Computational Problem Solving Sampler Quilt

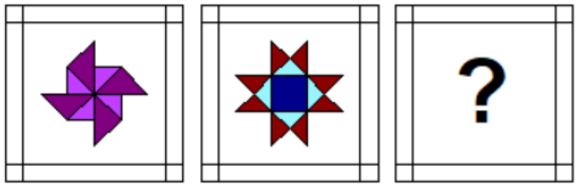
CSCI-603 Lab 1

20/05/2023

1 Problem: Designing Sampler Quilt[1]

A sampler quilt is a particular type of quilt invented by American quilters in the early 1800s. This type of quilt is made of different patchwork blocks of the same size, but not necessarily the same pattern or even the same fabric, laid out in a grid.

For this first lab, you will be required to write a program that draws three different patchwork blocks using Python's library. The image below shows the design of two of the blocks you must generate. For the third block, the one with the question mark, you must come up with your own design. Be creative!



2 Implementation (80%)

You will **individually implement** and submit your own solution to the problem as a Python program named `quilt_design.py`.

2.1 Requirements

Your program must do the following:

1. Draw one copy of each patchwork block in a row on the display in any order. Each block is fully visible and aligned in a neat row on the canvas.

1. This problem is inspired by Julie Zelenski's Quilt lab (`https://cs.stanford.edu/~zelenski/quilt/`)

2. Every block must be a perfect square of the same size. They also must have the same outline pattern. The functions drawing the blocks must reuse the same code to draw the border pattern. That way the size and proportions are the same.

3. The block structure has a central region which contains a particular drawing. There must be some separation between the central region and the outline border.

4. You may not write one function that draws an entire patchwork block. The pattern at the center (e.g. the windmill) must be drawn 'on top of' the block central region.

5. Every block must be some turtle-drawn graphic. You may use functions you researched and found in the turtle library module. However, the program must not use functions that insert/include image files. The graphic must be filled with one or more colors. The color palette of the image above is just a suggestion. You can choose whatever color scheme you want.

6. The drawing of a block must start from the same position relative to the block area, and finish so that the turtle is ready to draw the next block after the current block in a row on the display canvas.

7. You are not allowed to use commands like `turtle.goto`, `turtle.setPos`, or similar. You must move the turtle by using the commands `left`, `right`, `forward`, and `backward`.

8. Your functions must have docstrings that tell programmers what the function does. Use pre and post conditions to indicate the relative position of the turtle.

2.2 Block 3: Question Mark

The question mark block cannot be part of your final implementation. Instead you must replace this block with your own design, it's entirely up to you how fancy do you want to go. These are the only requiremets you must meet:

1. Your design must follows the requirements stated in the previous section.

2. Your design must be significantly different from the other two blocks, changing only the colors or rotating the drawings is not allowed.

2.3 Tools

To implement the filling of polygons, you'll need to learn to use the `turtle.begin_fill`, and `turtle.end_fill` functions. The code sequence:[2]

2. Note that these are Python 3 specific. You must follow the exact order above to draw a filled figure element, and the figure must define an enclosed area.

```
turtle.fillcolor( "blue" )
turtle.begin_fill()

# code to draw a single, closed-curve figure element.

turtle.end_fill()
```

will draw a single, closed-curve figure of a specific fill color, in this case blue. A blue line color can also be specified with `turtle.pencolor( "blue" )`.

2.4 Formula Calculations

To draw lines written on a diagonal, you will require mathematical functions, e.g. roots, trigonometry. Include `import math` at the top of your program file. To find the name of the math function you need, search the library documentation for the Python math module.

2.5 Program Operation

From inside PyCharm, you can run the program to display the patchwork blocks.

From a terminal window, you can run `python quilt_design.py` assuming that `python` is version 3.

When run, the program will draw the blocks, and wait. The wait can be done by including `turtle.done()` at the end of your program. This function call will pause the program until the user closes the window containing the turtle's canvas. Without this the canvas will disappear and the program will terminate without giving the user a chance to see the final turtle drawing.

```
turtle.setup(400, 200)
# set the world coordinates to match the number of cards.
turtle.setworldcoordinates(-15, -25, 350, 150)
```

Use the exemplar above to set up the window size to fit the blocks. The first function call determines how big the canvas will be, in pixels, on your computer screen. The second call sets up a *scale* of coordinates. In the above case the x coordinate ranges from -15 to 350 and the y coordinate ranges from -25 to 150. (We use a negative number at the lower left so that the home point (0,0) is near the bottom left corner.)

However, you are not required to adjust the size of the window. If you do not adjust the size of the window, your output must fit in the default window, you must adjust the size of your blocks accordingly. It is okay to have extra space around your output, as long as the blocks are visible.

3 Grading

The assignment grade is based on these factors:

- 20%: Attendance at problem-solving and results of problem-solving teamwork.
- 35%: The program draws the two required blocks, and the border encloses the entire content of each block design following the requirements.
- 15%: The program draws the third block, and the border encloses the entire content of the block design following the requirements.
- 20%: The design reuses multiple functions both to implement the layout of the block and to draw the patterns at the center.
- 10%: The code follows the style guidelines on the course web site. For a full style example refer to here.

Note: you will have 10 points deducted if your submitted file is named incorrectly or the code is written in Python version 2. The submitted file must be a **Python 3 program**.

4 Submission

Submit your `quilt_design.py` file to the MyCourses assignment before the due date.

- Go to the MyCourses assignment.
- Upload your `quilt_design.py` file to submit your work into Lab1 dropbox. You can upload to myCourses as often as you like, but only the last submission counts.
- Check that your uploaded submission worked. If you forget a step, your work may not be submitted, and you will lose credits.

Double check by going to the dropbox and refreshing. Also, you should receive an email when you successfully submit.