Learner Assignment _DAY-5

Learner Details

- **Name: SOWJANYA BS**
- **Enrollment Number:**
- **Batch / Class:**
- **Assignment: (Bridge Course Day 5)**
- **Date of Submission:30/06/2025**

SECTION 1 :
Problem Statement 1 :Create a class named Dog that has:

- Two class attributes:
  species and numLegs
- Three instance attributes:
  name, breed, and age
- A method called bark() that prints "Woof!"

Pseudo code :

Start

Define class Dog

Define static variable species = "Canis familiaris"

Define static variable numLegs = 4

Define instance variables: name, breed, age

Define constructor with name, breed, age

Set this.name = name

Set this.breed = breed

Set this.age = age

Define method bark()

Print "Woof!"

Create Scanner

Ask user to enter dog's name

Read name

Ask user to enter dog's breed

Read breed

Ask user to enter dog's age

Read age

Create Dog object using input values

Print dog's name, breed, age

Print species and number of legs

Call bark() method

Close Scanner

End

Algorithm: steps

1. Start
2. Declare class-level variables: species = "Canis familiaris" and numb Legs = 4
3. Declare instance variables: name, breed, age
4. Create a constructor to initialize name, breed, and age
5. Define method bark() to print "Woof!"
6. Create a Scanner object
7. Prompt the user to enter name, breed, and age of the dog
8. Create a Dog object using input values
9. Display dog details: name, breed, age, species, numb Legs
10. Call the bark() method
11. Close the Scanner
12. Stop

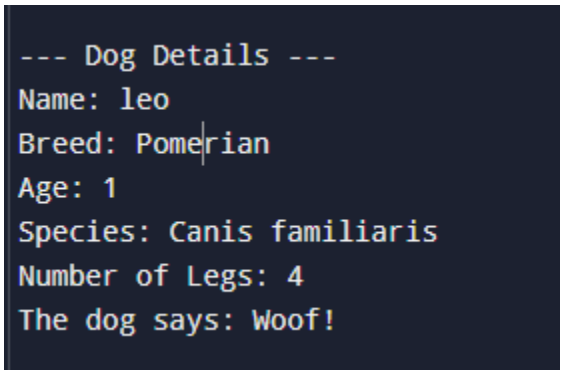
Code :

```
import java.util.Scanner;

public class Dog {
   static String species = "Canis familiaris";
   static int numLegs = 4;
```

```
    String name;
    String breed;
    int age;
    public Dog(String name, String breed, int age) {
        this.name = name;
        this.breed = breed;
        this.age = age;
    }

    public void bark() {
        System.out.println("Woof!");
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter dog's name: ");
        String name = scanner.nextLine();

        System.out.print("Enter dog's breed: ");
        String breed = scanner.nextLine();

        System.out.print("Enter dog's age: ");
        int age = scanner.nextInt();
        Dog myDog = new Dog(name, breed, age);
        System.out.println("\n--- Dog Details ---");
        System.out.println("Name: " + myDog.name);
        System.out.println("Breed: " + myDog.breed);
        System.out.println("Age: " + myDog.age);
        System.out.println("Species: " + Dog.species);
        System.out.println("Number of Legs: " + Dog.numLegs);
        System.out.print("The dog says: ");
        myDog.bark();

        scanner.close();
    }
}
```

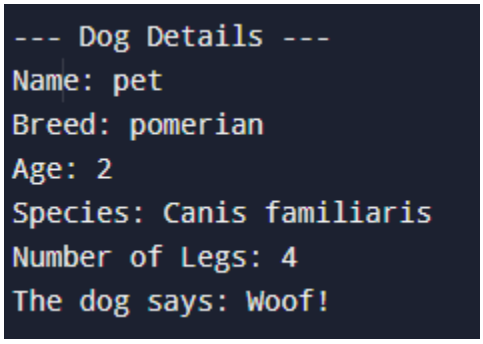
Test cases	Input	Expected Output	Actual Output	Status
TC1	Enter dog's name: leo Enter dog's breed: Pomerian Enter dog's age: 1	Name: leo Breed: Pomerian Age: 1 Species: Canis familiaris	Name: leo Breed: Pomerian Age: 1 Species: Canis familiaris Number of Legs: 4	Pass

