Specification

Introduction

Make sure you complete Creative Project 1: Drawing Shapes Tutorial before starting this assignment.

Learning Objectives

In this assignment, you will demonstrate your ability to:

- Write a `for` loop
- Write a nested `for` loop
- Call a library method
- Read the documentation of a Java library to figure out how to call a formerly unknown method
- Observe and document your personal debugging process

Debugging

Throughout this assignment you will be asked to reflect on your debugging process.

Debugging resources

Here are some course resources on debugging that you've seen already:

1. Several slides on debugging and interpreting error messages from Creative Project 0: Hello World! and Debugging.
2. Debugging practice from Section 3: Variables, Debugging.

As you work on this assignment, look back at these resources as needed.

Take notes about your debugging process

As you work on C1, take notes about the error messages and bugs you encounter, as well as your debugging process. This means you should be noting the tools and strategies you use to understand the errors, debug, and fix your code.

Here is what you should record in your notes:

- Record at least one time when your C1 program's behavior differed from what you wanted it to do.
    - What are the differences between the actual and desired behavior?
    - What is your process for figuring out the problem with your code?
    - What is the solution to this bug? That is, what do you need to fix about your code to make it have the desired behavior?
- Record at least one time when your C1 program didn't compile.
    - What compilation errors did you get?
    - What is your process for figuring out the problem with your code? Describe the resources or strategies you used in this process.
    - What is the solution to the bug? That is, what do you need to fix about your code to make it compile?

You do not need to turn in all of your debugging notes, but you will need to answer questions about your debugging process in your Reflection.

Reflection

The last slide of this assignment is a reflection with questions about each part of the assignment. The reflection is a significant portion of this assignment both in terms of time and grade. We recommend that you take a look at the reflection questions now and answer them as you complete each part of the assignment, rather than saving all the questions for the end.

Program Behavior

Task 1: Draw a regular shape

Your first task for this assignment is to draw a **regular shape** (a shape that has the same edge length and angle size for all of its edges and angles)!

- Write one for loop to draw a shape from the table below. **Do not choose a shape that is crossed off in the table as those were covered in previous exercises.**
    - Each iteration of the loop should draw one edge and vertex of the shape.
- Below is a table listing the angles that the Turtle would have to turn at each vertex to draw some common shapes.
    - We recommend you choose one of the common shapes in the table below, but if you choose to draw your own shape instead, it must be a regular shape.
- Write your code for this task in the `main` method of `Shapes.java` beneath the comment that says `// Task 1: Draw a regular shape`
- Take a screenshot of your image to turn in as part of your writeup.

Task 2: Draw a regular shape many times

The second task for this assignment is to draw your regular shape many times!

- Using nested loops, draw your regular shape from Task 1 **at least 10 times** on the canvas.
    - Each iteration of the inner loop should draw one edge and vertex of your shape
    - Each iteration of the outer loop should draw a whole shape once
    - Each iteration of the outer loop should add noticeable complexity to your drawing. That is, don't trace over exactly the same lines so that one iteration hides the work of a previous one. Your shapes can touch or overlap partially, and you can choose how to separate them (angle, position, size, etc...)
- Write your code for this task in the `main` method of `Shapes.java` beneath the comment that says `// Task 2: Draw a regular shape many times`
    - Before your Task 2 code, make sure to move the Turtle to a different part of the canvas so that the drawing from Task 2 does not overlap with the drawing from Task 1.
- Take a screenshot of your image to turn in as part of your writeup.

Applications

You must complete at least two applications to earn a grade of E on this project. See the Creative Project Rubric for details.

Instructions

- Write your code for your first application in the `main` method of `Shapes.java` beneath the comment that says `// Application A:` . Make sure to add the name of your

chosen application to that comment, for example: `// Application A: Triply Nested Loop`.

- Write your code for your second application in the `main` method of `Shapes.java` beneath the comment that says `// Application B:`. Make sure to add the name of your chosen application to that comment, for example: `// Application B: Random`.
- You may write code for additional optional applications in the `main` method of `Shapes.java`. Be sure to add a descriptive comment above the code for each optional application so it's easy to see where that application's code begins.
- Before the code for each application, make sure to move the Turtle to a different part of the canvas so that the drawing from each application does not overlap with the drawing from earlier tasks.
- In your write-up, describe what you did for each application and include separate screenshots of the drawings that your program produced for each application.

