CT3: CSE 203 (Sec C) - Object-Oriented Programming I: Java

Time: 25 min **Mark: 20**

Name: **ID:**

Question 1: **[10]**

Define an abstract class named "**Performer**". Add the following inside the class.

a) Attributes: ***name*** and ***field***
b) Add a parameterized **constructor** which will take parameters for all instance variables and initializes the respective attributes with the parameters passed to the constructor.
c) Add the following methods.
    i. Concrete method: Override **toString**() and return the values of attributes as a concatenated String,
    ii. Abstract method:
        - The first abstract method named ***perform*** that doesn't take any parameter and doesn't return anything either.
        - A method named ***getRank*** that will return a float and does not take any parameter.

Question 2: **[10]**

Define a concrete class named **Dancer** and make it subclass of **Performer** class. Write the minimum code required to make the **Dancer** class error-free. Additionally, add the following to this **Dancer** class.

a) Attributes: ***rank***
b) Add a parameterized **constructor** which will take parameters for all 3 instance variables (2 from parent and one from this class). Implement the constructor in proper way.

CT3: CSE 203 (Sec C) - Object-Oriented Programming I: Java

Time: 25 min **Mark: 20**

Name: **ID:**

Question 1: **[8]**

Declare an interface named **Engine**. Add 3 abstract methods named ***startEngine***, ***run***, ***stopEngine***. None of these methods will return anything and does not take any parameters.

Question 2: **[12]**

Carefully observe the code below. Create a class name **Aeroplane** which should be a subclass of the following Vehicle class and implement the Engine interface. Write the minimum code required to make the **Aeroplane** class error-free. Additionally, add the following to this **Aeroplane** class.

a) Attributes: ***enginePower***
b) Add a parameterized **constructor** which will take parameters; both ***model*** and ***enginePower*** attributes. Implement the constructor in proper way.

```
public class Vehicle {
    String model;

    public Vehicle(String model) {
        this.model = model;
    }

    public void accelerate() {
        System.out.println("Running");
    }
}
```