© Manfred Kerber
Alexandros Evangelidis

School of Computer Science
University of Birmingham

Worksheet 5

MSc/ICY Software Workshop

Assessed Exercise: 12.5% of the continuous assessment mark.

Submission: Thursday 3 December 2015 2pm

5% late submission penalty within the first 24 hours. No submission after 24 hours.
JavaDoc comments are mandatory. Follow the submission guidelines on
`http://www.cs.bham.ac.uk/internal/courses/java/msc/submission.php`.

Exercise 1: (Basic, 30%) Write a class `FloorLayout` with the only field variable `private ArrayList<Polygon> toDraw;` that allows to produce a rectangular layout of a floor. In detail write:

- A constructor `public FloorLayout(int width, int height)` that puts a rectangle of corresponding size on the panel (10 pixels away from the corner in both directions) and initializes the field variable.
- A method `public void addToDraw(Polygon poly)` that puts pieces of furniture in form of a polygon on the panel (that is, that adds to the `toDraw` field variable, which in turn is used to put the corresponding pieces on the floor).
- A method `public void addRectangle(int xPos, int yPos, int dX, int dY)` that converts the rectangle to a polygon and adds it to the `toDraw` ArrayList.
- Write a `main` method that puts at least three pieces of furniture on the floor layout which do not overlap.

Exercise 2: (Basic, 20%) In a computer game Christmas trees should be put in the background of a scene. Your task in this and the following exercises is to build corresponding Christmas trees that come with some variation.
The simplest type of tree has the shape of a (green) triangle plus a (brown) rectangle, the trunk. The brown trunk has a size of 2 units times 4 units, the width of the triangle at the bottom is 12 units, its height 12 units, as displayed in Figure 1. Trees can be re-sized by a scaling factor of type `int` which is the same for the `x` and for the `y` coordinate.

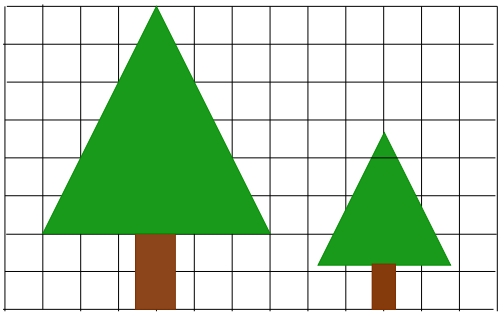

Figure 1: Two simple Trees

Write a class `Trees` to draw simple Christmas trees as in Figure 1, each made up of one green triangle and one brown rectangle. The constructor `Trees(int[] xTrees, int[] yTrees, int[] scaleTrees)` takes three `int` arrays which store the `x`- and `y`-values of the bounding boxes of the trees as well as their scales, respectively. (E.g., a tree with `scale` 2 is twice as tall and wide as a tree of `scale` 1). Use the `fillPolygon` method to draw your trees. Put at least 5 trees on a panel to display them in a main method.

Exercise 3: (Medium, 20%)

Write a class `TwoTierTrees` (extending your existing `Trees` class from the previous exercise) to draw two-tier trees. Each tree would come with a heptagon to represent the foliage as in Figure 2. The trees should still scale, and the constructor should be as in the previous exercise. Put at least 5 trees on a panel to display them in a main method.

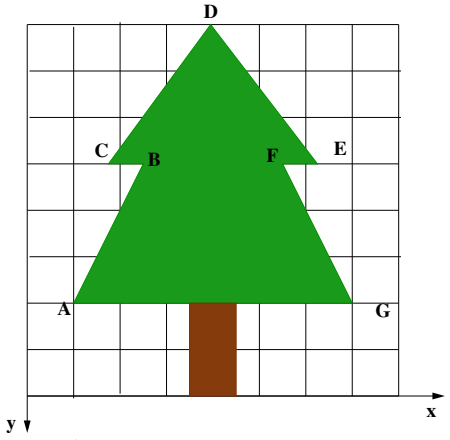

Figure 2: A two-tier tree

Exercise 4: (Advanced, 15%)
Write a class `BaubleTrees` (extending your existing `Trees` class) to draw one-tier trees with the same number of circular baubles on them. Write a constructor `public BaubleTrees( int[] xTrees, int[] yTrees, int[] scaleTrees, int numberOfBaubles)`.
Baubles should be inside (or at least overlap with) the green area of a tree. Baubles should be one of three colours (red, blue, or yellow), randomly chosen per bauble. Put at least 5 trees on a panel to display them in a main method.

Exercise 5: (Advanced, 15%) Write a class `StarTrees` that extends `BaubleTrees` so that each star tree has a star at its top (as described, e.g., at `https://en.wikipedia.org/wiki/Star_polygon`). A star is characterized by the number of vertices and the number of steps to come to the next vertex. Stars should be drawn correctly for the cases where the number of vertices and the steps do not have common divisors. Write a constructor `public StarTrees(int[] xTrees, int[] yTrees, int[] scaleTrees, int numberOfBaubles, int numberOfVertices, int steps)`. Put at least 5 trees (with stars that have 11 vertices and step size 4 on a panel to display them in a main method (see, e.g., Figure 3).

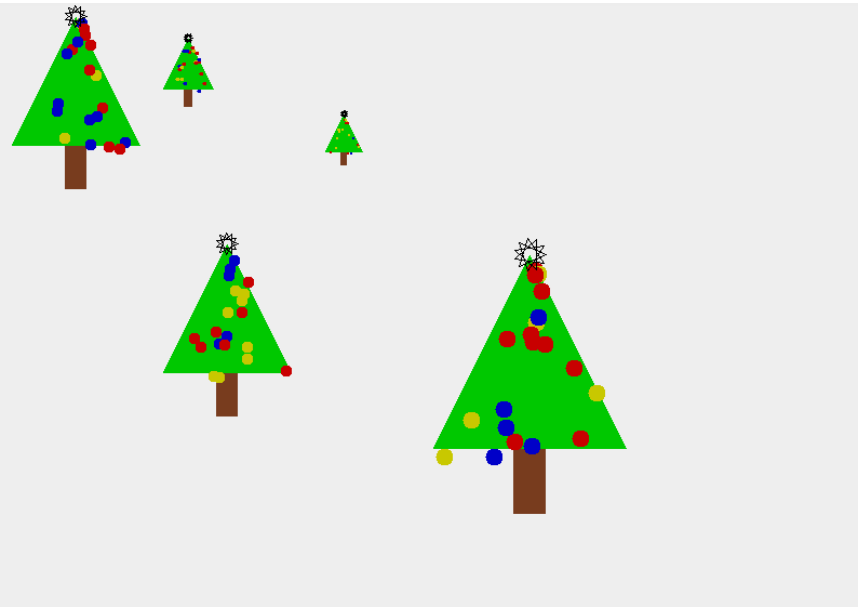

Figure 3: 5 `StarTrees`. Each star has 11 vertices and steps 4