		Number of Legs: 4 The dog says: Woof!	The dog says: Woof!	
TC2	Enter dog's name: pet Enter dog's breed: pomerian Enter dog's age: 2	Name: pet Breed: pomerian Age: 2 Species: Canis familiaris Number of Legs: 4 The dog says: Woof!	Name: pet Breed: pomerian Age: 2 Species: Canis familiaris Number of Legs: 4 The dog says: Woof!	Pass
TC3	Enter dog's name: 12 Enter dog's breed: dog Enter dog's age: df	ERROR!	ERROR!	Pass



TC1 :

TC2 :



TC3 :

```
Enter dog's name: 12
Enter dog's breed: dog
Enter dog's age: df
ERROR!
```

Observation:

In this activity, I learned how to create a simple class in Java using real-life examples. I understood the difference between class variables and object variables . I also learned how to use a constructor to set the values of an object when it is created.I practiced taking input from the user using the `Scanner` class, and I used that input to create a dog object. I also created a method called `bark()` that shows how an object can perform an action

Problem Statement 2 : Create a class name Book.

1. Inside the class, define three variables :title,author,price
2. In the main program:
    - Ask the user to enter the book's title, author, and price
    - Use these inputs to create a Book object
    - Then, call the displayDetails() method to show the entered book details

Pseudo code :

Start

Class Book:

Declare title, author, price

Constructor(title, author, price):

Set this.title = title

Set this.author = author

Set this.price = price

Method displayDetails():

Print title, author, price

Main Method:

Create Scanner

Ask user for title → Read title

Ask user for author → Read author

Ask user for price → Read pric

Create Book object using input

Call displayDetails(

Close Scanner

Stop

Algorithm: steps

1. Start
2. Define a class Book with variables: title, author, price
3. Create a constructor to initialize the values
4. Create a method displayDetails() to print book details
5. In the main() method:
6. Create Scanner object
7. Take input for title, author, and price
8. Create a Book object using the input value
9. Call displayDetails() method
10. Close the scanner
11. Stop

Code :

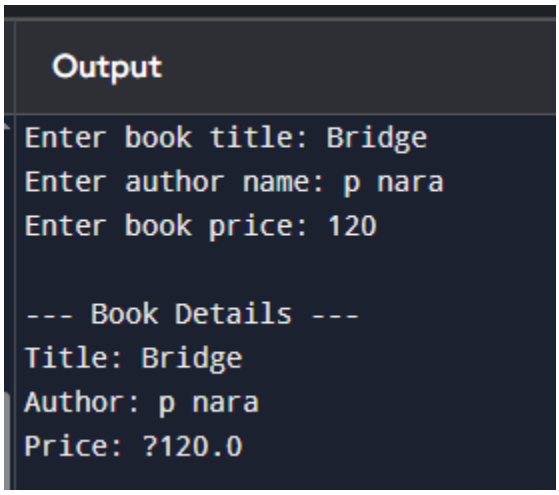
```
import java.util.Scanner;

public class Book {
   String title;
   String author;
   double price;
   public Book(String title, String author, double price) {
      this.title = title;
      this.author = author;
      this.price = price;
   }
   public void displayDetails() {
      System.out.println("\n--- Book Details ---");
      System.out.println("Title: " + title);
      System.out.println("Author: " + author);
      System.out.println("Price: " + price);
   }
```

```
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter book title: ");
        String title = scanner.nextLine();

        System.out.print("Enter author name: ");
        String author = scanner.nextLine();

        System.out.print("Enter book price: ");
        double price = scanner.nextDouble();
        Book myBook = new Book(title, author, price);

        myBook.displayDetails();

        scanner.close();
    }
}
```

