**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.

Solution:

**package** assignment.in;

**import** java.util.Scanner;

**class** BankAccount {

**private** String accountHolder;

**private** **double** balance;

// Constructor

**public** BankAccount(String accountHolder, **double** balance) {

**this**.accountHolder = accountHolder;

**this**.balance = balance;

}

// Deposit method

**public** **void** deposit(**double** amount) {

**if** (amount > 0) {

balance += amount;

System.***out***.println("Deposited: " + amount);

}

**else**

{

System.***out***.println("Deposit amount must be positive.");

}

}

// Withdraw method

**public** **boolean** withdraw(**double** amount) {

**if** (amount > 0 && amount <= balance) {

balance -= amount;

System.***out***.println("Successfully withdrew: " + amount);

**return** **true**;

}

**else**

{

System.***out***.println("Insufficient balance or invalid amount.");

**return** **false**;

}

}

// Method to display account details

**public** **void** showDetails() {

System.***out***.println("Account Holder: " + accountHolder);

System.***out***.println("Balance: " + balance);

}

// Getter for balance

**public** **double** getBalance() {

**return** balance;

}

}

**class** SavingsAccount **extends** BankAccount {

**private** **double** withdrawalLimit;

// Constructor

**public** SavingsAccount(String accountHolder, **double** balance, **double** withdrawalLimit) {

**super**(accountHolder, balance);

**this**.withdrawalLimit = withdrawalLimit;

}

// Overridden withdraw method with limit

@Override

**public** **boolean** withdraw(**double** amount) {

**if** (amount > withdrawalLimit) {

System.***out***.println("Withdrawal amount exceeds the limit of: " + withdrawalLimit);

**return** **false**;

}

**else**

{

**return** **super**.withdraw(amount); // Call parent class method

}

}

// Method to show details with withdrawal limit

@Override

**public** **void** showDetails() {

**super**.showDetails();

System.***out***.println("Withdrawal Limit: " + withdrawalLimit);

}

}

**public** **class** BankDetails {

**public** **static** **void** main(String[] args) { // Corrected main method name

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("Enter account holder name: ");

String accountHolder = sc.nextLine();

System.***out***.print("Enter initial balance: ");

**double** balance = sc.nextDouble();

System.***out***.print("Enter withdrawal limit: ");

**double** withdrawalLimit = sc.nextDouble();

// Create SavingsAccount object

SavingsAccount savings = **new** SavingsAccount(accountHolder, balance, withdrawalLimit);

**int** choice;

**do** {

// Display menu options

System.***out***.println("\n--- Menu ---");

System.***out***.println("1. Deposit Amount");

System.***out***.println("2. Withdraw Amount");

System.***out***.println("3. Show Account Details");

System.***out***.println("4. Exit");

System.***out***.print("Enter your choice: ");

choice = sc.nextInt();

**switch** (choice) {

**case** 1:

// Deposit

System.***out***.print("Enter amount to deposit: ");

**double** depositAmount = sc.nextDouble();

savings.deposit(depositAmount);

**break**;

**case** 2:

// Withdraw

System.***out***.print("Enter amount to withdraw: ");

**double** withdrawAmount = sc.nextDouble();

savings.withdraw(withdrawAmount);

**break**;

**case** 3:

// Show Account Details

savings.showDetails();

**break**;

**case** 4:

// Exit

System.***out***.println("Exiting...");

**break**;

**default**:

System.***out***.println("Invalid choice.");

}

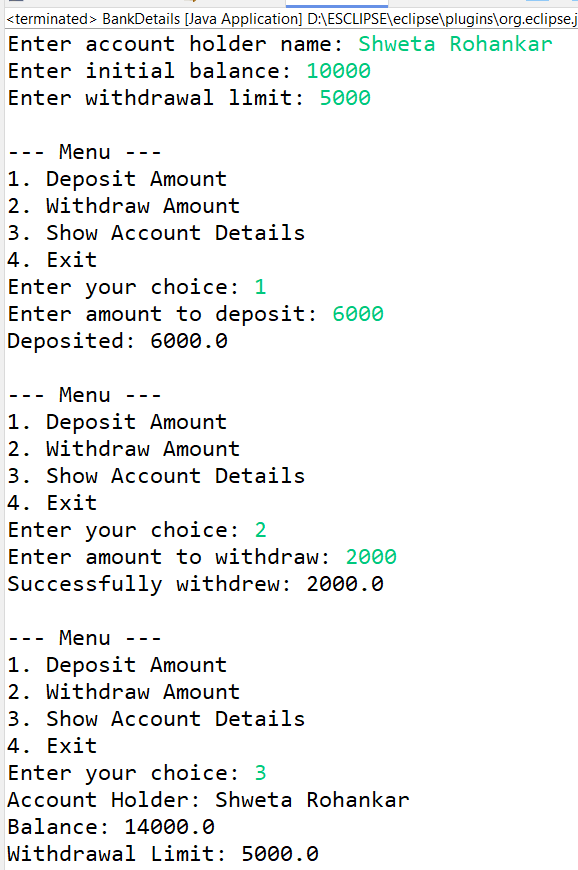
} **while** (choice != 4);

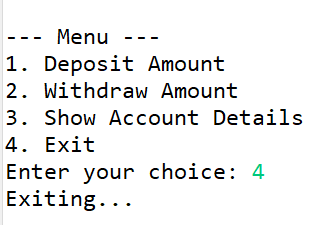
sc.close();

}

}

Output:





1. 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.

