Creating Patterns with while loops
CS 5001 Intensive Foundations of Computer Science

In this assignment you'll create a 2X2 grid with turtle. Each quadrant will have a different pre-described pattern, but your user gets to choose which quadrants to fill. To make it even more interesting I want you to allow your user to clear the grid and choose to start all over. You'll get started with help in recitation, but it'll be up to you to finish the application. Good luck.

1. Goals

- Explore iteration with while loops
- Implement a menu driven program
- Add user validation to a simple program
- Work through more complicated algorithmic ideas

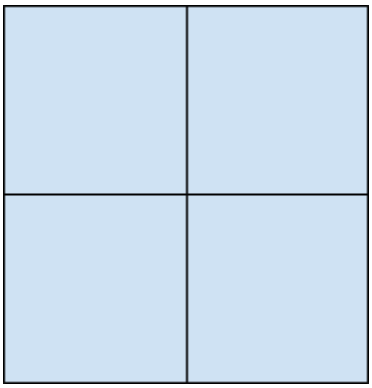
2. In Recitation

I'd like you to use recitation to build a frame for your application with the help of your TA. Here's what I'd like you to accomplish:

Objective 1 : Setup your program workspace

Objective 2 : Add each of the following functions making sure to test each as you create them:

a. `goto(x,y)` - moves a turtle cursor without drawing
b. `reset()` - moves the turtle cursor to home position without drawing
c. `square(x,y,len)` - draws a square at a given position with a provided side length
d. `frame(box_size)` - draws a grid of 4 boxes in the below pattern. Each box should have sides equal to the sent value.



e. `windowSlates(x,y,blockSize,width)` - draws horizontal stripes at the given position equidistant apart based on the block size. See the upper left quadrant in the image below.

f. menu(): - print the following menu:

```
1. Fill quadrant top left with vertical bars
2. Fill quadrant top right with zebra stripes
3. Fill quadrant bottom left with stairs
4. Fill quadrant bottom right with wallpaper
pattern
5. Fill all quadrants
6. Clear screen
7. Quit
```

Objective 3 : First watch your TA run a completed version of this program without seeing any of the code. Then, comment out all of your testing and add the below block of comments to main to help you get started. Discuss this with your TA and gather some tips and tricks to get started with the rest of the application:

```
#1. max out the turtle drawing speed
#2. setup any variables you are going to need
#3. create an infinite while loop
#4. print user menu
#5. ask user for selection
#6. validate user input
#7. call fill method based on user selection
(See instructions below)
#8. or if the user selects 6, clear and redraw
the frame
#9. or if the user selects 7, close the turtle
window and exit the while loop
```

Note: These starter assignments are meant to help you get started with each assignment by allowing you to code as a group and provide each other assistance. Make sure you understand everything that you code.

For your recitation ICE : Your TA will divide you up into groups for a planning session. Submit a short paragraph explaining a step-by-step procedure on how you will approach this problem based on this meeting. Will you flow chart; build a frame; or other?

3. Lab Assignment Instructions:

Add the following functions:

`fill(quad1, quad2, quad3, quad4, window_size)` - quad1, quad2, quad3, and quad4 are boolean variables letting the function know which quadrants to fill and window_size is the size of square to be filled.

`zebra_stripes(x, y, block_size)` - creates vertical strips across the screen to fill a given block

`steps(x, y, block_size, step_size)` - creates steps from the bottom left to the top right of a block with the given step size.

