CS 251: Code warrior: Java and Simple Gui (Outlab)

- Due: 11:55 AM 10/10
- Please write (only if true) the honor code. If you used any source (person or thing) explicitly state it. You can find the honor code on the web page.

Overview

We introduced you to Java in the inlab, and here we will dive deeper into some of its features. Along with that we want to expose you to the Graphical User Interface in Java using Swing. Swing is the official way in Java to create GUI for desktop applications (unlike the most common web applications you see these days), although JavaFX is gaining ground.

For swing, see this compact tutorial introduction[1]. You probably need the first 8 steps barring the font step.

A: Prisoner of Azkaban

In the inlab, you learned the basics of using the eclipse IDE. In this task, we will take a look at some other useful features of the IDE.

0. Download the `Azkaban.zip` file from support and import this project into eclipse by choosing the archive file option
1. After importing, you can see right away that there is a red cross beside the project. This means that there are errors in the project somewhere. Track down these errors and fix them. Run the code after fixing the error.
2. You can see in the output that it says “Sirius is a criminal”. But, we know that’s not true. So, the code still has bugs and these need to be fixed. Use the debug mode of eclipse to find the bug and fix it so that the program prints “Sirius is innocent.”
3. Mention all the steps you have taken to achieve this in the file named `A.txt`. you have to submit the debugged Project as an archive with the same name `Azkaban.zip` Make sure that we can import the file, but do not include any unnecessary files; we simply want to recompile and run the sources.

B: The Triwizard tournament

Harry has reached the final of the Triwizard tournament (can’t rename the legendary cup to Quadwizard tournament, right?). He just entered the maze. In his pocket, he has a map showing the complete structure of the maze given to him by Mad Eye Moody. He heard that you guys learned Dijkstra’s algorithm in lab06. Help him find the shortest path to the trophy so that he can win.

0. The entire map will be given to you in the form of a weighted graph. The weight of each edge represents the time it takes to cross that edge. You have to give Harry the path that takes the least amount of time in order that he reaches the cup before Krum, Cedric or Fleur.

[1]BTW, this HTML document was created in LaTeX – scorll the end.

1. The first line of the input file `map.txt` will have 4 numbers, the number of vertices (m), the number of edges (n) in the map, the start vertex s and the end vertex t. The next m lines have 3 numbers each, the vertices (v_i, v_j) and the weight of the edge between the two vertices (w_{ij}). Vertices are numbered from 1 to m. Write a Java program, `Map.java` which reads `input.txt` file, builds the graph and finds the shortest path from vertex s to vertex t using Dijkstra's algorithm. The output of the program should be the total time to reach the vertex t from vertex s and the path to be followed for that. If there exists no path, return a statement saying so onto the screen before exiting the program.

   Your program need not be super efficient (but you should know what you are submitting :) Also the input will be in the format specified, so do not worry about error handling. (You have had enough of it in the inlab).

C: Priori Incantatem

Priori Incantatem gave Harry time to escape from the graveyard when the Dark lord was resurrected. The spirits wanted to give Harry time to escape so they started to mimic the famous Producer-Consumer problem. The spirits are the producers, and Lord Voldemort is the consumer. Spirits kept on producing stuff and Voldemort destroyed them in order. Assume there is a limited buffer, and producers can fill it till it is full and Consumer can remove/destroy only when it is not empty.

0. Read about the producer consumer problem. In this task, you will implement the solution for the one producer and one consumer case of the problem using threads in Java.

1. Create one thread each for the producer and consumer and mimic the entire problem where buffer size is 5 (five). You might want to look at wait, notify and synchronized blocks of Java. Name your java file as `ProducerConsumer.java`

We shall skip the most boring book - *Order of the Phoenix*

D: The Half Blood Prince

Vanishing Cabinets are a really good analogy of client server computing. When there was a failed connection between both the vanishing cabinets (even when Draco sent messages) he was not getting the expected reply.

0. In this part, you will learn how to connect to a server using a Java program and get data from it.

1. Write a Java program `Prince.java` which connects to 10.129.3.2 (port 80) and gets the name of the half blood prince from there. The path on 10.129.3.2 is ~sharat/current/cs251/Assign/Lab09/support/Prince.php. Output the line (possibly with spelling mistakes) obtained from the server to the file `secret.txt`.

