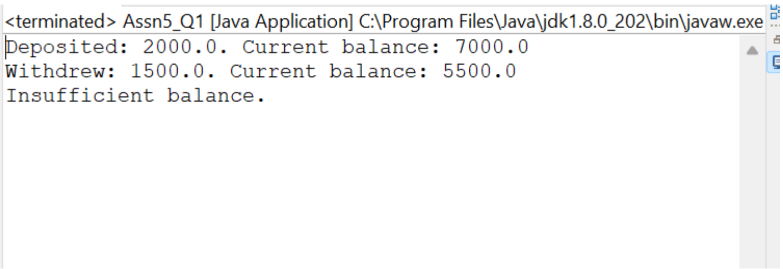
CDAC Mumbai PG-DAC August 24

Assignment No- 5

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.classWork;
class BankAccount {
private double balance;
// Default constructor initializing balance to 0.0
public BankAccount() {
this(0.0);
}
// Constructor to initialize balance with a given amount
public BankAccount(double balance) {
this.balance = balance;
}
// Method to deposit an amount into the account
public void deposit(double amount) {
balance += amount;
System.out.println("Deposited: " + amount + ". Current balance: " + balance);
}
// Method to withdraw an amount from the account
public void withdraw(double amount) {
if (amount > balance) {
System.out.println("Insufficient balance.");
} else {
balance -= amount;
System.out.println("Withdrew: " + amount + ". Current balance: " + balance);
}
}
// Getter method for balance
public double getBalance() {
return balance;
}
}
class SavingsAccount extends BankAccount {
private double withDrawLimit = 100000; // Withdrawal limit
// Constructor calling the parent class constructor
public SavingsAccount(double balance) {
super(balance);
}
@Override
public void withdraw(double amount) {
// Check if the withdrawal amount exceeds the limit
if (amount > withDrawLimit) {
System.out.println("Withdrawal amount exceeds the limit of " + withDrawLimit);
}
// Check if the balance is sufficient for withdrawal
else if (amount > getBalance()) {
System.out.println("Insufficient balance.");
}
// Perform the withdrawal if all checks pass
else {
super.withdraw(amount);
}
}
}
public class Assn5_Q1 {
public static void main(String[] args) {
SavingsAccount sa = new SavingsAccount(5000); // Initial balance of 5000
sa.deposit(2000); // Deposit 2000
sa.withdraw(1500); // Try to withdraw 1500
sa.withdraw(100000); // Try to withdraw above the limit
}
}
```



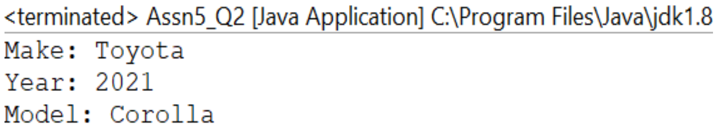
<terminated> Assn5_Q1 [Java Application] C:\Program Files\Java\jdk1.8.0_202\bin\javaw.exe

```
Deposited: 2000.0. Current balance: 7000.0
Withdrew: 1500.0. Current balance: 5500.0
Insufficient balance.
```

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.classWork;

class Vehicle {

private String make;

private int year;

// Constructor for Vehicle to initialize make and year

public Vehicle(String make, int year) {

this.make = make;

this.year = year;

}
```

```java
// Getter for make
public String getMake() {
return make;
}
// Getter for year
public int getYear() {
return year;
}
// Method to display Vehicle details
public void displayDetails() {
System.out.println("Make: " + make);
System.out.println("Year: " + year);
}
}
class Car extends Vehicle {
private String model;
// Constructor for Car that initializes make, year, and model
public Car(String make, int year, String model) {
// Call the constructor of the base class (Vehicle)
super(make, year);
this.model = model;
}
// Getter for model
public String getModel() {
return model;
}
// Override the displayDetails method to include model information
@Override
public void displayDetails() {
super.displayDetails(); // Display make and year from Vehicle class
System.out.println("Model: " + model);
}

}

public class Assn5_Q2 {

public static void main(String[] args) {

// Create a Car object and initialize make, year, and model

Car car = new Car("Toyota", 2021, "Corolla");

// Display the details of the car

car.displayDetails();

}

}
```