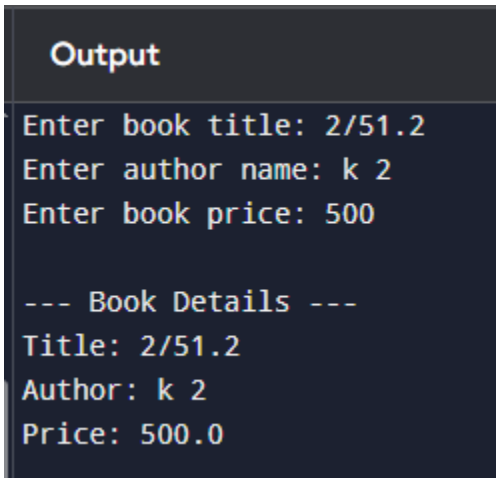
Test cases	Input	Expected Output	Actual Output	Status
TC1	Enter book title: Bridge Enter author name: p nara Enter book price: 120	Enter book title: Bridge Enter author name: p nara Enter book price: 120 --- Book Details --- Title: Bridge Author: p nara Price: ?120.0	Enter book title: Bridge Enter author name: p nara Enter book price: 120 --- Book Details --- Title: Bridge Author: p nara Price: ?120.0	Pass
TC2	Enter book title: 2/51.2 Enter author name: k 2 Enter book price: 500	Enter book title: 2/51.2 Enter author name: k 2 Enter book price: 500 --- Book Details --- Title: 2/51.2 Author: k 2 Price: 500.0	Enter book title: 2/51.2 Enter author name: k 2 Enter book price: 500 --- Book Details --- Title: 2/51.2 Author: k 2 Price: 500.0	Pass
TC3	Enter book title: alone	Enter book title: alone Enter author name: raj	Enter book title: alone Enter author name: raj Enter book price: 56.21	Pass

	Enter author name: raj Enter book price: 56.21	Enter book price: 56.21 --- Book Details --- Title: alone Author: raj Price: 56.21	 --- Book Details --- Title: alone Author: raj Price: 56.21	

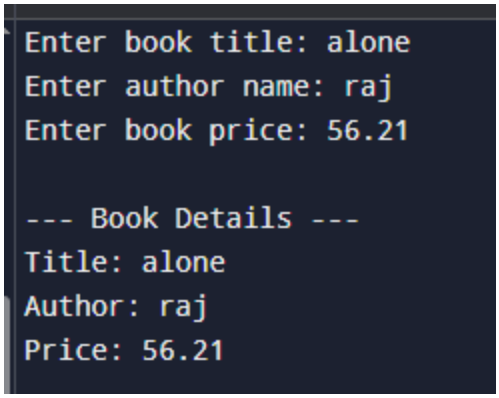
TC1 :



TC2 :



TC3 :



Observation:

I learned how to define a class with multiple attributes.I used a constructor to initialize the values for title, author, and price.I learned how to take user input using the Scanner class.I used a method displayDetails() to print the book information.The program was compiled and executed successfully.

Problem Statement 3: Create a class called Car.

- Let the user enter details like model, color, and year.
- Display all the car's details and actions.

Pseudo code :

Class car:

Static numwheels = 4

Variables: model, color, year

Method constructor(model, color, year):

Set instance variables with given values

Method startengine():

Print "<model> engine started"

Method drive():

Print "<model> is driving"

Method displayinfo():

Print model, color, year, numwheel

Main:

Ask user for model

Ask user for color

Ask user for year

Create car object with input value

Call car.displayinfo()

Call car.startengine()

Call car.drive()

End

Algorithm: steps

1. Start the program.
2. Create a class named Car with:
3. A static variable numWheels = 4.
4. Instance variables model, color, and year.
5. A constructor to initialize these values.
6. Methods: startEngine(), drive(), and displayInfo().
7. In the main() method:
8. Ask the user to enter car model, color, and year.
9. Store the inputs.
10. Create an object of the Car class using the entered values.
11. Call displayInfo() to show the car details.
12. Call startEngine() to show the engine is started.
13. Call drive() to show the car is driving.
14. End the program.

