COSC 1P02 Assignment 1

"Crystal Clear"

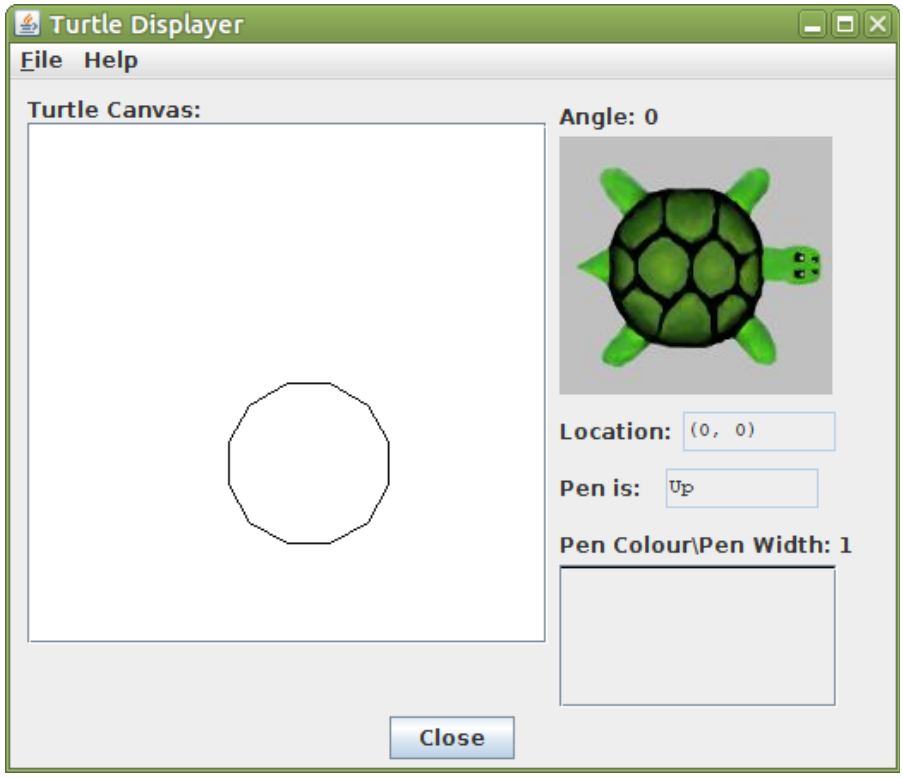
Refer to Sakai for due date

This assignment is meant to reinforce *composition*.

In preparation for this assignment, create a folder called `Assign_1` and for the DrJava project for the assignment. The problem is described in two parts, however you only submit the final result from Part B.

Part A

As part of a package called `Assign_1`, write a Java class called `Dodecagon`. A dodecagon is a regular closed figure with 12 sides. The class should draw a dodecagon with sides of length 25, with the first side drawn from the initial turtle position, in the initial turtle direction, and should produce a drawing as below:



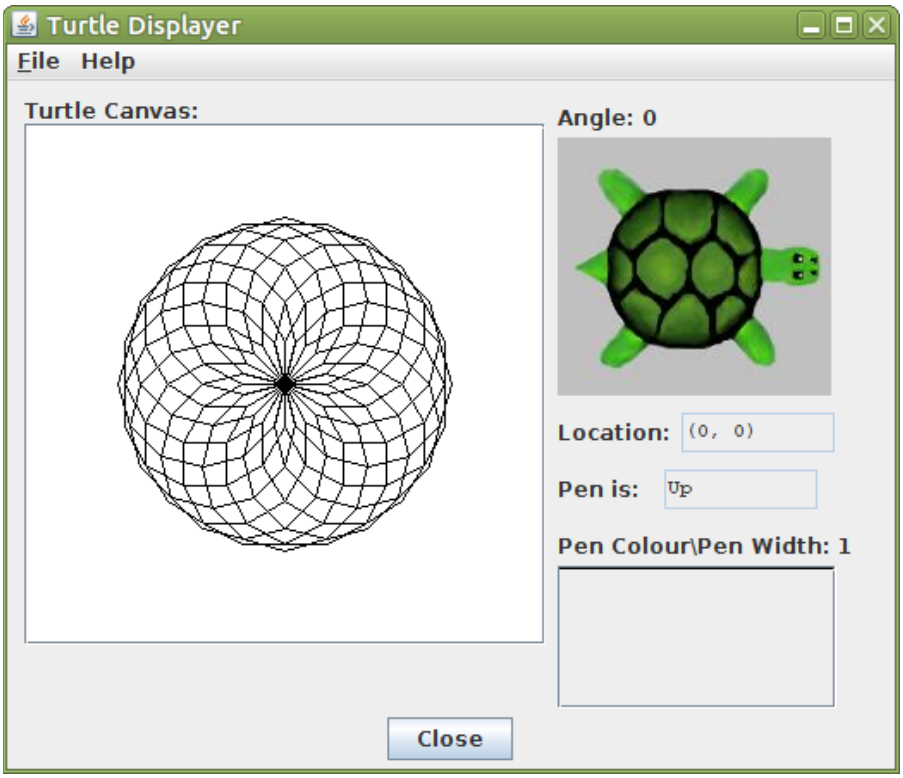
Note that this involves some *repetition*.

Part B

For the actual task of this assignment, adapt your source file to write a new Java class: `Crystal`. The crystal is comprised simply of twenty-four dodecagons, all rotated about a single point.

This solution should involve *nesting*. i.e. a crystal is composed of multiple dodecagons, with minor reorientation between each polygon.

Refer to the next page for a sample.



Tips:

- Rather than modifying your Dodecagon directly into your Crystal, you're probably better off copying&pasting/renaming the file
  - Remember to change the `public static void main...` line to reflect `Crystal`!
- Remember, a full circle is 2π
  - Also, remember that you can get π with `Math.PI`
- You might wish to speed up the turtle. e.g.:
  - `squirtle=new Turtle(Turtle.FAST);`

Submission:

For submission, you must submit a .zip file containing the following:

- A folder called `Assign_1`, containing your submission
  - At least your `Crystal.java` source file
  - Optionally, also your `Dodecagon.java` file
  - Your `.drjava` file (or other project file; see below)
  - A `.pdf` copy of your crystal pattern
    - When the program finishes, before clicking *Close*, just click *File* → *Print Image of Window...* → and then select to print to a file
    - If you're having any trouble doing this on Windows, CutePDF might make it easier

        - If you find yourself struggling with the output shortly before you need to submit, for *only this assignment*, you can just use *File → Save Window as Image...* and save it as a picture instead

On Sakai, submit your .zip file as an attachment, and click to Submit.

Note: Do not submit a `.rar`, `.tar.gz`, `.7z`, etc. *Every* major operating system supports `.zip`, so it is mandatory if you wish to receive a grade for your assignment.

Standards:

Ostensibly, you'll be graded for following basic coding standards and documentation requirements. However, as we haven't really learned any yet, all you really need is:

- A comment at the top of all source (`.java`) files including your *name* and *student number*
- Variable names that are either standard or descriptive
    - Using things like `i` and `j` for loop counters? Standard
    - Using a variable like `count`? We can guess what you meant
- For any *blocks* of code (e.g. loops) insert a brief comment so the reader knows what's inside (e.g. *what*'s being repeated)
- Anything you think the marker might not understand? Add a quick comment
    - This *probably* isn't a concern for this assignment

Additionally, try to remember to fix the indentation of your source files. Jagged margins can be harder to follow along with.

DrJava (and other platforms)

DrJava is recommended, to make grading as easy as possible. However, anything that will run on the COSC lab computers also works.

Make sure to include *everything* (i.e. the whole folder), to avoid forgetting an important file. (e.g. even if you include a .class file, and a .pdf copy of the source file, the marker can't confirm that compiles)