**CDAC Mumbai PG-DAC August 24**

**Assignment No- 5 with code solution**

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

instance.

1. 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.

5. class Vehicle {

String make;

int year;

public Vehicle(String make, int year) {

this.make = make;

this.year = year;

}

}

class Car extends Vehicle {

String model;

public Car(String make, int year, String model) {

super(make, year);

this.model = model;

}

public void displayDetails() {

System.out.println("Make: " + make);

System.out.println("Year: " + year);

System.out.println("Model: " + model);

}

}

public class VehicleDemo {

public static void main(String[] args) {

Car car = new Car("Toyota", 2023, "Camry");

car.displayDetails();

}

}

4. class Vehicle {

public void startEngine() {

System.out.println("Starting engine...");

}

public void stopEngine() {

System.out.println("Stopping engine...");

}

}

class Car extends Vehicle {

@Override

public void startEngine() {

System.out.println("Starting car engine...");

}

@Override

public void stopEngine() {

System.out.println("Stopping car engine...");

}

}

class Motorcycle extends Vehicle {

@Override

public void startEngine() {

System.out.println("Starting motorcycle engine...");

}

@Override

public void stopEngine() {

System.out.println("Stopping motorcycle engine...");

}

}

public class VehicleDemo {

public static void main(String[] args) {

Vehicle car = new Car();

car.startEngine();

car.stopEngine();

Vehicle motorcycle = new Motorcycle();

motorcycle.startEngine();

motorcycle.stopEngine();

}

}

3. class Student {

String name;

int age;

String grade;

public Student(String name, int age, String grade) {

this.name = name;

this.age = age;

this.grade = grade;

}

public void displayDetails() {

System.out.println("Name: " + name);

System.out.println("Age: " + age);

System.out.println("Grade: " + grade);

}

}

public class StudentDemo {

public static void main(String[] args) {

Student student = new Student("Alice", 18, "12th Grade");

student.displayDetails();

}

}

2. class Animal {

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 InheritanceDemo {

public static void main(String[] args) {

Animal animal = new Animal("Generic Animal");

animal.eat();

animal.sleep();

Dog dog = new Dog("Buddy");

dog.eat();

dog.sleep();

dog.bark();

}

}

1. class BankAccount {

protected double balance;

public BankAccount(double initialBalance) {

balance = initialBalance;

}

public void deposit(double amount) {

balance += amount;

}

public void withdraw(double amount) {

if (balance >= amount) {

balance -= amount;

} else {

System.out.println("Insufficient funds.");

}

}

public double getBalance() {

return balance;

}

}

class SavingsAccount extends BankAccount {

private final double withdrawalLimit = 5000;

public SavingsAccount(double initialBalance) {

super(initialBalance);

}

@Override

public void withdraw(double amount) {

if (amount > withdrawalLimit) {

System.out.println("Withdrawal limit exceeded.");

} else {

super.withdraw(amount);

}

}

}

public class BankAccountDemo {

public static void main(String[] args) {

SavingsAccount savingsAccount = new SavingsAccount(10000);

savingsAccount.deposit(2000);

System.out.println("Balance after deposit: " + savingsAccount.getBalance());

savingsAccount.withdraw(3000);

System.out.println("Balance after withdrawal: " + savingsAccount.getBalance());

savingsAccount.withdraw(8000); // Should print "Withdrawal limit exceeded."

System.out.println("Balance after failed withdrawal: " + savingsAccount.getBalance());

}

}