Code :

```
import java.util.Scanner;

public class Car {

   private String brand;
   private String fuelType;
```

```
private int year;

static String category = "Automobile";

public Car(String brand, String fuelType, int year) {
    this.brand = brand;
    this.fuelType = fuelType;
    this.year = year;
}


public void showDetails() {
    System.out.println("\n--- Car Details ---");
    System.out.println("Brand: " + brand);
    System.out.println("Fuel Type: " + fuelType);
    System.out.println("Year: " + year);
    System.out.println("Category: " + category);
}


public void checkAge(int currentYear) {
    int age = currentYear - year;
    if (age >= 10) {
        System.out.println("This is an old car (" + age + " years old).");
    } else {
        System.out.println("This is a relatively new car (" + age + " years old).");
    }
}

public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    System.out.print("Enter car brand: ");
    String brand = scanner.nextLine();

    System.out.print("Enter fuel type (Petrol/Diesel/Electric): ");
    String fuel = scanner.nextLine();

    System.out.print("Enter year of manufacture: ");
    int year = scanner.nextInt();

    Car myCar = new Car(brand, fuel, year);

    myCar.showDetails();
```

```
        System.out.print("\nEnter current year to check car age: ");
        int currentYear = scanner.nextInt();

        myCar.checkAge(currentYear);
    }
}
```

Test cases	Input	Expected Output	Actual Output	Status
TC1	Enter car brand: BMW Enter fuel type (Petrol/Diesel/Electric): petrol Enter year of manufacture: 2021	--- Car Details --- Brand: BMW Fuel Type: petrol Year: 2021 Category: Automobile Enter current year to check car age: 2025 This is a relatively new car (4 years old).	--- Car Details --- Brand: BMW Fuel Type: petrol Year: 2021 Category: Automobile Enter current year to check car age: 2025 This is a relatively new car (4 years old).	Pass
TC2	Enter car brand: suzuki Enter fuel type (Petrol/Diesel/Electric): adf Enter year of manufacture: 2015	--- Car Details --- Brand: suzuki Fuel Type: adf Year: 2015 Category: Automobile Enter current year to check car age: 2025 This is an old car (10 years old).	--- Car Details --- Brand: suzuki Fuel Type: adf Year: 2015 Category: Automobile Enter current year to check car age: 2025 This is an old car (10 years old).	Pass
TC3	Enter car brand: benz Enter fuel type (Petrol/Diesel/Electric): diesel Enter year of manufacture: 2025	--- Car Details --- Brand: benz Fuel Type: diesel Year: 2025 Category: Automobile Enter current year to check car age: 2025 This is a relatively new car (0 years old).	--- Car Details --- Brand: benz Fuel Type: diesel Year: 2025 Category: Automobile Enter current year to check car age: 2025 This is a relatively new car (0 years old).	Pass

TC1 :

```
Enter car brand: BMW
Enter fuel type (Petrol/Diesel/Electric): petrol
Enter year of manufacture: 2021

--- Car Details ---
Brand: BMW
Fuel Type: petrol
Year: 2021
Category: Automobile

Enter current year to check car age: 2025
This is a relatively new car (4 years old).
```

TC2 :

```
Enter car brand: suzuki
Enter fuel type (Petrol/Diesel/Electric): adf
Enter year of manufacture: 2015

--- Car Details ---
Brand: suzuki
Fuel Type: adf
Year: 2015
Category: Automobile

Enter current year to check car age: 2025
This is an old car (10 years old).
```

TC3 :

```
Enter car brand: benz
Enter fuel type (Petrol/Diesel/Electric): diesel
Enter year of manufacture: 2025

--- Car Details ---
Brand: benz
Fuel Type: diesel
Year: 2025
Category: Automobile

Enter current year to check car age: 2025
This is a relatively new car (0 years old).
```

Observation:

I learned how to create and store multiple objects using an array of objects.I used a for loop to take input for multiple employees.I practiced storing details like ID, name, and salary for each employee.The program uses a constructor to initialize data and a method to display it.The program executes successfully and displays all employee details one by one.