Solution:

**package** assignment.in;

**class** vehicle{

**private** String make;

**private** **int** year;

**public** vehicle(String make, **int** year) {

**this**.make = make;

**this**.year = year;

}

**public** String getMake() {

**return** make;

}

**public** **int** getYear() {

**return** 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;

}

**public** String getModel() {

**return** model;

}

@Override

**public** **void** displayDetails() {

**super**.displayDetails(); // Display make and year from Vehicle

System.***out***.println("Model: " + model); // Display model from Car

}

}

**public** **class** Victor {

**public** **static** **void** main(String[] args) {

// Create a Car object

Car car = **new** Car("Toyato", 2009, "Fortuner");

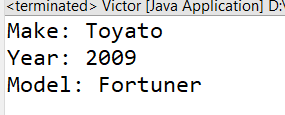
// Display the car details

car.displayDetails();

}

}

Output:



1. 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.

Solution:

package assignment.in;

class Animal{

private 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.");

}

public String getName() {

return name;

}

}

//subclass

class Dog extends Animal{

public Dog(String name) {

super(name);

}

public void bark() {

System.out.println(getName() + " is barking.");

}

}

public class InheritanceAnimal {

public static void main(String[] args) {

Animal animal = new Animal("Bear");

animal.eat();

animal.sleep();

System.out.println();

Dog dog = new Dog("Teddy");

dog.eat();

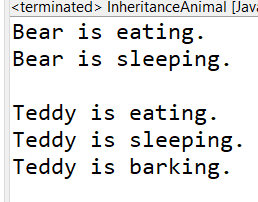
dog.sleep();

dog.bark();

}

}

Output:



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

instance.

Solution:

**package** assignment.in;

**class** Student{

**private** String name;

**private** **int** age;

**private** **int** PrnNumber;

**private** String course;

**private** String batchYear;

**public** Student(String name, **int** age, **int** prnNumber, String course, String batchYear)

{

**super**();

**this**.name = name;

**this**.age = age;

PrnNumber = prnNumber;

**this**.course = course;

**this**.batchYear = batchYear;

}

**public** String getName() {

**return** name;

}

**public** **int** getAge() {

**return** age;

}

**public** **int** getPrnNumber() {

**return** PrnNumber;

}

**public** String getCourse() {

**return** course;

}

**public** String getBatchYear() {

**return** batchYear;

}

**public** **void** displayStudentsDetails() {

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

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

System.***out***.println("Student PRN : " +PrnNumber);

System.***out***.println("Student course Name : " +course);

System.***out***.println("Student Batch year : " +batchYear);

}

}

**public** **class** StudentDetails {

**public** **static** **void** main(String[] args) {

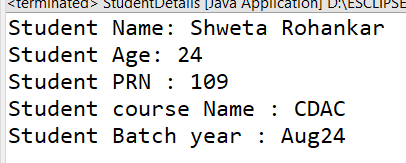
Student student = **new** Student("Shweta Rohankar", 24, 109, "CDAC" ,"Aug24");

student.displayStudentsDetails();

}

}

Output:



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.

Solution:

**package** assignment.in;

**class** Vehicle1 {

**public** **void** startEngine() {

System.***out***.println("The vehicle's engine is starting.");

}

**public** **void** stopEngine() {

System.***out***.println("The vehicle's engine is stopping.");

}

}

// Subclass Car

**class** Car1 **extends** Vehicle1 {

@Override

**public** **void** startEngine() {

System.***out***.println("The car's engine is starting.");

}

@Override

**public** **void** stopEngine() {

System.***out***.println("The car's engine turns off smoothly.");

}

}

// Subclass Motorcycle

**class** Motorcycle **extends** Vehicle1 {

@Override

**public** **void** startEngine() {

System.***out***.println("The motorcycle's engine starts with a button press.");

}

@Override

**public** **void** stopEngine() {

System.***out***.println("The motorcycle's engine shuts down quickly.");

}

}

**public** **class** VehicleEngine {

**public** **static** **void** main(String[] args) {

Vehicle1 car = **new** Car1();

System.***out***.println("Car:");

car.startEngine();

car.stopEngine();

System.***out***.println();

Vehicle1 motorcycle = **new** Motorcycle();

System.***out***.println("Motorcycle:");

motorcycle.startEngine();

motorcycle.stopEngine();

}

}

Output:

