The purpose of this lab is to give you some practice manipulating strings and writing Python functions on your own.

1 Specifications: Read these Carefully

You will need to create a file `lab9.py` that meets the following specifications:

- The first line in the file is a comment that includes your name.
- The second line in the file is a comment that indicates which version of the course you're taking (CPS 300 or CPS 500).
- The file should contain the functions described below.

  Use the same parameter names as stated in the descriptions.
- There should be no global variables or any other definitions. In particular, there should be no `main ()` function.
- Include white space between functions to aid in legibility.

2 Your Tasks

1. Recall the `line` function defined in lecture:

   ```
   def line (width, ch='!'):
       return width*ch+"\n"
   ```

   You should include this definition in your file. You should also make use of it in subsequent functions you write (when applicable).

2. Write a function `mixedLine` that takes four arguments (as defined below) and returns a string:

   - `num1` (required), an integer
   - `ch1` (required), a string (ideally, a single character)
   - `num2` (required), an integer
   - `ch2` (required), a string (ideally, a single character)

   The function returns the string obtained by combining `num1` copies of `ch1`, `num2` copies of `ch2`, and a newline character.

   For example, your function should have the following behavior:

   ```
   >>> mixedLine (5,'*',10,'!')
   '*****!!!!!!!!!!\n'

   >>> print (mixedLine (5,'*',10,'!'))
   *****!!!!!!!!!!
   ```

3. Write a function `rectangle` that takes three arguments (as defined below) and returns a string representing a rectangle with the given characteristics:

   - `height` (required), an integer representing the height (i.e., number of rows) for the rectangle
   - `width` (required), an integer representing the width (i.e., number of columns) for the rectangle
   - `ch` (optional), a string (ideally, a single character) indicating the item to be replicated throughout the entire rectangle

     If this argument is not provided when the function is called, it defaults to the asterisk character (`'*'`).

   For example, your function should have the following behavior:

   ```
   >>> rectangle (3,5,'!')
   '!!!!!\n!!!!!\n!!!!!\n'

   >>> print (rectangle (3,5,'!'))
   !!!!!
   !!!!!
   !!!!!

   >>> rectangle (3,5)
   '*****\n*****\n*****\n'

   >>> print (rectangle (3,5))
   *****
   *****
   *****
   ```

4. Write a function `triangle` that takes two arguments (as defined below) and returns a string representing a right triangle with the given characteristics:

   - `size` (required), an integer representing the height (i.e., number of rows) for the triangle, as well as the width of its base
   - `ch` (optional), a string (ideally, a single character) indicating the item to be replicated throughout the entire triangle

     If this argument is not provided when the function is called, it defaults to the dollar-sign character (`'$'`).

   The intention is that the triangle is rightward-facing, with the point of the triangle at the top and the base of the triangle at the bottom. Each row has one more copy of `ch` than the row above it.

   For example, your function should have the following behavior:

```
>>> triangle (5,'#')
'#\n##\n###\n####\n#####\n'

>>> print (triangle (5,'#'))
#
##
###
####
#####

>>> triangle (8)
'$\n$$\n$$$\n$$$$\n$$$$$\n$$$$$$\n$$$$$$$\n$$$$$$$$\n'

>>> print (triangle (8))
$
$$
$$$
$$$$
$$$$$
$$$$$$
$$$$$$$
$$$$$$$$
```

   Hint: Consider a for-loop that builds up the string one line at a time using concatenation (`+`).

What and How to Submit

Submit this lab through Blackboard. In addition, print out `lab9.py` and hand it in.