Name and, if possible, ID#:_____

AMERICAN UNIVERSITY OF ARMENIA

College of Science and Engineering

COMP120 Introduction to Object-Oriented Programming

FINAL EXAM

8/10

Date:	Monday, May 18 2015
Starting time:	09:20
Duration:	1 hour 40 minutes
Attention:	**ANY TYPE OF COMMUNICATION IS PROHIBITED**

Please write down your name at the top of all used pages

Problem 1

Consider below a ***public interface Valuable*** that includes the only method ***public double value(double x)***:

```
public interface Valuable {

      public double value(double x);
}
```

1.1 Implement a ***public class Function*** that encapsulates a member variable of type ***Valuable*** and computes its derivative at the specified point ***x*** using the approximation:

$$f'(x) \approx \frac{f(x+dx)-f(x-dx)}{(2*dx)}$$

```
public class Function {

      private Valuable f;
      private double dx;

      public Function(Valuable newValuable, double newDX) {
            //TO BE IMPLEMENTED
      }

      public double derivative(double x) {
            //TO BE IMPLEMENTED
      }
```

1.2 Implement an expression

exp(-a * (x – c)²)

as a ***public class Gauss*** that implements the interface ***Valuable*** and encapsulates double parameters ***a*** and ***c***. The parameters are initialized by the two-argument constructor ***public Gauss(double newA, double newC)***;

1.3 In a separate ***public static void main(String args[])*** write a code that inputs two double values, creates an object of type ***Gauss*** and, using the class ***Function***, prints the value of its derivative at the ***x = 1.0*** point:

```
public static void main(String args[]) {
      Scanner input = new Scanner(System.in);
      double a = input.nextDouble(), c = input.nextDouble();

      //TO BE COMPLETED

}
```

```
public Function (double newValuable, double newdX)
    {        dx = newdX
             f = nevValuable
    }

public  derivative (double X).
    {  ~~double newFunction~~.int function = new.inputnextInt();
   function = (Function.f(dX +X). - Function.f( X-dX ))/ (2*dX)
     }
```

1.2.

```
public   Gauss ( double ~~newA~~, double ~~defo~~C)          exp(-a * (X-c)^2)
 {  double a = input.nextDouble();
    double c= input.nextDouble();
    doble  exp = input.nextDuble();

  exp = Math.exp.(-a .(dX. -c)(dX-c);
       }
```

4

1.3

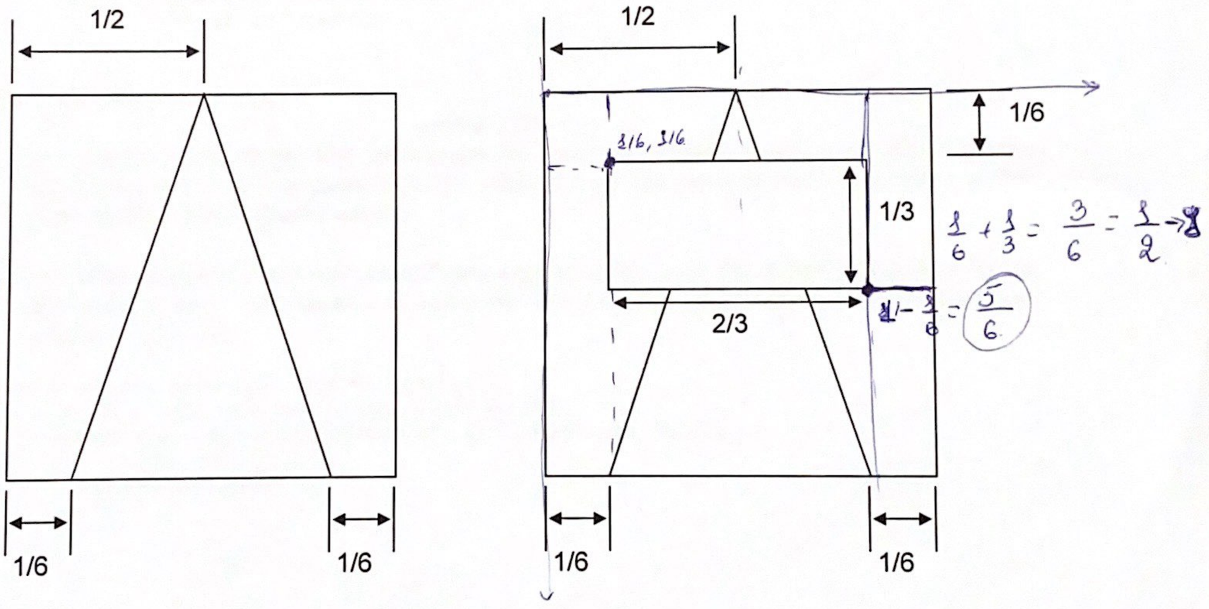
Name and, if possible, ID#:_____

Problem 2

All 6 types of chess pieces can be drawn based on simple sketches consisting of a triangular base and rectangular cap. Consider below a ***public class ChessPiece*** that implements the triangular base only. Its geometry relative to the unit size of the square field is also sown:

```
public class ChessPiece {

        private Rectangle field;
        private Polygon base;

        public ChessPiece(int size) {
                field = new Rectangle(size, size);
                base = new Polygon(); //initially empty polygon
                base.addPoint(size / 6, size); //left vertex of the base
                base.addPoint(5 * size / 6, size); //right vertex of the base
                base.addPoint(size / 2, 0); //top vertex of the base
        }

        public void drawBase(Graphics g) {
                g.drawRect(field.x, field.y, field.width, field.height);
                g.drawPolygon(base);
        }

        public void drawCap(Graphics g) {
        }

        public void draw(Graphics g) {
                g.drawBase(g);
                g.drawCap(g);
        }
}
```

Extend a ***public class Rook extends ChessPiece*** that encapsulates ***Rectangle cap*** member variable. Implement the constructor and override ***public void drawCap(Graphics g)***. The geometries of the general chess piece and the rook are shown below:


1/2
1/6
1/6
1/2
1/6
1/6, 1/6
1/3
2/3
$\frac{1}{6}+\frac{1}{3}=\frac{3}{6}=\frac{1}{2}\to 8$
$1-\frac{1}{6}=\frac{5}{6}$
1/6
1/6

Use the backside, if needed

OOP.FT.180515.M117

```
public class Rook extends ChessPiece{
     private Rectangle cap;

Public Rook (intsize) {
     ~~[illegible]~~    super( );
      cap = newRegtcengle()
     cap.addpoint(size/6, size/6); // top-right hand corner
     cap.addpoint(size/2, 5*size/6. ); //bottom-left hand corner
     }

public void drawCap(Graphics g)
    {
       g.drawRegt (cap.~~1~~, cap.2)
       }
```

4