Object-Oriented Programming
Second Session, 2016

Notes or documents of any kind forbidden. Duration: 3 1/2h. Please answer all questions on separate sheets labelled with your name and section.

1. For a Java project involving the manipulation of 2D geometrical objects, one needs to define:

   - a class `Point2D` for representing points in the 2D plane, i.e., pairs (x, y) of coordinates.
   - a class `Rectangle2D` for representing rectangles that have their sides parallel to the axes of the coordinate system.

   These classes should satisfy the following properties:

   - One should be able to modify at all times the coordinates of a point or the corners of a rectangle.
   - Only nonnegative integer coordinates are allowed for points and rectangle corners. Any operation resulting in a negative coordinate must report an error.
   - One should be able to clone points and rectangles, as well as to check whether two such objects are equivalent (i.e., whether they both have the same coordinates or the same four corners).
   - Both classes should implement the following interface, defining an operation for adding a translation vector `(trX, trY)` to the coordinates of a point or of the four corners of a rectangle:

     ```
     package org.acme.geometry;
     public interface Translatable2D
     {
       void translate(int trX, int trY);
     }
     ```

   Provide Java source code for the `Point2D` and `Rectangle2D` classes. You are free to create as many additional classes as you want. All errors should be reported by means of custom exceptions. Any detail that is not mentioned in this problem statement can be freely interpreted.

2. Consider the following Java code.

```
public class Pizza
{
  public static final String name = "Base";
  public void displayIngredients()
  {
    System.out.println("Tomato");
  }
}

public class MargheritaPizza extends Pizza
{
  public static final String name = "Margherita";
  public void displayIngredients()
  {
    super.displayIngredients();
    System.out.println("Cheese");
  }
}

public class HawaiianPizza extends MargheritaPizza
{
  public static final String name = "Hawaiian";
  public void displayIngredients()
  {
    super.displayIngredients();
    System.out.println("Ham");
    System.out.println("Pineapple");
  }
}
```

For each of the three following sequences of Java instructions (1, 2 and 3), answer these two questions:

(a) Would this sequence compile successfully, assuming that it is placed in the body of a suitable method?

(b) If your answer to (a) is positive, what would the execution of this sequence display? Otherwise, how can these instructions be corrected?

(Please justify thoroughly all answers.)

1.
```
Pizza p = new HawaiianPizza();
p.displayIngredients();
System.out.println(p.name);
```

2.
```
HawaiianPizza hp = new HawaiianPizza();
hp.displayIngredients();
System.out.println(hp.name);
```

3.
```
MargheritaPizza mp = new Pizza();
```

3. Consider the following Java class.

```
public class Counter
{
  private int value = 0;
  public void increment()
  {
    ++value;
  }
  public int getValue()
  {
    return value;
  }
}
```

(a) Show that this class is not thread-safe, by giving an example of execution scenario leading to a problematic situation.

(b) Modify this class definition in order to make it thread-safe.

(c) Write a complete Java program that instantiates once the `Counter` class, and then creates 10 threads that each invoke 20 times the `increment()` method of the corresponding object.

(Please justify thoroughly your answers to (a) and (b).)