Assignment 3 Scenery

Due: May 17th, 2020 (6am) **COMP 110 Object-oriented Programming**

In this assignment, you are asked to draw a scenery, consisting of buildings and trees. Scenery details such as building and tree information is provided to you in an input text file. Sample file contents and the drawn scene are shown in Figure 1. In the text file, buildings have the following information at each line: floor count, x center, width of the building and color. For trees, only the x coordinate is provided in the input text file.

Buildings and trees should be modeled as classes. You need to write Building and Tree classes, as shown in the UML class diagrams (See Figure 2). You should store all buildings in buildings array list, of type Building class and store all trees in a tree array list, of type Tree class.

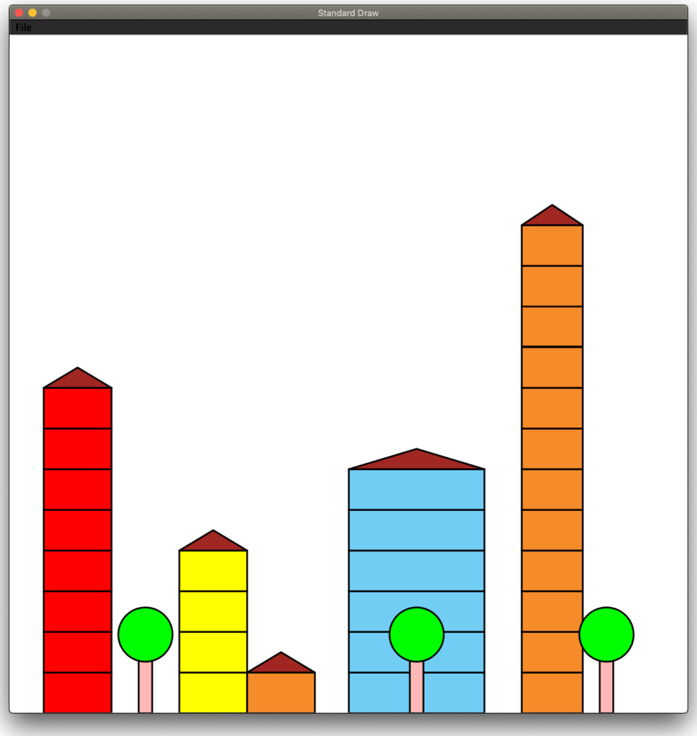
In the main method, you should use loadData and plotScene methods to load and draw the scene. Building class should have drawBuilding method and the Tree class should have the drawTree method. You should invoke these methods to draw these scene elements. Your program should output only one drawing at a time which is specified by its data file, e.g., data1.txt.

Requirements

- Write Building and Tree classes. See the UML diagrams shown in Figure 2.
- Store buildings and trees in buildings and trees array lists, respectively, as shown in the main method template.
- Main method should be written exactly as given below.
- Write plotScene and loadData methods with the input/output parameters as given in the template.
- Use for-each loop to draw scene stored in array lists, in the plotScene method. Do not use standard for loop in the plotScene method.
- Use polygon and filledPolygon to draw triangles for roofs (see graphics library documentation).
- Use the following scene measurements to draw buildings and trees: Floor height: 0.06, Roof height: 0.03, Tree body height: 0.08, Tree body width: 0.02, Tree radius: 0.04
- Use the following colors to draw buildings: RED, YELLOW, GREEN, BOOK_LIGHT_BLUE, and PRINCETON_ORANGE

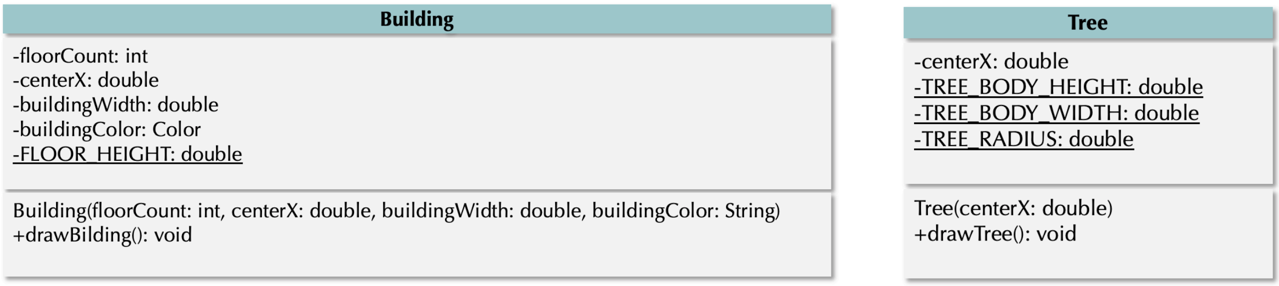
In your report, provide the output scenery images for the two data files provided to you (see Figure 1). Additionally, create a new scenery file yourself (data3.txt) and provide the drawing in your report.

Tip for reading double type values from a text file: If you encounter errors while reading double types, replace dots with commas: e.g., convert 0.6 to 0,6. In some Windows systems, decimal separator might be different.



```
Building;8;0.1;0.1;Red
Building;4;0.3;0.1;Yellow
Tree;0.6
Building;1;0.4;0.1;Orange
Tree;0.88
Building;6;0.6;0.2;Blue
Building;12;0.8;0.09;Orange
Tree;0.2
```

Figure 1. Scenery example given in data1.txt input file. File contents are shown on the right. Buildings have the following information at each line: floor count, x center, width of the building and color. For trees, only x coordinate is provided.



Building

-floorCount: int
-centerX: double
-buildingWidth: double
-buildingColor: Color
-FLOOR_HEIGHT: double

Building(floorCount: int, centerX: double, buildingWidth: double, buildingColor: String)
+drawBilding(): void

Tree

-centerX: double
-TREE_BODY_HEIGHT: double
-TREE_BODY_WIDTH: double
-TREE_RADIUS: double

Tree(centerX: double)
+drawTree(): void

Figure 2. UML class diagrams for building and tree classes. Note that underscored variables have a special meaning in Java.

Main Method Template (Should be exactly the same in your program)

```
public static void main(String[] args) throws FileNotFoundException  {

   String filename = "data1.txt";        // input filename

  // store scene elements in array lists
  ArrayList<Building> buildings = new ArrayList<>();
  ArrayList<Tree> trees = new ArrayList<>();

   loadData(filename, buildings, trees); // load scene elements from the input file
   plotScene(buildings, trees);          // plot scene elements
}
```

Evaluation Criteria and Grading for Assignments

Code

20% Compliance to submission rules and programming style, e.g., naming conventions, indentation, comments.
70% Correctness of the solution

Report

10% Completeness of the report, compliance to the report format, correctness of the content and language.

Submission Guide

Submission Files

Submit a single compressed (.zip) file to Blackboard.
Name your zip file as name_surname.zip.
Zip file should contain all source codes (under the \code directory), and report (in PDF format, under the \report directory).
Name the main code which is used to run your assignment as name_surname.java.
Name your report as name_surname.pdf.
Contents of each Java file should start with your name, student ID, date, and a brief code summary in a Javadoc style comment block.

Mandatory Submission

Submission of assignments is mandatory. If you do not submit an assignment, you will fail the course.

Late Submission Policy

Maximum submission delay is two days. Late submission will be graded on a scale of 50% of the original grade.
Submission is mandatory even if you submit your assignment late.

Plagiarism

Plagiarism leads to grade F and YÖK regulations will be applied