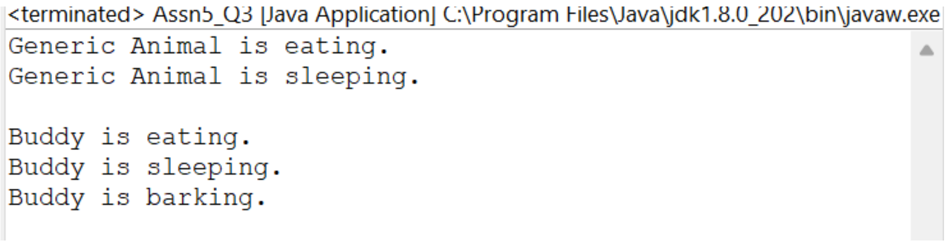


```
<terminated> Assn5_Q2 [Java Application] C:\Program Files\Java\jdk1.8
Make: Toyota
Year: 2021
Model: Corolla
```

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.classWork;

// Base class Animal

class Animal {

private String name;

// Constructor for Animal to initialize the name

public Animal(String name) {

this.name = name;

}

// Method to simulate eating

public void eat() {

System.out.println(name + " is eating.");

}
```

```java
// Method to simulate sleeping
public void sleep() {
System.out.println(name + " is sleeping.");
}
// Getter for the name
public String getName() {
return name;
}
}
// Subclass Dog that inherits from Animal
class Dog extends Animal {
// Constructor for Dog to initialize the name by calling the base class constructor
public Dog(String name) {
super(name); // Call to the superclass constructor to set the name
}
// Method specific to Dog
public void bark() {
System.out.println(getName() + " is barking.");
}
}
// Main class to demonstrate inheritance
public class Assn5_Q3 {
public static void main(String[] args) {
// Creating an Animal object
Animal animal = new Animal("Generic Animal");
animal.eat();
animal.sleep();
System.out.println();
// Creating a Dog object (subclass of Animal)
Dog dog = new Dog("Buddy");
dog.eat(); // Inherited from Animal
dog.sleep(); // Inherited from Animal

dog.bark(); // Method specific to Dog

}

}
```



<terminated> Assn5_Q3 [Java Application] C:\Program Files\Java\jdk1.8.0_202\bin\javaw.exe

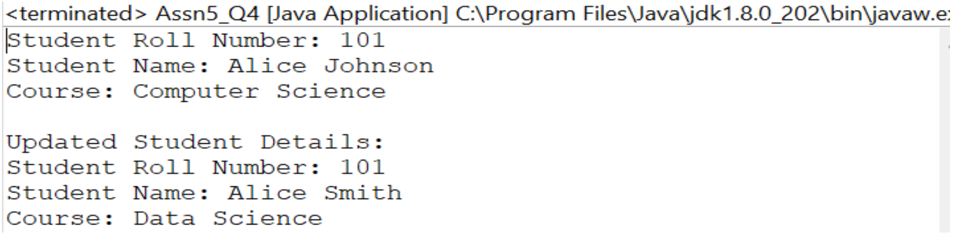
```
Generic Animal is eating.
Generic Animal is sleeping.