Section 2 :

Problem Statement 1 : Create a class for a Dog.
Then, make two dogs:

- On named Buddy, breed Golden Retriever, age 5
- Another named Lucy, breed Poodle, age 2

After creating them:

- Print their name and age
- Make both dogs bark

Pseudo code :

Start

Class Dog:

Variables: name, breed, age

Constructor(name, breed, age):

    Set this.name = name

    Set this.breed = breed

    Set this.age = age

Method bark():

    Print name + " says: Woof!"

Method displayInfo():

    Print name and age

Main Method:

Create Scanner

Input Dog 1:

    Ask name, breed, age → store in name1, breed1, age1

Input Dog 2:

    Ask name, breed, age → store in name2, breed2, age2

Create Dog objects dog1 and dog2

Call displayInfo() and bark() for dog1 and dog 2

Close Scanner

Stop

Algorithm: steps

1. Start
2. Create a class Dog with attributes: name, breed, and age
3. Create a constructor to initialize the dog's values
4. Create a method bark() to print the dog's name with "Woof!"
5. Create a method displayInfo() to show name and age
6. In the main() method:

7. Create Scanner for user inpu
8. Take input for Dog 1 → name, breed, age
9. Take input for Dog 2 → name, breed, age
10. Create 2 Dog objects using the inputs
11. Call displayInfo() and bark() for both dogs
12. Close the Scanner
13. Stop

Code :

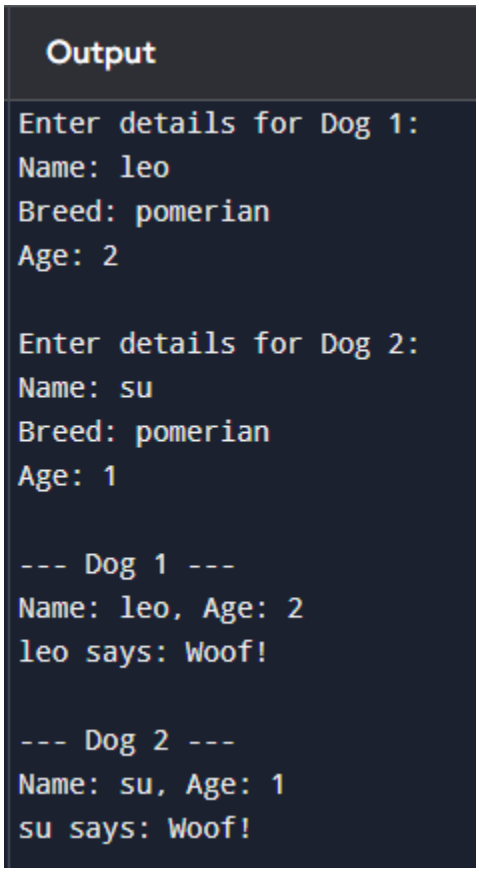
```
import java.util.Scanner;

public class Dog {

    String name;
    String breed;
    int age;

    public Dog(String name, String breed, int age) {
        this.name = name;
        this.breed = breed;
        this.age = age;
    }

    public void bark() {
        System.out.println(name + " says: Woof!");
    }

    public void displayInfo() {
        System.out.println("Name: " + name + ", Age: " + age);
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter details for Dog 1:");
        System.out.print("Name: ");
        String name1 = scanner.nextLine();
        System.out.print("Breed: ");
        String breed1 = scanner.nextLine();
        System.out.print("Age: ");
        int age1 = scanner.nextInt();
        scanner.nextLine();
        System.out.println("\nEnter details for Dog 2:");
        System.out.print("Name: ");
```

```
        String name2 = scanner.nextLine();
        System.out.print("Breed: ");
        String breed2 = scanner.nextLine();
        System.out.print("Age: ");
        int age2 = scanner.nextInt();
        Dog dog1 = new Dog(name1, breed1, age1);
        Dog dog2 = new Dog(name2, breed2, age2);
        System.out.println("\n--- Dog 1 ---");
        dog1.displayInfo();
        dog1.bark();
        System.out.println("\n--- Dog 2 ---");
        dog2.displayInfo();
        dog2.bark();

        scanner.close();
    }
}
```

