Objective: To use recursion in problem solving. The first question and the second question have to be completed. Question three is a bonus question. You do not have to answer question three, but you will receive extra marks if you do.

1 Base 8 representation

Write a program to convert a non-negative integer decimal number with based 10 to the representation of base 8. Base 8 representation uses only 8 numbers 0 – 7. For example, the base 8 representation of number 10 is 12 (since $1*8 + 2 = 10$). To obtain the base 8 representation of a decimal number, we use the following recursive procedure

```python
if the number is less than 8
    output the number
else
    output the base 8 representation for the number divided by 8
    output the mod of the number with 8
```

Write a recursive program to implement the base 8 conversion based on the above procedure.

2 Sierpinski Triangle

Write a recursive program to draw the Sierpinski triangle. Fig. 1 shows Sierpinski triangles with different orders. As shown in Fig. reffig-triangle (a), the first order Sierpinski triangle is just a common triangle. Sierpinski triangle can be constructed recursively: the order n triangle is composed of three (n-1) triangle whose three outer vertices are from the points of three outer vertices and three middle edge points of the order n triangle.

![Figure 1: Sierpinski Triangles.](image)

(a) Order 1.  (b) Order 2.  (c) Order 3.  (d) Order 8.

Use StdDraw class in your implementation.
3 Recursive Tree (Bonus Question)

Using Recursion, we can construct trees as shown in Fig. 2. Random factors are included so that each time you run the program it will generate a different tree.

There are different methods to generate a tree. The following method is just one way of growing it. The tree grows from a point. It grows a trunk along a direction with a certain length. Then, the tree splits into two growing directions at the end of the trunk.

To make the tree look like a real tree, we make the sub-branches shorter and thinner. This can be achieved by multiplying a factor in (0, 1) to the branch length and thickness of the previous level. The factor can be chosen something like $0.95 + \delta$, where $\delta$ is a small random number.

The two branching directions are determined by 2 random angles $\theta_1$ and $\theta_2$, so that, if the trunk growing direction is $\alpha$ the two branches will follow the direction of $\alpha + \theta_1$ and $\alpha + \theta_2$. Use small $\theta_1$ and $\theta_2$ if you want the tree to grow mostly upwards. The last step is to draw a red “leave” at each branch terminal.

![Recursive Tree](image)

Figure 2: Recursive Tree.

4 What to Submit

You should submit the Java programs. Pay attention to good Java programming style. Upload your Java files to webCT before the submission deadline. There will be 3-day grace period. But late submission would involve 10% point deduction for each day. Submissions later than 3 days are not accepted.