Buddy is eating.
Buddy is sleeping.
Buddy is barking.
```

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

```
package org.classWork;

// Student class containing details about a student

class Student {

// Attributes

private int rollNumber;

private String name;

private String course;

// Constructor to initialize the student's details

public Student(int rollNumber, String name, String course) {

this.rollNumber = rollNumber;

this.name = name;

this.course = course;

}

// Getter for roll number

public int getRollNumber() {

return rollNumber;

}

// Setter for roll number

public void setRollNumber(int rollNumber) {

this.rollNumber = rollNumber;
```

```
}
// Getter for name
public String getName() {
return name;
}
// Setter for name
public void setName(String name) {
this.name = name;
}
// Getter for course
public String getCourse() {
return course;
}
// Setter for course
public void setCourse(String course) {
this.course = course;
}
// Method to display student details
public void displayDetails() {
System.out.println("Student Roll Number: " + rollNumber);
System.out.println("Student Name: " + name);
System.out.println("Course: " + course);
}
}
// Main class to test the Student class
public class Assn5_Q4 {
public static void main(String[] args) {
// Creating an instance of the Student class
Student student1 = new Student(101, "Alice Johnson", "Computer Science");
// Display student details
student1.displayDetails();
```

```
// Modifying student details
student1.setName("Alice Smith");
student1.setCourse("Data Science");
System.out.println("\nUpdated Student Details:");
student1.displayDetails();
}
}
```

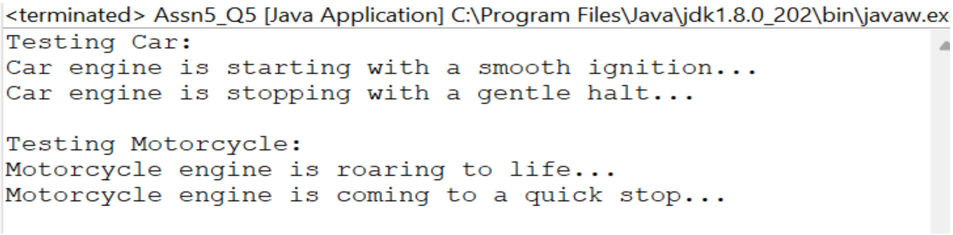


```
<terminated> Assn5_Q4 [Java Application] C:\Program Files\Java\jdk1.8.0_202\bin\javaw.e
Student Roll Number: 101
Student Name: Alice Johnson
Course: Computer Science

Updated Student Details:
Student Roll Number: 101
Student Name: Alice Smith
Course: Data Science
```

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.classWork;
// Base class Vehicle
class BaseVehicle {
// Method to start the engine
public void startEngine() {
System.out.println("Vehicle engine is starting...");
}
// Method to stop the engine
public void stopEngine() {
System.out.println("Vehicle engine is stopping...");
}
}
// Subclass Car inheriting from BaseVehicle
class SubCar extends BaseVehicle {
// Overriding the startEngine method for Car
```

```java
@Override
public void startEngine() {
System.out.println("Car engine is starting with a smooth ignition...");
}
// Overriding the stopEngine method for Car
@Override
public void stopEngine() {
System.out.println("Car engine is stopping with a gentle halt...");
}
}
// Subclass Motorcycle inheriting from BaseVehicle
class SubMotorcycle extends BaseVehicle {
// Overriding the startEngine method for Motorcycle
@Override
public void startEngine() {
System.out.println("Motorcycle engine is roaring to life...");
}
// Overriding the stopEngine method for Motorcycle
@Override
public void stopEngine() {
System.out.println("Motorcycle engine is coming to a quick stop...");
}
}
// Main class to test the Vehicle, Car, and Motorcycle classes
public class Assn5_Q5 {
public static void main(String[] args) {
// Create an instance of SubCar
BaseVehicle car = new SubCar();
System.out.println("Testing Car:");
car.startEngine();
car.stopEngine();
// Create an instance of SubMotorcycle

BaseVehicle motorcycle = new SubMotorcycle();

System.out.println("\nTesting Motorcycle:");

motorcycle.startEngine();

motorcycle.stopEngine();

}

}
```



<terminated> Assn5_Q5 [Java Application] C:\Program Files\Java\jdk1.8.0_202\bin\javaw.ex

```
Testing Car:
Car engine is starting with a smooth ignition...
Car engine is stopping with a gentle halt...

Testing Motorcycle:
Motorcycle engine is roaring to life...
Motorcycle engine is coming to a quick stop...
```