TYPE 242

1. (1p) Which of the following statements are true? Give short explanations about your answers.
S1) An instance method can only be called after at least one object of the class is created.
S2) Inside a main method (public static void main()) we can access all the attributes and methods of the class that contains the inspected main method.
S3) We are allowed to use and change the current object inside the previous main method: **this = new MyClass();** where MyClass is the class without a constructor that contains method main.

2. (1p) a) What is returned by the two methods toString() from the next sequence of code? In-depth explain your answer.
b) Remove the code duplication among them (i.e. from the two toString() methods). The behaviour of the code is preserved and the underlined/bold code will appear only once!

```
class Expression { }

class CompoundExpression extends Expression {
    private Expression e1, e2;

    public CompoundExpression(Expression e1, Expression e2) {
        this.e1 = e1;
        this.e2 = e2;
    }
    public Expression getLExpression() { return e1; }
    public Expression getRExpression() { return e2; }
}

class AddExpression extends CompoundExpression {
    public AddExpression(Expression e1, Expression e2) { super(e1,e2); }
    public String toString() {
      return "(" + this.getLExpression() + "+" + this.getRExpression() + ")";
    }
}

class MultiplyExpression extends CompoundExpression {
    public MultiplyExpression(Expression e1, Expression e2) { super(e1,e2); }
    public String toString() {
        return "(" + this.getLExpression() + "*" + this.getRExpression() + ")";
    }
}
```

3. (1p) Considering the source code given below, answer the following questions and properly explain your answer. a) What is going to be printed after the execution of Line 2?
b) Since class Player extends the Object class, are we allowed to modify Line1 as if follows, Object c = new Player("MyPlayer"); without generating a compilation error inside main?

```
interface Person {
 String getName(); }

class Student implements Person {
 private String name;
 public Student(String n) {name = n;}
 public String getName() {return name;}
}
```

```
class Player extends Student {
 private String name;
 public Player(String n) {super(n);}
 public String getName() {return name;}}

class Main {
 public static void main(String argv[]) {
  Player c = new Player("MyPlayer"); //Line1
  System.out.println(c.getName());   //Line2
 }
}
```

4. (1.5p) Define a class named Person with a constructor accepting two parameters: a String for the individual's name and an int for their age. Provide two distinct code solutions to facilitate the creation of persons, with an emphasis on the system explicitly restricting the instantiation to no more than 32 persons. Please provide a detailed explanation for each solution.

5. (0.5p) Present the mechanism that allows us to use an instance of a derived class instead of an instance of its base class.

6. (1p) What is wrong inside the next piece of code? Motivate your answer.

```
import java.io.IOException;
class SuperClass {
    public void start() throws Exception{
        throw new IOException("Not able to open file");
    }
}

class SubClass extends SuperClass{
    public void start() throws IOException{
        throw new Exception("Not able to start");
    }
}
```

7. (0.25p) What is an immutable class? Is class Integer an immutable class?

8. (0.75p) Considering the source code given below, what is going to be printed after its execution (Lines L1, L2 and L3)? Motivate the answer.

```
class MyClass {
    private static int p;
    private int q;

    public void set(int p, int q)
    {
        this.p = p;
        this.q = q;
    }

    public int getP() {
        return p;
    }
    public int getQ() {
        return q;
    }}
```

```
class Mainn {
    public static void myMethod(MyClass c1, MyClass c2) {
        c1.set(25, 55);
        c1 = c2;
        c2 = new MyClass();
        c2.set(12, 18);
    }
    public static void main(String[] args) {
        MyClass c1 = new MyClass();
        c1.set(4, 8);
        MyClass c2 = new MyClass();
        c2.set(44, 88);
        System.out.println(c1.getP() + " " + c1.getQ()); //L1
        myMethod(c1,c2);
        System.out.println(c1.getP() + " " + c1.getQ()); //L2
        System.out.println(c2.getP() + " " + c2.getQ()); //L3
    }}
```

9. (1p) Define a Tagged Class and present at least three disadvantages of their usages within the source code of a system. Provide a small example (source code) in which you define a tagged class as well as an example where the previously defined Tagged Class has been removed.

10. (1p) Answer the following questions and motivate the given answers.
a) Do attributes have default values?
b) Can we override a private method?
c) Is it possible to specify private and protected modifiers for members within interfaces?
d) How many objects will be created in the following code? **String s = new String("Welcome");**