Test cases	Input	Expected Output	Actual Output	Status
TC1	Enter details for Dog 1: Name: leo Breed: pomerian Age: 2 Enter details for Dog 2: Name: su Breed: pomerian Age: 1	--- Dog 1 --- Name: leo, Age: 2 leo says: Woof! --- Dog 2 --- Name: su, Age: 1 su says: Woof!	--- Dog 1 --- Name: leo, Age: 2 leo says: Woof! --- Dog 2 --- Name: su, Age: 1 su says: Woof!	Pass
TC2	Enter details for Dog 1: Name: 12 Breed: 2 Age: 1 Enter details for Dog 2: Name: 56 Breed: 2 Age: 1	--- Dog 1 --- Name: 12, Age: 1 12 says: Woof! --- Dog 2 --- Name: 56, Age: 1 56 says: Woof!	--- Dog 1 --- Name: 12, Age: 1 12 says: Woof! --- Dog 2 --- Name: 56, Age: 1 56 says: Woof!	Pass

TC3	Enter details for Dog 1: Name: k Breed: 12 Age: 1 Enter details for Dog 2: Name: p Breed: breed Age: 2	--- Dog 1 --- Name: k, Age: 1 k says: Woof! --- Dog 2 --- Name: p, Age: 2 p says: Woof!	--- Dog 1 --- Name: k, Age: 1 k says: Woof! --- Dog 2 --- Name: p, Age: 2 p says: Woof!	Pass

TC1 :



TC2 :

```
Output
Enter details for Dog 1:
Name: 12
Breed: 2
Age: 1

Enter details for Dog 2:
Name: 56
Breed: 2
Age: 1

--- Dog 1 ---
Name: 12, Age: 1
12 says: Woof!

--- Dog 2 ---
Name: 56, Age: 1
56 says: Woof!
```

TC3 :

```
Output
Enter details for Dog 1:
Name: k
Breed: 12
Age: 1

Enter details for Dog 2:
Name: p
Breed: breed
Age: 2

--- Dog 1 ---
Name: k, Age: 1
k says: Woof!

--- Dog 2 ---
Name: p, Age: 2
p says: Woof!
```

Observation:

I learned how to create multiple objects using the same class.I took input for two dogs, each with name, breed, and age.I used a constructor to set values and methods to print info.The `bark()` method shows how to give behavior to an object.The program helped me understand how to use methods and constructors with real user input.

Problem Statement 2 : Create a class called Book. Each book should have:Title,Author,A status: whether it is open or close

Then:

- Create two book objects using a constructor.
- Add a method displayStatus() that prints:
    - If the book is open: "Title by Author is Open"
    - If the book is closed: "Title by Author is Closed"

Pseudo code :

Start

Class Book:

Variables: title, author, isOpen

Constructor(title, author, isOpen):

Set this.title = title

Set this.author = author

Set this.isOpen = isOpen

Method displayStatus():

If isOpen is true

Print "<title> by <author> is Open"

Else

Print "<title> by <author> is Closed"

Main Method:

Create Scanner

Input Book 1:

Ask for title, author, isOpen → save in title1, author1, open1

Input Book 2:

Ask for title, author, isOpen → save in title2, author2, open2

Create Book objects book1 and book2

Call displayStatus() for book1

Call displayStatus() for book2

Close Scanner

Stop

Algorithm: steps

1. Start
2. Create a class Book with variables: title, author, isOpen
3. Create a constructor to initialize book details
4. Create a method displayStatus() to check if the book is open or closed
5. In the main() method:
6. Create Scanner for user input
7. Take input for title, author, and isOpen for Book 1
8. Take input for title, author, and isOpen for Book 2
9. Create two Book objects using the input
10. Call displayStatus() for both books
11. Close the scanner
12. Stop

