three files: task2.py (rename the starter code file), chessboard.py, and plan2.txt. You will draw a chessboard based on user inputs. You will ask for the starting location (x, y) and a height and width of the board. You will use this information to draw the chessboard (just a single chessboard). Keep in mind that it may not be a square. The location is the bottom left corner of the chessboard.

The user must enter the location, but if they do not enter a value for height or width you will use a default value of 250. Note, that input() returns an empty string(“”) if the user just hits enter without giving input. Here is some code that you must use in your program. It should be placed in a main() function.

```
    if width == "" and height == "":
        drawChessboard(startX, startY)
    elif height == "":
        drawChessboard(startX, startY, width=eval(width))
    elif width == "":
        drawChessboard(startX, startY, height=eval(height))
    else:
        drawChessboard(startX, startY, eval(width), eval(height))
```

Coding Notes

- The Software Development Plan will change slightly now that we are using functions. You will write an individual design for each function. Then in your main design (where the program starts) you will note where those functions are called.
- Use the assn7-task2-starter.py file to get started. You should add to this file in the areas indicated, but not modify any existing code. This should give insight to the drawChessboard() function definition you need to write.
- Note how the user's input is validated in the starter code. Make sure you're getting input properly so it matches how the existing code will use what is input from the user.
- DO NOT modify the starter code in assn7-task2-starter.py, only add code where indicated.
- Create a module in a file called chessboard.py. This contains all functions associated with drawing the chessboard.
  - Do not use global variables in this file
  - Define drawChessboard() with appropriate parameters
    - This function should draw the outline of the board, then call drawAllRectangles()
  - Define drawAllRectangles() with appropriate parameters
    - This should handle drawing all of the black rectangles by calling drawRectangle()
    - Note: It may be easiest to call this function twice, but not required
  - Define drawRectangle() with appropriate parameters

- This should draw a single black rectangle
- This function will be called many times with a loop!

Make sure to read the rubric to see all requirements.

Common Errors

- The first of the two most common errors is one that really shouldn't be a problem. DO NOT modify the starter code except where indicated. It's actually pretty surprising at how many students modify the code because they think there's an error.
- The code in the shaded box above is very important to understand. You must write code that will work with that code. Review default parameters if you're having problems.

Points	100
Submitting	a file upload
File Types	zip

Due	For	Available from	Until
Oct 22, 2021	Everyone	-	-

Unit 4 (1)

Criteria	Ratings		Pts
Task 1: Implementation matches system design	**8 pts** **Full Marks**	**0 pts** **No Marks**	8 pts
Task 1: Error is discovered and described in plan1fix.txt	**4 pts** **Full Marks**	**0 pts** **No Marks**	4 pts
Task 1: Error is discovered and fixed in code file	**4 pts** **Full Marks**	**0 pts** **No Marks**	4 pts
Task 1: Pyramid output is correct	**4 pts** **Full Marks**	**0 pts** **No Marks**	4 pts
Task 2: Software Development Design Make sure you have a design for each function. Indicate where a function you design is called in another function.	**5 pts** **Full Marks**	**0 pts** **No Marks**	5 pts
Task 2: Get input properly from users	**5 pts** **Full Marks**	**0 pts** **No Marks**	5 pts
Task 2: main() function used properly main() should be the only function defined in task2.py. main() should also be called in task2.py to get the program execution started.	**5 pts** **Full Marks**	**0 pts** **No Marks**	5 pts
Task 2: Create a module called chessboard.py This contains all of the functions that will be used to draw the board. No global variables are allowed in this file	**10 pts** **Full Marks**	**0 pts** **No Marks**	10 pts
Task 2: Define drawChessboard() with appropriate parameters This should draw the board outline, then call drawAllRectangles()	**10 pts** **Full Marks**	**0 pts** **No Marks**	10 pts
Task 2: Define drawAllRectangles() with appropriate parameters This should handle drawing all of the black rectangles by calling drawRectangle() NOTE: It may be easiest to call this function twice in drawChessboard, but not required	**15 pts** **Full Marks**	**0 pts** **No Marks**	15 pts
Task 2: Define drawRectangle() with appropriate parameters This should draw a single black rectangle (This will be called many times by a loop!)	**10 pts** **Full Marks**	**0 pts** **No Marks**	10 pts

Criteria	Ratings		Pts
Task 2: A grid of 8x8 rectangles is properly drawn with alternating black/whites rectangles Really this means to draw the black rectangles	**10 pts** **Full Marks**	**0 pts** **No Marks**	10 pts
Task 2: Proper output for any location, height and width values Height and width can be different and can be blank (then use default) Note: The location should be the bottom-left corner of the chessboard	**5 pts** **Full Marks**	**0 pts** **No Marks**	5 pts
Task 2: Follow proper coding conventions	**5 pts** **Full Marks**	**0 pts** **No Marks**	5 pts
			Total Points: 100