Exam
DIT948: Introduction to Programming

Date:	2014-12-12
Time	8:30–12:30
Place	Lindholmen
Teacher and examiner	Cezar Ionescu
Questions	+4915127113484 first visit around 9:30
Results	will be posted within 15 working days
Grades	Pass (G) 50p, Pass with Honors (VG) 80p
Allowed aids	Any book on Java programming is allowed Any English dictionary No electronic translators allowed
Reviewing:	2015-02-13, 10:00–12:00, Lindholmen Second floor room (note: subject to change, changes will be posted on the course website).

Please observe the following:

- Write in legible English (illegible translates to "no points"!).
- Motivate your answers, and clearly state any assumptions made.
- Code or text copied from the book will not be rewarded. Your own contribution is required.
- Start each task on a new sheet!
- Write on only one side of the paper!
- Before handing in your exam, number and sort the sheets in task order!
- Write your anonymous code and page number on every page!

Not following these instructions will result in the deduction of points!

1. [10pts] Consider the class:

```
package Exam;

public class A {
    private int a;
    protected int b;
    int c;

    public int foo() {
        return a;
    }

    protected int bar() {
        return c;
    }
}
```

Class `B` is a subclass of class `A`:

```
 1  package Exam;
 2
 3  public class B extends A {
 4      public void test() {
 5          System.out.println(a);
 6          System.out.println(b);
 7          System.out.println(c);
 8          System.out.println(foo());
 9          System.out.println(bar());
10      }
11  }
```

Which of the lines 5–9 will cause a compilation error and why? How does the answer change if `B` is in a different package from `A` (for example, if we replace the first line with `import Exam.A;`)?

2. [10pts] Your boss has written a method that was supposed to return the first index at which a given integern `n` appears in a given array `a`, and to return `-1` if `n` does not appear in `a`.

   For example:

```
a = {1, 2, 2} n = 2   =>  firstIndex(a, n) =  1
a = {2, 2, 2} n = 2   =>  firstIndex(a, n) =  0
a = {1, 2, 3} n = 4   =>  firstIndex(a, n) = -1
```

   Unfortunately, things didn't really work out that way: in fact, the method doesn't even compile! Still, it's your boss's program, you can't just throw it away and write a new one. Make the changes necessary for the method to compile and return the correct result.

```
public static int firstIndex(int a[], int n) {
    for (int i = 0; i <= length; i++)
        if (a == n)
            n = i;
    return n;
}
```

3. [10pts]

   Given the following classes defined in separate files:

```
abstract class Polygon {
  int nrSides;

  abstract public void printDescription();

} // end class Polygon



class Rectangle extends Polygon {

  Rectangle() {
    nrSides = 4;
  }
```

```
  public void printDescription() {
    System.out.println("Four sides");
  }
} // end class Rectangle

class Square extends Rectangle {

  public void printDescription() {
    System.out.println("Four sides, all equal");
  }
} // end class Square

public class Test {
  public static void main (String args []) {
    Polygon p1, p2;

    p1 = new Rectangle();
    p1.printDescription(),

    p2 = new Square();
    p2.printDescription();

    Square square
    Rectangle rectangle;

    square = new Rectangle();
    square.printDescription();

    rectangle = new Square();
    rectangle.printDescription;
  }
} // end class Test
```

Which lines do you need to comment out in order to get the class `Test` to compile? Once you do that, what will be printed on the screen when running the class `Test`?

4. [30pts]

   Consider the following canvas made up of pixels:

```
    0  1  2  3  4  5  6  .  .  .
0   .  .  .  .  .  .  .
1   .  .  .  .  .  .  .
2   .  a  .  .  .  .  .
3   .  .  .  .  .  .  .
4   .  .  .  .  .  b  .
5   .  .  .  .  .  .  .
6   .  .  .  .  .  .  .
.
.
.
```

Each pixel has two coordinates: a horizontal one, and a vertical one. The coordinates are integers. For example, the pixel denoted with `a` in the figure above has the horizontal coordinate 1, and the vertical coordinate 2. The pixel denoted with `b` has horizontal coordinate 5, and vertical coordinate 4.

The canvas extends indefinitely both horizontally and vertically.

You are given the following interface which is implemented by objects that can be drawn on the canvas:

```
public interface Drawable {
  public void draw();
}
```

You are also given the following class, which draws a line between two pixels given by their horizontal and vertical coordinates:

```
public class Line implements Drawable{

  // The coordinates of the end points
  private int x1, y1, x2, y2;

  // Constructor
  public Line(int x1, int y1, int x2, int y2) {
      this.x1 = x1;
      this.y1 = y1;
```

```
        this.x2 = x2;
        this.y2 = y2;
    }

    // Draws a line from (x1, y1) to (x2, y2)
    public void draw() {
       // ... not shown ...
    }
}
```

For example, the code

```
Line line = new Line(1, 2, 5, 4);
line.draw();
```

draws a line between the points `a` and `b` above.

In this exercise you are asked implement to implement three classes which are part of an API for drawing polygons. You will do this by "filling in" the data and method definitions. You don't need to do any error checking or write comments, unless specifically asked to do so.