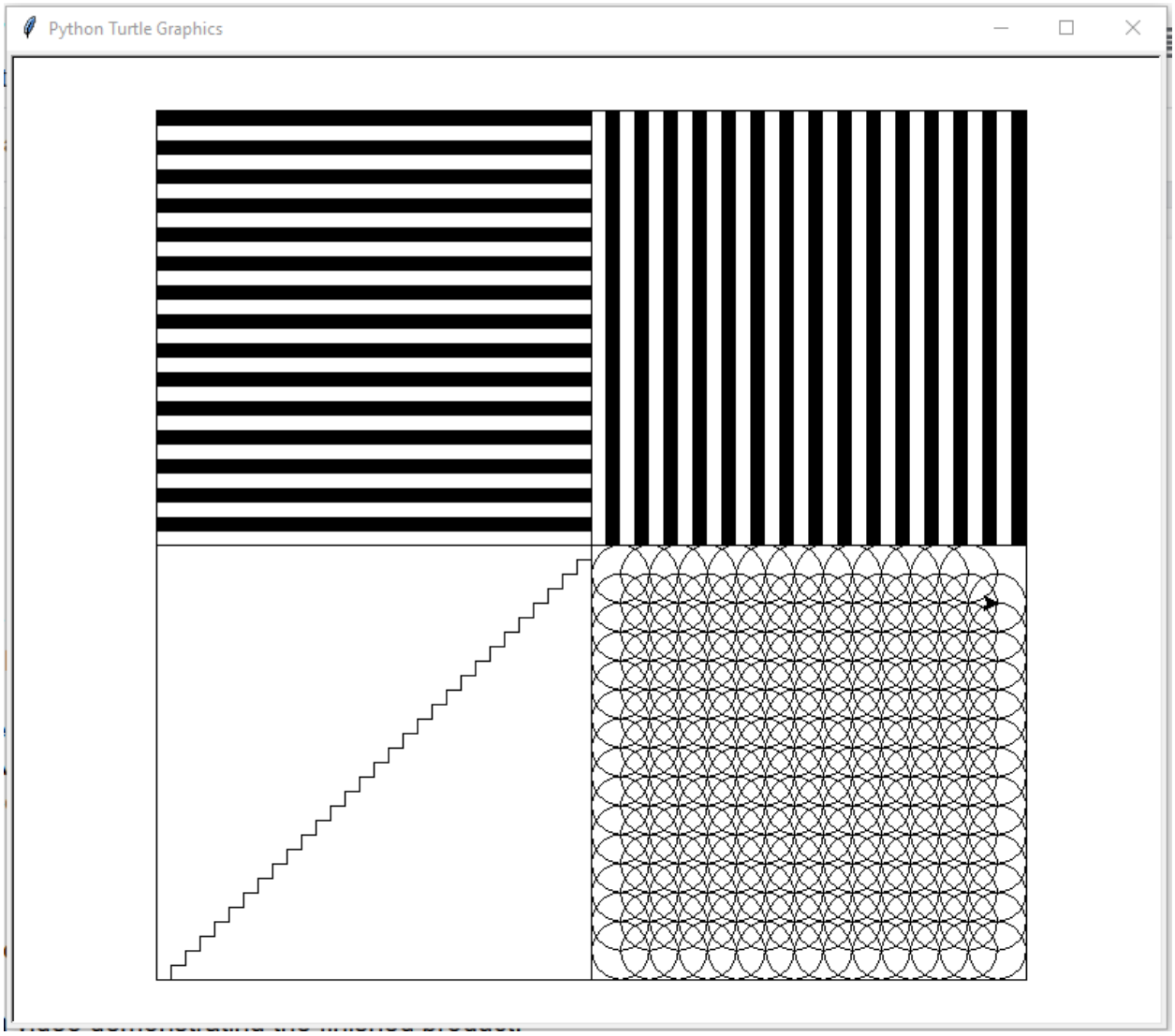
`wall_paper(x, y, block_size, radius)` - creates a while paper pattern with interlocking circles based on the provided radius. Make sure to stay within the block.

Build out all of the above functions and finish your application as a completed application.

Notes:

- Make sure to stay within the quadrant
- Make sure to completely fill the quadrant
- Only ask for frame size once
- Make sure you reuse code where you can
- You can add helper functions if needed
- You can modify function inputs for your design, as long as there is a valid reason to do so

Finished produce with all quadrants filled:



4. Extension suggestions:

Remember you can get up to 26/30 points for finishing the lab, but for 30 points you have to go above and beyond. You don't have to use any of the below extensions. They are just examples. Explore on your own and come up with something fun.

Extension ideas:

1. Add color to your pattern creations
2. Create additional boxes in your frame and add new patterns.
3. Separate your code into separate files for portability
4. Add text or other bells and whistles to your application

5. Report:

There was a lot of thinking needed in this week's assignment. So, let's just do a small report. You may use a word processor of your choice, but please submit your report as a pdf with the name report04.pdf along with your code.

Reflection:

How was this application designed with usability in mind?
What was the easiest and hardest part of this assignment?
What did you learn?
What grade would you give yourself?

Extension:

What extension did you add to the assignment?

6. Submission:

You should be submitting these at the least, but you may submit more files if you separated out your assignment for organizational purposes:

- Report04.pdf
- patterns.py

Submit your project code on canvas as Lab4_"Your_name".zip

When you submit, double-check the following.

- ☐ Is your name and an appropriate header at the top of each Python file?
- ☐ Does every function have a comment or docstring specifying what it does?
- ☐ Does your report have all six sections completed?
- ☐ Is your report a pdf document?

7. Rubric:

	Possible	Given
Requested functions		
all of the functions requested in recitation	3	0
fill(...)	3	0
zebra_stripes(...)	3	0
steps(...)	3	0
wall_paper(...)	3	0
Misc		
Application complete/works as requested	3	
Report (all or nothing)	4	0
Code Quality (correct indentation, comment blocks, variable naming, etc)	4	0
Not included in total possible:		
Quadrant fills exeed boarders	-8	0
Other drawing issues	-15	0
Extensions (Not calculated without report)	4	0
Creative or went above and beyond	4	0
Code does not compile	-30	0
Late penalty	-6	0
Not implemented as requested	-30	0
TOTAL POINTS POSSIBLE out of 30	26	0