Choose two or more of the following applications

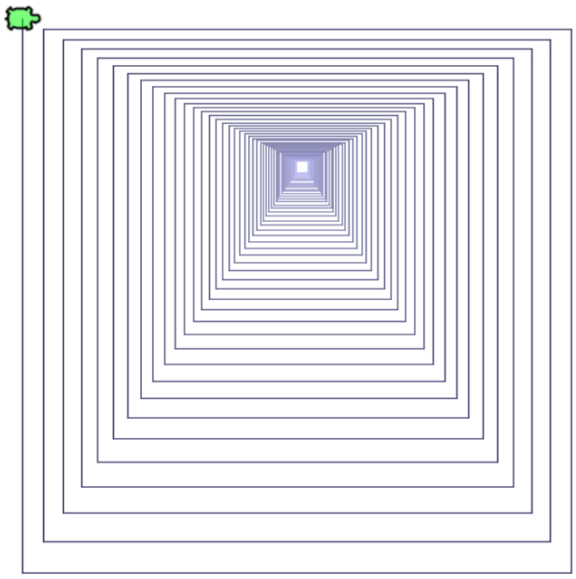
1. **Additional use of the loop variable.** In each iteration of a loop, use the loop variable (or some value computed from the loop variable) to determine some attribute of what is drawn in that iteration.

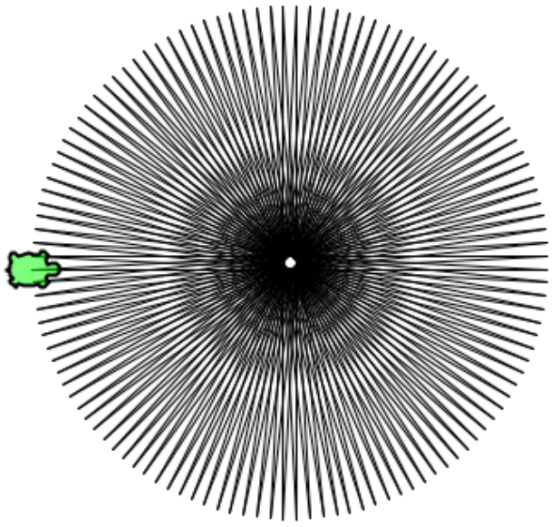
   Here are some ideas for values that you might compute based on the loop variable. You can choose your own idea if you'd like, just make sure that the result of the computation can be seen in the resultant drawing.
   - The distance to move forward
   - The coordinates of the location to move to
   - The number of iterations for an inner loop
2. **Random.** In each iteration of a loop, generate a random number, and use it to determine some attribute of what is drawn in that iteration. For example, draw one shape multiple times in random locations and/or at random sizes

   Here are some resources to help you with generating random numbers:
   - Pre-class work on nested for loops and Random
   - Java library documentation for the java.util.Random class
3. **Many Iterations.** Write a loop whose body gets executed at least 100 times.
   - Each iteration should add noticeable complexity to your drawing, e.g., do not just retrace the 4 sides of a square 25 times.

     Here are a couple images that were drawn with loops that executed over 100

times. Feel free to use these ideas for inspiration:





4. **Triply nested loop.** Use a triply nested for loop to add even more complexity to your drawing than with a doubly nested loop. All loops **must** have at least one turtle statement/action in its body.
5. **Choose your own adventure!**
    - This must involve innovation beyond repeating something you were already asked to do in the concrete exercise. If you want to get credit for this application, describe your idea on this discussion board thread to get approval no later than **11:59pm on Sunday, January 22**. Approval means that your idea, *if executed well*, can earn credit for this application-- it does not guarantee any particular grade.

The next slide contains the workspace where you should complete and submit your program. The last slide in this assessment contains the reflection write-up that you must complete to earn full credit for the assessment.