

CDAC
Mumbai PG-DAC August 24

Assignment No- 5

Name: Qureshi Mohammed Muqarrab
PRN No: 240840520036
CDAC Juhu

1) Create a base class BankAccount with methods like deposit() and withdraw(). Derive a class SavingsAccount that overrides the withdraw() method to impose a limit on the withdrawal amount. Write a program that demonstrates the use of overridden methods and proper access modifiers & return the details.

```
package org.example;

class BankAccount {
    protected double balance;

    public BankAccount(double balance) {
        this.balance = balance;
    }

    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited: ₹" + amount + ", New Balance: ₹" + balance);
    }

    public void withdraw(double amount) {
        if(balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn: ₹" + amount + ", New Balance: ₹" + balance);
        } else {
            System.out.println("Insufficient funds. Withdrawal failed");
        }
    }

    public double getalance() {
        return balance;
    }
}

class SavingsAccount extends BankAccount {
    private double withdrawallimit;
```

```

    public SavingsAccount(double balance, double withdrawalLimit) {
        // TODO Auto-generated constructor stub
        super(balance);
        this.withdrawalLimit = withdrawalLimit;
    }

    @Override
    public void withdraw(double amount) {
        if(amount > withdrawalLimit) {
            System.out.println("Withdrawal limit exceeded. Limit: ₹" + withdrawalLimit);
        }else {
            super.withdraw(amount);
        }
    }
}

public class BankAccountDemo {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        SavingsAccount savingAccount = new SavingsAccount(1000, 500);
        savingAccount.deposit(200);
        savingAccount.withdraw(600); //Exceeds Limit
        savingAccount.withdraw(400); //Allowed
    }
}

```

Output:

The screenshot shows the Eclipse IDE interface. The main editor displays the source code for the `SavingsAccount` and `BankAccountDemo` classes. The Package Explorer on the left shows the project structure. The Console window at the bottom shows the output of the program execution.

```

-terminated- BankAccountDemo [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (17 Sept 2024, 8:07:51 pm - 8:07:56 pm) [pid: 2916]
Deposited: ₹200.0, New Balance: ₹1200.0
Withdrawal limit exceeded, Limit: ₹500.0
Withdrawn: ₹400.0, New Balance: ₹800.0

```

2) Create a base class Vehicle with attributes like make and year. Provide a constructor in Vehicle to initialize these attributes. Derive a class Car that has an additional attribute model and write a constructor that initializes make, year, and model. Write a program to create a Car object and display its details.

```
package org.example1;

class Vehicle{
    protected String make;
    protected int year;

    public Vehicle(String make, int year) {
        this.make = make;
        this.year = year;
    }

    public void displayDetails() {
        System.out.println("Make: " + make);
        System.out.println("Year: " + year);
    }
}

class Car extends Vehicle {
    private String model;

    public Car(String make, int year, String model) {
        super(make, year);
        this.model = model;
    }

    @Override
    public void displayDetails() {
        super.displayDetails();
        System.out.println("Model: " + model );
    }
}

public class VehicleDemo {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Car car = new Car("Toyota", 2024, "Supra");

        car.displayDetails();
    }
}
```

Output:

```
13 public class VehicleDemo {
14     System.out.println("Make: " + make);
15     System.out.println("Year: " + year);
16 }
17
18 class Car extends Vehicle {
19     private String model;
20
21     public Car(String make, int year, String model) {
22         super(make, year);
23         this.model = model;
24     }
25
26     @Override
27     public void displayDetails() {
28         super.displayDetails();
29         System.out.println("Model: " + model);
30     }
31 }
32
33 public class VehicleDemo {
34
35     public static void main(String[] args) {
36         // TODO Auto-generated method stub
37         Car car = new Car("Toyota", 2024, "Supra");
38
39         car.displayDetails();
40     }
41 }
42 }
43 }
```

terminated: VehicleDemo [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (17 Sept 2024, 9:07:35 pm - 9:07:38 pm) [pid: 4204]