Code :

```
import java.util.Scanner;
public class Book {
   String title;
   String author;
   boolean isOpen;
```

```
    public Book(String title, String author, boolean isOpen) {
        this.title = title;
        this.author = author;
        this.isOpen = isOpen;
    }

    public void displayStatus() {
        String status = isOpen ? "Open" : "Closed";
        System.out.println(title + " by " + author + " is " + status);
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter details for Book 1:");
        System.out.print("Title: ");
        String title1 = scanner.nextLine();
        System.out.print("Author: ");
        String author1 = scanner.nextLine();
        System.out.print("Is the book open?  ");
        boolean open1 = scanner.nextBoolean();
        scanner.nextLine();
        System.out.println("\nEnter details for Book 2:");
        System.out.print("Title: ");
        String title2 = scanner.nextLine();
        System.out.print("Author: ");
        String author2 = scanner.nextLine();
        System.out.print("Is the book open? : ");
        boolean open2 = scanner.nextBoolean();

        Book book1 = new Book(title1, author1, open1);
        Book book2 = new Book(title2, author2, open2);
        System.out.println("\n--- Book Status ---");
        book1.displayStatus();
        book2.displayStatus();
        scanner.close();
    }
}
```

Test cases	Input	Expected Output	Actual Output	Status

TC1	Enter details for Book 1: Title: alone Author: clone Is the book open? true Enter details for Book 2: Title: clone Author: k Is the book open? : false	--- Book Status --- alone by clone is Open clone by k is Closed	--- Book Status --- alone by clone is Open clone by k is Closed	Pass
TC2	Enter details for Book 1: Title: self Author: Mr k Is the book open? true Enter details for Book 2: Title: presence Author: Mr s Is the book open? : false	--- Book Status --- self by Mr k is Open presence by Mr s is Closed	--- Book Status --- self by Mr k is Open presence by Mr s is Closed	Pass
TC3	Enter details for Book 1: Title: silence Author: @123 Is the book open? true Enter details for Book 2: Title: presence Author: @45 Is the book open? : false	--- Book Status --- silence by @123 is Open presence by @45 is Closed	--- Book Status --- silence by @123 is Open presence by @45 is Closed	Pass

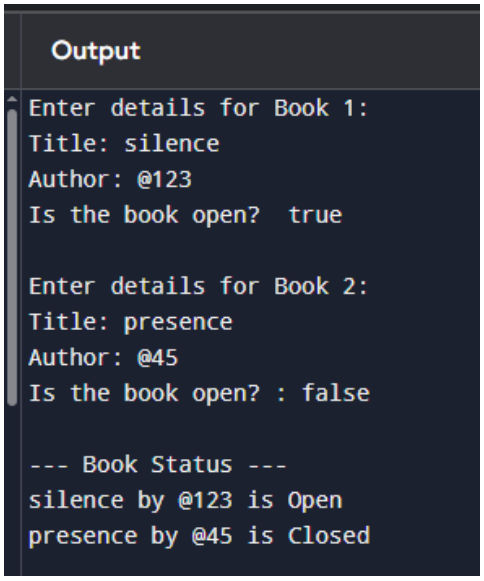
TC1 :

```
Output
Enter details for Book 1:
Title: alone
Author: clone
Is the book open?  true

Enter details for Book 2:
Title: clone
Author: k
Is the book open? : false

--- Book Status ---
alone by clone is Open
clone by k is Closed
```

TC2 :

```
Output
Enter details for Book 1:
Title: self
Author: Mr k
Is the book open?  true

Enter details for Book 2:
Title: presence
Author: Mr s
Is the book open? : false

--- Book Status ---
self by Mr k is Open
presence by Mr s is Closed
```

TC3 :



Observation:

I learned how to take boolean input (true/false) from the user.I created two Book objects using a constructor with different values.The method `displayStatus()` helped me practice if-else condition.The program checks if the book is open or closed and prints the correct message.This activity helped me understand how to use boolean logic with objects and methods.

Problem Statement 3 : (Student Record) Create three Student objects and print their info using different Variables like String name; String idNumber; String major;.