E. Final War (Battle of Hogwarts)

Every book reader knows that David Yates has spoiled the war completely in the movies. There were cases where he made Harry and Dark-Lord go in circles for no reason while they were flying. David Yates wanted you guys to make something for him getting his circles business done. In this task, you'll design GUIs (Graphical User Interfaces) in JAVA using swing.

You can look at this link for how the initial graphical sketching program called Sketchpad looked like. Graphics has evolved!

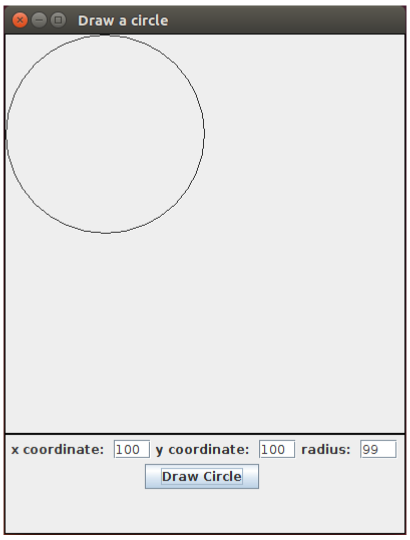
For this task create a folder called `taskE`

1. The circle is to be drawn by providing the center, and radius. Your interface consists of two parts.

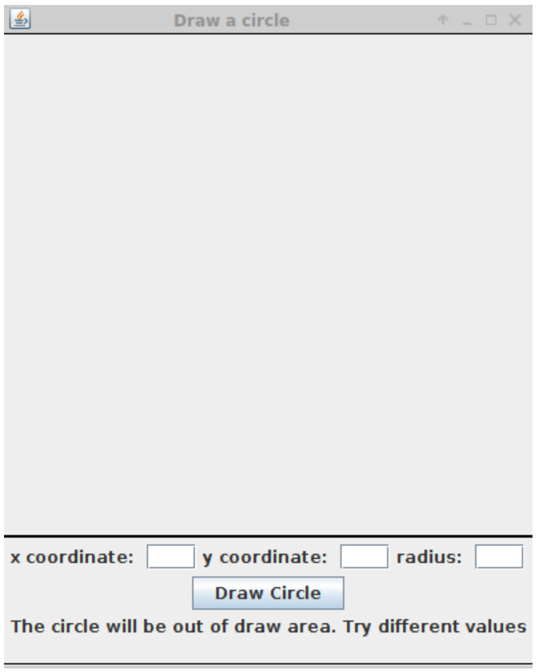
   The first part is the drawing area (towards the top portion of the GUI). The second is the controller area contains areas for providing the inputs: the x co-ordinate,the y co-ordinate, and finally the radius. There should also be a button, which on clicking, accomplishes the task, namely, drawing the circle.

   Sanity checks have to be done on the input values. If the input fails an error message message should be printed like in 1b. Note that the entire window may be damaged: it may be, for example, moved, resized, hidden (and then brought to the front). You should handle all these events gracefully. You could also consider making the picture more elegant (e.g., using a thicker pen brush so that we can actually see the circle), but leave the UI otherwise uncluttered. Close the application by using the standard "cross" mark in the window decoration (no need for an exit button). How does your program behave when we enter two sets of parameters (for two different circles) consecutively?

   Create a folder named `task0`. The code for the above sub-task should be in the file named `DrawCircleInGUI.java`. All other java files created as helpers should be included in the same folder; mention the functionality of each file in `readme.txt`.



(a) Drawing a circle



The circle will be out of draw area. Try different values

(b) Error message

Figure 1: : Circling around drives me nuts.

