Q1

10 Points

```
// Shapes.java
package shapes;
```

```
class Point {
  private double x; private double y;

  public Point(double x, double y) {
    this.x = x; this.y = y;
  }

  public double getX() { return x; }
  public double getY() { return y; }
}
```

```
abstract class Shape {
  public static double PI = 3.14;

  private Point location;

  public Shape(Point location) {
    this.location = location;
    System.out.println("Shape()");
  }

  public void rotate() {
    System.out.println("Shape.rotate()");
  }
  public Point getLocation() { return location; }
  public void changeLocation(Point location) {
    this.location = location;
  }

  abstract public void draw();
  abstract public double calculateArea();
}
```

```
class Circle extends Shape {
  private double radius;

  public Circle(Point location, double radius) {
    super(location);
    this.radius = radius;
    System.out.println("Circle()");
  }

  public void draw() {
    System.out.println("Circle.draw()");
  }
  public double calculateArea() {
    System.out.println("Circle.calculateArea()");
    return PI * radius * radius;
  }
  public double getRadius() { return radius; }
}
```

```
class Rectangle extends Shape {
  private double width; private double height;

  public Rectangle(Point location,
                   double width, double height) {
    super(location);
    this.width = width; this.height = height;
    System.out.println("Rectangle()");
  }

  public void draw() {
    System.out.println("Rectangle.draw()");
  }
  public double calculateArea() {
    System.out.println(
      "Rectangle.calculateArea()");
    return width * height;
  }
  public double getHeight() { return height; }
  public double getWidth() { return width; }
}
```

```
class Square extends Rectangle {
  public Square(Point location, double length) {
    super(location, length, length);
    System.out.println("Square()");
  }

  public double getLength() { return getHeight(); }
}
```

```
class Triangle extends Shape {
  private double baseLength; private double height;

  public Triangle(Point location,
                  double baseLength,
                  double height) {
    super(location);
    this.baseLength = baseLength;
    this.height = height;
    System.out.println("Triangle()");
  }

  public void draw() {
    System.out.println("Triangle.draw()");
  }
  public double calculateArea() {
    System.out.println("Triangle.calculateArea()");
    return baseLength * height / 2;
  }
  public double getBaseLength() {
    return baseLength;
  }
  public double getHeight() { return height; }
}
```

```
public class Shapes {
  public static void main(String[] args) {

    Point point = new Point(0.0, 0.0);

    Circle circle = new Circle(point, 2.0);
    Rectangle rectangle =
      new Rectangle(point, 3.5, 2.0);
    Square square = new Square(point, 4.0);
    Triangle triangle =
      new Triangle(point, 2.0, 6.0);

    System.out.println("-----");

    System.out.println(circle.calculateArea());
    System.out.println(rectangle.calculateArea());
    System.out.println(square.calculateArea());
    System.out.println(triangle.calculateArea());

    System.out.println("-----");

    circle.rotate();
    rectangle.rotate();
    square.rotate();
    triangle.rotate();
  }
}
```

Answer all questions according to the codes contained in the Shapes.java file given on this page.

Indicate whether each statement below is True or False by typing T or F in the boxes.

Q1.1

2 Points

Rectangle HAS-A Square (T/F)

F

Q1.2
2 Points

Shape HAS-A Point (T/F)

F

Q1.3
2 Points

Square IS-A Rectangle (T/F)

T

Q1.4
2 Points

Circle IS-A Shape (T/F)

T

Q1.5
2 Points

Point IS-A Shape (T/F)

F

Q2
14 Points

Assume we have three classes: **Person**, **Teacher**, and **Student**.**Teacher** and **Student** are both subclasses of **Person**. Which of the following assignments are legal? Please select correct one for each assignment.

```
Person p;
Teacher t;
Student s;
```

Q2.1
2 Points

`t=new Teacher();`

- (●) legal
- ( ) illegal

Q2.2
2 Points

`p=t`

- ( ) legal
- (●) illegal

Q2.3
2 Points

`s=(Student) t;`

- ( ) legal
- (●) illegal

Q2.4
2 Points

`s=(Student) p;`

- ( ) legal
- (●) illegal

Q2.5

2 Points

```
p=new Student();
```

- (●) legal
- ( ) illegal

Q2.6

2 Points

```
t=new Person();
```

- ( ) legal
- (●) illegal

Q2.7

2 Points

```
t=(Teacher) p;
```

- ( ) legal
- (●) illegal

Q3

25 Points

Consider the following class hierarchy where Class **Car** is the supper class and the classes **ClassicCar** and **SportCar** are two subclasses derived from **Car**. Class **CarExhibition** contains a filed of type ArrayList that stores objects of type **Car**.

