Information and Computer Science Department

King Fahd University of Petroleum and Minerals
College of Computer Sciences and Engineering
Information and Computer Science Department
Second Semester (082)
ICS 201 - Introduction to Computer Science

Major Exam 1
Wednesday, 8th April, 2009
Time: 120 minutes

Name:

ID#:

Please circle your section number below:

Section	01	02	03	04
Instructor	Alvi	Sukairi	Sukairi	Yahyaoui
Day and Time	SMW 9 - 9:50	SMW 9 -9:50	SMW 8 - 8:50	SMW 11 – 11:50

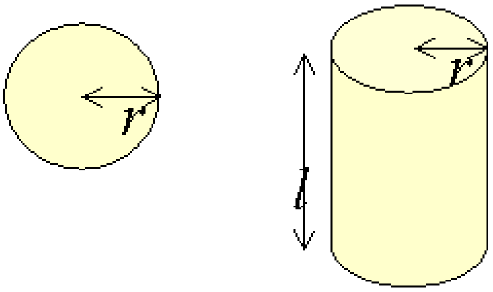
Question #	Maximum Mark	Obtained Mark
1	20	
2	15	
3	25	
4	10	
5	10	
6	20	
Total	**100**	

Question 1 (20 Marks)

Suppose we have the Circle class:

```
class Circle
{
        private double radius;

        public Circle(double r) { radius = r; }

        public double getRadius() { return radius; }

        public double computeCircumference() {
        return 2*Math.PI*radius;
        }

        public double computeArea() {
        return radius*radius*Math.PI;
        }
}
```

We want to define a *Cylinder* class. Cylinders are like Circles, but they have another variable: their length (See the figure).



1. [5 marks] Define a Cylinder constructor.

2. [5 marks] Define a method *computeVolume* that computes the volume of a cylinder which is the area of one of the associated circles (area of the cross-section) multiplied by its length.

3. [5 marks] We want to compute the surface area of a cylinder. That surface area is equal to the double of the area of one of the associated circles (area of the cross-section) plus the circle circumference multiplied by the length of the cylinder. Define that method in the class Cylinder using the **overriding** concept.

4. [5 marks] Consider the following piece of code:

```
Circle cl = new Circle();
Cylinder cy = (Cylinder)cl;
```

   a. Does this code compile? Why?

   b. Does this code run properly? Why?

Question 2 (15 Marks)

1. [2 marks] What is the difference between dynamic and static binding?

2. Consider the following classes:

```
class Employee {
 private String name;
 private double salary;
  public void work() {
    System.out.println("I am an employee.");
  }
  public static void manage() {
    System.out.println("I'm not responsible for managing ...");
  }

}

class Manager extends Employee {
  private double bonus;
  public void work() {
    System.out.println("I am a manager.");
  }

  public static void manage() {
    System.out.println("I can do managing ...");
  }
}


public class PolymorphismTest {
  public static void main(String[] args) {
    Employee employee;
    employee = new Manager();
    System.out.println(employee.getClass().getName());
    employee.work();
    Manager manager = (Manager) employee;   // S1
    manager.manage();
  }
}
```

a. [2 marks] What kind of casting is S1? Does it generate a run-time error and why?

b. [4 marks] Identify dynamic and static bindings of method calls in the main method of *PolymorphismTest*.

Dynamic Binding	Static Binding

Note that you have the following method declarations:
public String getName()
public final Class getClass()

c. [3 marks] What is the output of *PolymorphismTest*?

3. [4 marks] Provide an implementation of the method **equals(Object o)** in the class *Manager*. You can assume that the class Employee has accessors and mutators for its fields.

Question 3 (25 Marks)

1. [10 marks] Write an abstract class **CommercialVehicle** with the following features:

    - *Instance Variables*: **name** (String), **numberOfTrips** (int).
    - *Abstract Methods*: a **totalCharge()** method which returns the total charge for the vehicle.
    - *Constructor*.
    - *Methods*: Accessor methods, **toString()** method which displays the name of the vehicle and its total charge (returned by the total charge method).

2. [10 marks] Write a subclass of **CommercialVehicle** called **Taxi.** The class **Taxi** consists of the following additional instance variables: **chargePerTrip** (double). Implement the **totalCharge**() method by multiplying the **chargePerTrip** by the **numberOfTrips**.

3. [5 marks] The following test class is written here for the above program. Find the output for the following:

```
public class VehicleTest {
        public static void main(String[] args) {
                CommercialVehicle[] t = new Taxi[3];

//Note that the order of parameters in your constructor might be different
                t[0] = new Taxi("Camry", 20.00, 5);
                t[1] = new Taxi("Hyundai", 10.00, 9);
                t[2] = new Taxi("Suzuki", 12.50, 8);

                for(int i = 0; i < t.length; i++)
                        System.out.println(t[i]);

        }
}
```

Question 4 (10 marks)

Consider the following program:

```
import java.util.*;

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

                Scanner s = new Scanner(System.in);
                System.out.println("Enter a double-precision value: ");

                double val = s.nextDouble();
                double logarithm = Math.log(val);
                int z = (int) logarithm;

                System.out.println("Log of "+val+ " is >= " +z);
        }
}
```

Two possible sources of exceptions in this program are:

- For example, if the user enters a string instead of a double-precision value, an exception is thrown.

- Similarly, if the user enters a non-postive value (value <= 0), the line **double logarithm = Math.log(val);** will give erroneous result.

1. [4 marks] Write your own exception class **NonPositiveValueException** to indicate that a non-positive number has been entered by the program user.

2. [6 marks] Re-write the above program **ExceptionHandling** using a try-catch block and handle both these exceptions. Print appropriate message specifying the type of exception and then terminate the program. If no exception occurs, print the result.

Question 5 (10 marks)

Show, **using a diagram**, all the steps needed to develop and run java program.

Question 6 (20 marks)

1. [15 marks] Design and implement a recursive method **printInt** to print its integer argument with commas in the correct places. For instance, **printInt(1928764)** should print **1,928,764**. Also write a test class to test the method.

2. [5 marks] What is the output of the following program:

```
public class Recursion
{
   public static void main(String[] args)
   {
      for (int n = 1; n < 3; n++)
         System.out.println("3^" +n+ " is " + rec(3, n));
   }

   public static int rec(int x, int n)
   {
      if (n < 0)
      {
         System.out.println("Illegal argument");
         System.exit(0);
      }

      if (n > 0)
         return ( rec(x, n - 1)*x );
      else
          return 1;
   }
}
```