Make: Toyota
Year: 2024
Model: Supra

3) Create a base class `Animal` with attributes like `name`, and methods like `eat()` and `sleep()`. Create a subclass `Dog` that inherits from `Animal` and has an additional method `bark()`. Write a program to demonstrate the use of inheritance by creating objects of `Animal` and `Dog` and calling their methods.

```
package org.example;

class Animal {
    protected String name;

    public Animal(String name) {
        this.name = name;
    }

    public void eat() {
        System.out.println(name + " is eating.");
    }

    public void sleep() {
        System.out.println(name + " is sleeping.");
    }
}

class Dog extends Animal {
    public Dog(String name) {
        super(name);
    }

    public void bark() {
        System.out.println(name + " is barking");
    }
}

public class AnimalDemo {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Animal animal = new Animal("Lion");
        animal.eat();
        animal.sleep();

        Dog dog = new Dog("Shiro");
        dog.eat();
        dog.sleep();
        dog.bark();
    }
}
```

```
}
```

Output:

The screenshot shows the Eclipse IDE interface. The main editor displays the source code for `AnimalDemo.java`. The code defines an `Animal` class with a `name` attribute and methods `eat()` and `sleep()`. A `Dog` class extends `Animal` and implements `bark()`. The `AnimalDemo` class contains a `main` method that creates instances of `Animal` and `Dog` and calls their methods. The console at the bottom shows the output of the program, which is: `Lion is eating.`, `Lion is sleeping.`, `Shiro is eating.`, `Shiro is sleeping.`, and `Shiro is barking.`

```
1 package org.example;
2
3 class Animal {
4     protected String name;
5
6     public Animal(String name) {
7         this.name = name;
8     }
9
10    public void eat() {
11        System.out.println(name + " is eating.");
12    }
13
14    public void sleep() {
15        System.out.println(name + " is sleeping.");
16    }
17 }
18
19 class Dog extends Animal {
20     public Dog(String name) {
21         super(name);
22     }
23
24     public void bark() {
25         System.out.println(name + " is barking");
26     }
27 }
28
29 public class AnimalDemo {
30     public static void main(String[] args) {
31         // ...
32     }
33 }
```

Console Output:

```
<terminated> AnimalDemo [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (17 Sept 2024, 9:47:47 pm - 9:47:50 pm) [pid: 17120]
Lion is eating.
Lion is sleeping.
Shiro is eating.
Shiro is sleeping.
Shiro is barking.
```

4) Build a class Student which contains details about the Student and compile and run its instance.

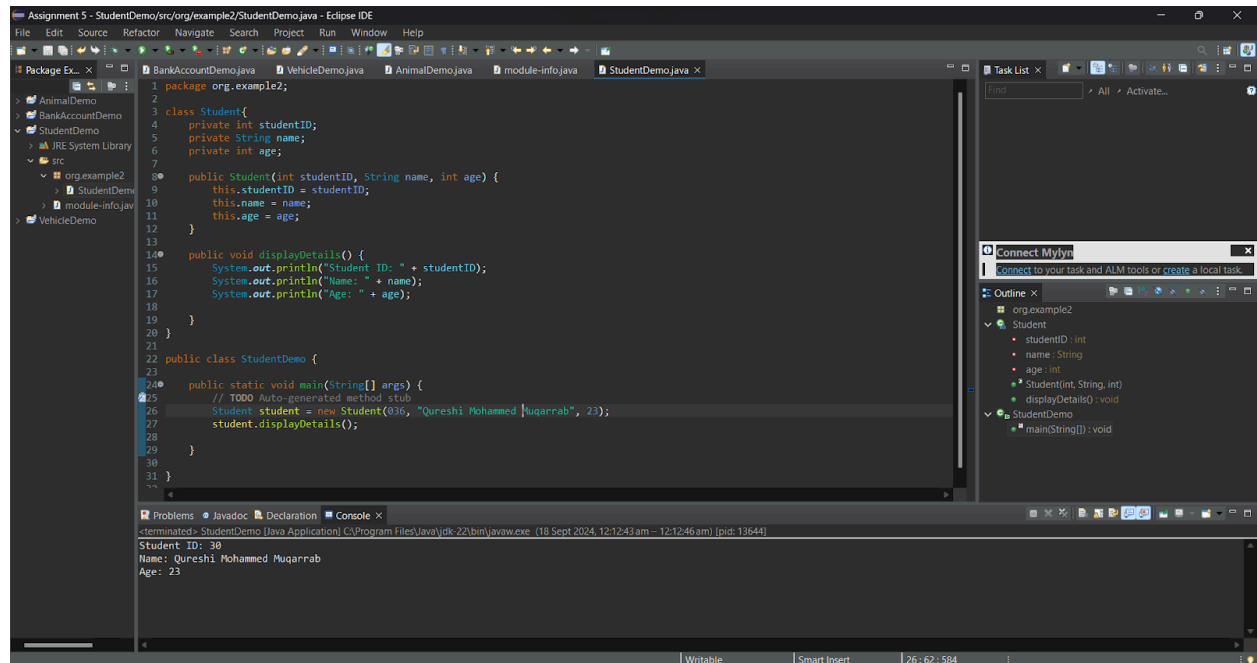
```
package org.example2;
class Student{
    private int studentID;
    private String name;
    private int age;

    public Student(int studentID, String name, int age) {
        this.studentID = studentID;
        this.name = name;
        this.age = age;
    }

    public void displayDetails() {
        System.out.println("Student ID: " + studentID);
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
    }
}

public class StudentDemo {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Student student = new Student(036, "Qureshi Mohammed Muqarrab", 23);
        student.displayDetails();
    }
}
```

Output:



5) Write a Java program to create a base class Vehicle with methods startEngine() and stopEngine(). Create two subclasses Car and Motorcycle. Override the startEngine() and stopEngine() methods in each subclass to start and stop the engines differently.

```

package org.example3;

class Vehicle {
    public void startEngine() {
        System.out.println("Vehicle engine is starting."); //start vehicle
    }
    public void stopEngine() {
        System.out.println("Vehicle engine is stopping."); //stop vehicle
    }
}

class Car extends Vehicle {
    @Override
    public void startEngine() { //class car extends vehicle (Car is a vehicle)
        System.out.println("Car engine is starting with a key.");
    }
    @Override
    public void stopEngine() {
        System.out.println("Car engine is stopping."); //override methods for car
    }
}

class Motorcycle extends Vehicle {
    @Override
    public void startEngine() {
        System.out.println("Motorcycle engine is starting with a button.");
    }
    @Override

```



```

    public void stopEngine() { //override-method (Motorcycle is a vehicle)
        System.out.println("Motorcycle engine is stopping.");
    }
}

public class VehicleDemo2 { //main class and method
    public static void main(String[] args) {
        Vehicle car = new Car();
        car.startEngine();
        car.stopEngine();
        Vehicle motorcycle = new Motorcycle();
        motorcycle.startEngine();
        motorcycle.stopEngine();
    }
}

```

Output:

The screenshot shows the Eclipse IDE with the following components:

- Editor:** Displays the source code of `VehicleDemo2.java`. The code includes a `stopEngine()` method override for `Motorcycle` and a `main` method that creates and operates `Car` and `Motorcycle` objects.
- Outline:** Shows the project structure with packages `org.example3` and classes `Vehicle`, `Car`, `Motorcycle`, and `VehicleDemo2`.
- Console:** Displays the runtime output:


```

      <terminated> VehicleDemo2 [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe. (18 Sept 2024, 1:22:29 am - 1:22:31 am) [pid: 15288]
      Car engine is starting with a key.
      Car engine is stopping.
      Motorcycle engine is starting with a button.
      Motorcycle engine is stopping.
      
```