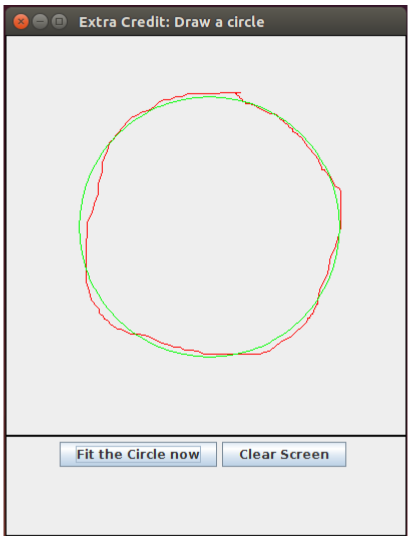
2. **Extra Credit**: In this part, create a new file named `FitCircleInGUI.java` in a folder named as `Extra-Credit`.

   We actually want to draw a circle free hand but it should be a perfect circle. Design a GUI which on running, displays an interface consisting of two parts. See Example: Fig 2a.

   Functionality: When a user drags the mouse in the drawing area by holding the left mouse button down, a trail of the path should be left behind on the screen. You should also have a button named "Clear screen". If the user feels any time that the contour has gone wrong, clicking the "Clear" button should erase the current drawing. Once the user has completed her task, clicking "Fit the circle now" should display the (so-called) best fit circle. An example code for getting the best fit circle is found in the support folder named `CircleFitter.java`. Use this code to draw the output circle. The color of the line to be drawn by the mouse pointer should be red in color and that of the fitted circle should be in green color. Both figures are to be displayed after clicking the "Fit the circle now" button.

Finally output the center, and radius of the drawn circle as a message (similar to the error message).

Any other java files created as helpers should be included in the same folder and mention what each file does in `readme.txt`. Also submit `CircleFitter.java`.



(a) Fit circle. Notice the red and green contours

Figure 2: Extra Credit GUI

Submission Guidelines

1. When you submit, please document individual percentages such as Student 1: 80%, Student 2:100%, Student 3:10% in the readme.txt. In this example, the second student will get full marks (10/10) and the first student will receive 8/10.

2. Do include a readme.txt (telling us whatever you want to tell me). (The reflection essay is not required for inlab, but is required for outlab.) Do include group members (name, roll number), group number, honor code, citations etc.

3. The folder and its compressed version should both be named `lab09_groupXY_outlab` for example folder should be named `lab09_group07_outlab` and the related `tar.gz` should be named `lab09_group07_outlab.tar.gz`

4. The submission folder should contain 5 sub-folders `taskA`, `taskB`, `taskC`, `taskD` and `taskE`

5. The subfolder taskA should contain the Azkaban.zip and A.txt

6. The subfolder taskB contains Map.java

7. The subfolder taskC should contain producer_consumer.java

8. The subfolder taskD should contain the Prince.java

9. The subfolder `taskE` should contain two folders `task0` and `Extra-Credit`. The folder `task0` should contain `DrawCircleInGUI.java` and any other relevant files. The folder `Extra-Credit` should contain `FitCircleInGUI.java` and `CircleFitter.java` and any other relevant files.

```
lab09_groupXY_outlab
├── taskA
│   ├── Azkaban.zip
│   └── A.txt
├── taskB
│   └── Map.java
├── taskC
│   └── ProducerConsumer.java
├── taskD
│   └── Prince.java
├── taskE
│   ├── task0
│   │   └── DrawCircleInGUI.java
│   └── Extra-Credit
│       ├── FitCircleInGUI.java
│       └── CircleFitter.java
└── readme.txt
```

How We will Grade You [70 + 10 Marks]

Extra credit points are available to you only if you get the basics right.

1. Task A [**10 Marks**]
2. Task B [**15 Marks**]
3. Task C [**15 Marks**]
4. Task D [**10 Marks**]
5. Task E [**20 + 10 = 30 Marks**]
   - Displaying GUI with necessary controls: 4 Marks
   - Drawing the circle on entering necessary values: 4 Marks
   - Sanity checks on input values: 4 Marks
   - Code should work on window damage (resize, etc.): 8 Marks
   - **Extra Credit:** 10 Marks
     - Displaying GUI with necessary controls: 2 Marks
     - Display of contour on mouse drag: : 2 Marks
     - Display of fit circle and dragged contour: 2 Marks
     - Display co-ordinates of mouse position on hover in top part: 2 Marks
     - Color of circle and contour: 2 Marks