After the ..... operations in the program section below, write the java code for `public double getTotalPrice()` function. **CarExhibition** has cars of different types stored in an arraylist and **getTotalPrice** method that returns the total prices of all cars in the exhibition.

```
//Car Class Definition
abstract public class Car {
        protected double price;
        protected  int year;
        public Car(double price, int year)

                this.price = price;
                this.year = year;
        }
        public String toString()
        {
                return ("Price = "+price+" Year = "+year);
        }
        public abstract double  calculateSalePrice ( );
//abstract method
}
```

```
// CarExhibition Class Definition
import java.util.ArrayList;
import java.util.Iterator;
public class CarExhibition {
        private ArrayList cars;
        public CarExhibition(){
                cars = new ArrayList();
        }
        public void addCar (double price, int year){
                //Superclass/subclass relationship
                Car cr = new ClassicCar(price, year);
                cars.add(cr);
        }
        public void addSportCar (double price, int year){
        cars.add(new SportCar(price, year));
        }
        public double getTotalPrice()
        {
                ..............
                ..............

        }
```

```
//SportCar Class Definition
public class SportCar extends Car {
        public SportCar(double  price, int year){
                super (price, year);
        }
        public double  calculateSalePrice ( ) {
                double salePrice;
                if (year > 2000)
                        salePrice = 0.75 * price;
                else if (year > 1995)
                        salePrice = 0.5 * price;
                else
                        salePrice = 0.25 * price;
                return  salePrice;
        }
}
```

```
//ClassicCar Definition
public class ClassicCar extends Car {
        public ClassicCar(double price, int year)
        {
                super (price, year);
        }
        public double  calculateSalePrice ( )
        {
        return 10000;
        }
}
```

```
double total = 0;

for (Car car : cars ) {
    total += car.calculateSalePrice();
}

return total;
```

Q4

16 Points

```
class Item {
  private String s = new String(" 2000 ");

  public void append(String a) {
    s += a;
  }

  public void op2() {
    append(".1 ");
  }

  public void op3() {
    append(".2 ");
  }

  public void op4() {
    append(".3 ");
  }

  public String toString() {
    return s;
  }
}
```

```
class NewItem extends Item {
  // Change a method:
  public void op1() {
    append(" n1 ");
    }

  public void op4() {
      super.op4();
    append(" n4 ");
  }
}
class NewNewItem extends NewItem {
        // Change a method:
        public void op2() {
            super.op2();
          append(" nn3 ");
          }

        public void op4() {
          append(" nn4 ");
        }
      }
public class Library2 {
  // Test the new class:
  public static void main(String[] args) {
    NewNewItem x = new NewNewItem();
    x.op1();
    x.op2();
    x.op3();
    x.op4();
    System.out.println(x.toString());
  }
}
```

Write the output generated by executing the code above.

2000 n1 .1 nn3 .2 nn4

Q5

15 Points

Predict the output of following Java programs and select correct ones.

Q5.1

5 Points

```
class Main {
   public static void main(String args[]) {
      try {
         throw 10;
      }
      catch(int e) {
         System.out.println("Got the  Exception " + e);
      }
   }
}
```

- ○ Got the Exception 10
- ○ Got the Exception 0
- ◉ Compilation Error

Q5.2

5 Points

```
public class Test{

        String className;

        public static void main(String[] args){


                try{

                        Test t = new Test();
                        if(t.className.equals("Test"))
                                System.out.print("Test ");
                        else
                                System.out.print("Other ");

                }catch(Exception e){
                        System.out.print("Exception ");
                }catch(NullPointerException ne){
                        System.out.print("Null ");
                }

        }

}
```

- ○ Exception
- ◉ Compilation error
- ○ Null

Q5.3

5 Points

```
class Test extends Exception { }

class Main {
   public static void main(String args[]) {
      try {
         throw new Test();
      }
      catch(Test t) {
         System.out.println("Got the Test Exception");
      }
      finally {
         System.out.println("Inside finally block ");
      }
  }
}
```

- (●) Got the Test Exception \n Inside finally block
- ( ) Got the Test Exception
- ( ) Inside finally block

Q6

10 Points

Given the Java declarations

`interface I { void foo(); }`

and

`class B extends A implements I { ... }`

which of the following statement is true:

- ( ) Class B must provide a definition for foo(), no matter how class A is defined.
- (●) Class B need only provide a definition of foo() if A does not.
- ( ) Class B need only provide a definition of foo() if A does not implement I.
- ( ) Class B inherits foo() from I, thus B does not have to provide a definition of foo().

Q7

10 Points

Which of the following declarations could be correct in Java (where ... represents some code)?

- ○ `class C extends A, B { ... }`
- ○ `interface C implements A, B { ... }`
- ◉ `interface C extends A, B { ... }`
- ○ `class A {...} class B { ...} class C implements A, B { ... }`

Quiz4

● UNGRADED

STUDENT
Yasin Şimşek

TOTAL POINTS
- / 100 pts

QUESTION 1

(no title) 10 pts

1.1	(no title)	2 pts
1.2	(no title)	2 pts
1.3	(no title)	2 pts
1.4	(no title)	2 pts
1.5	(no title)	2 pts

QUESTION 2

(no title) 14 pts

2.1	(no title)	2 pts
2.2	(no title)	2 pts
2.3	(no title)	2 pts
2.4	(no title)	2 pts
2.5	(no title)	2 pts
2.6	(no title)	2 pts

2.7 (no title) 2 pts

QUESTION 3

(no title) 25 pts

QUESTION 4

(no title) 16 pts

QUESTION 5

(no title) 15 pts

5.1 (no title) 5 pts

5.2 (no title) 5 pts

5.3 (no title) 5 pts

QUESTION 6

(no title) 10 pts

QUESTION 7

(no title) 10 pts