Name and, if possible, ID: ___ / lan

AMERICAN UNIVERSITY OF ARMENIA
College of Science and Engineering
COMP120 Introduction to Object-Oriented Programming

FINAL EXAM

2/15

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 max in the specified range from x_1 to x_2 by looking at:

$f(x_1), f(x_1+dx), f(x_1+2dx), \ldots, f(x_1+k*dx)$, where $k = 1, 2, \ldots$ and $x_1+k*dx < x_2$

```
public class Function {

      private Valuable f;
      private double dx;

      public Function(Valuable newValuable, double newDX) {     int K [100] ;
            //TO BE IMPLEMENTED                                 if ( K >0 ; x,+K*dx<x2;
      }                                                                        K++)

      public double max(double x1, double x2) {                 cout<< f(x,+x*dx)
            //TO BE IMPLEMENTED
      }
```

1.2 Implement an expression

$$a * \sin(x) + b * \cos(x)$$

as a ***public class Harmonic*** that implements the interface ***Valuable*** and encapsulates double parameters ***a*** and ***b***. The parameters are initialized by the two-argument constructor ***public Harmonic(double newA, double newB)***; Privcle Valueble a ; Private Valuable b, double a ; double b ;

1.3 In a separate ***public static void main(String args[])*** write a code that inputs two double values, creates an object of type ***Harmonic*** and, using the class ***Function***, prints its maximal value in the range from $x_1 = -1.5$ to $x_1 = 1.5$: a*(sin x) + b*(cos x

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

      //TO BE COMPLETED     if (x >-1.5 && x≤1,5 )
                                   cout<< a*sin(x) + b*(cos x)
}
```

1

OOP.FT.180515.L120

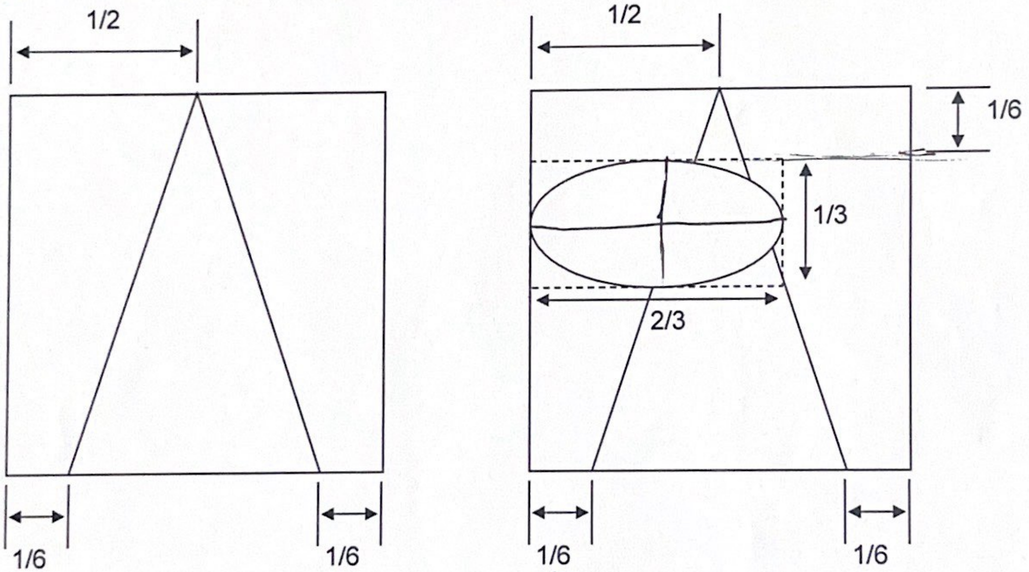
Name and, if possible, ID#:_____

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 Knight 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 knight are shown below:



1

Publrc clas, kinght {

Public void triangulre {

base.addPont(size/6);

Bass.addPont (5·size/6);

beze. add Pont (size /2);

Use the backside, if needed

OOP MT 180515 L120

```
size/6 * size/12 && 5*size/6 * size/12
          cout << triangular;
```