Do not write "between the lines"! On your paper, clearly mark out which code is supposed to go where. For example, the code supposed to fill in the block `A`, the member variables of the class `Triangle`, should appear as

```
+------------------ A ------------------+
|  private int x1, y1, x2, y2, x3, y3;  |
+---------------------------------------+
```

and not squeezed on the page with the exam subjects!

Important: failure to clearly mark the blocks will result in deduction of points!

Remarks:

- Since block A has been filled in for you above, there are seven blocks left for you to fill in for this exercise: B, C, D, E, F, G, H. Block D is worth 2pts, blocks B and E are worth 4pts each, the rest, blocks C, F, G, and H are worth 5pts each.

- When in doubt, make a drawing in order to get the coordinates of the points right. For example, the lower right-hand corner of the rectangle has the horizontal coordinate `x1 + width` and the vertical coordinate `y1 + height`.
- Do not try to be too clever in blocks C and E. A solution that constructs the needed edges and draws them will be given full points, even though it might seem somewhat repetitive. A "clever" solution using loops which does not work will **not** get full point. Of course, an elegant solution which does work would be preferrable, but it will get no extra points.

```
public class Triangle implements Drawable {
    // Block A
    // private member variables:
    // the coordinates of the vertices


    // Constructor
    public Triangle(int x1, int y1, int x2, int y2, int x3, int y3) {
        // Block B
        // Initialize the member variables to the given values
    }

    public void draw() {
        // Block C
        // Use the Line class to create and
        // draw the edges of the triangle
    }
}


public class Rectangle implements Drawable {
    // Block D
    // private member variables:
    // the coordinates of the upper left-hand corner,
    // the width and the height of the rectangle

    // Constructor
    public Rectangle(int x1, int y1, int width, int height) {
        // Block E
```

```
            // Initialize the member variables to the given values
        }

        public void draw() {
            // Block F
            // Use the Line class to create and
            // draw the edges of the rectangle
        }
    }


    public class Square extends Rectangle {
        // No member variables

        // Constructor
        public Square(int x1, int y1, int length) {
            // Block G
            // Use the constructor of the superclass to
            // construct a square with the given upper left-hand
            // corner and edges of the given length
        }

        public void draw() {
            // Block H
            // Use the draw method of the superclass
            // to draw the edges of the square
        }
    }
```

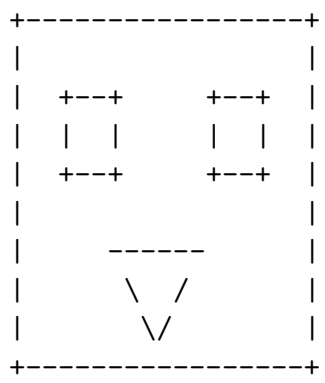
5. [20pts]

   In this exercise, we refer to the classes from above, but do not require you to have implemented them (only that you understand how to create and use instances of them).

   Write a method with the signature

   ```
   public void drawHappy(int x, int y)
   ```

   to draw the following depiction of a "happy face":



The arguments `x` and `y` represent the horizontal and vertical coordinates of the upper left-hand corner of the drawing. The large rectangle has width 18 and height 20. The "eyes" are squares, with edges of length 5. The uppper left-hand corner of the left eye has horizontal coordinate `x+3` and vertical coordinate `x+5`. The right eye is symmetrically placed. The lowest vertex of the mouth has coordinates `x+9` and `y+20`, respectively. The upper left-hand corner of the mouth is at `x+8` and `y+15`, and the right-hand corner is symmetrically placed.

Each correctly constructed and drawn element of the "happy face" will bring you 5pts.

6. [20pts] (Adapted from *CodingBat*)

   Write a subroutine with the signature

   ```
   public int[] replaceZero(int[] nums)
   ```

   taking as input an array of integers `nums`, and returning a new array of integers in which every zero in `nums` is replaced by the largest integer to the right of that zero in the array. If the last element of the array is a zero, then it is not replaced (after all, there is nothing to its right in the array).

   For example:

   ```
   replaceZero({7, 0, 1})       ==>  {7, 1, 1}

   replaceZero({0, 8, 1, 0, 7}  ==>  {8, 8, 1, 7, 7}

   replaceZero({0, 1, 2, 0})    ==>  {2, 1, 2, 0}

   replaceZero({0, 0, 0})       ==>  {0, 0, 0}
   ```

In the first example, there is one zero at index 1, and it is replaced with the largest element with index larger than 1, which is, well, 1 (at index 2).

In the second example, there are two zeros. The first is at position 0, and it gets replaced with the largest element with index larger than 0, which is 8 (at index 1). The second 0 is at index 3, and it is replaced by the largest integer to the right, which is 7 (at index 4).

In the third example, the last element in the array is a zero, which therefore does not get replaced.

In the last example, the first zero gets replaced with the maximal integer to its right, which is zero. The second zero gets replaced with the maximal integer to its right, which is zero. The last zero does not get replaced, since it is the last element of the array.

Make sure to check the corner cases, e.g., when the array `nums` is empty!