Pseudo code :

Start

Define student class with name, id, major

Create constructor to initialize values

Create getinfo() to return formatted string

In main():

create array to store 3 student objects

for each student:

prompt user to enter name, id, major

create student object and store in array

print all student information using getinfo()

End

Algorithm: steps

1. Start
2. Define a Student class with name, idNumber, major
3. Define a constructor to initialize those fields
4. Define getInfo() method to return student details
5. In main():
    - Create array of 3 students
    - Loop 3 times to get user input (name, id, major)
    - Store each new Student in the array
6. Loop through the array and print each student's info using getInfo()
7. End

Code :

```
import java.util.Scanner;

public class Student {
   String name;
   String idNumber;
   String major;

   public Student(String name, String idNumber, String major) {
      this.name = name;
      this.idNumber = idNumber;
      this.major = major;
   }

   public String getInfo() {
      return name + ", ID: " + idNumber + ", Major: " + major;
   }

   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      Student[] students = new Student[3];

      for (int i = 0; i < 3; i++) {
         System.out.println("Enter details for Student " + (i + 1) + ":");

         System.out.print("Name: ");
```

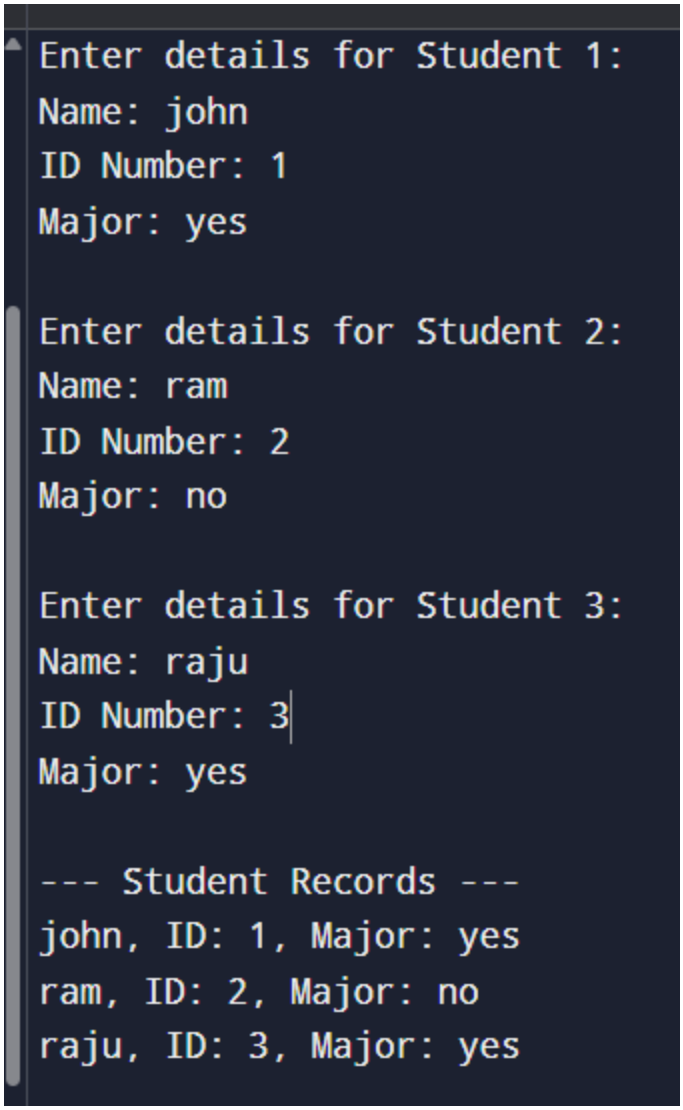
```
            String name = scanner.nextLine();

            System.out.print("ID Number: ");
            String id = scanner.nextLine();

            System.out.print("Major: ");
            String major = scanner.nextLine();

            students[i] = new Student(name, id, major);
            System.out.println();
        }

        System.out.println("----- Student Records -----");
        for (Student student : students) {
            System.out.println(student.getInfo());
        }
    }
}
```

Test Cases:

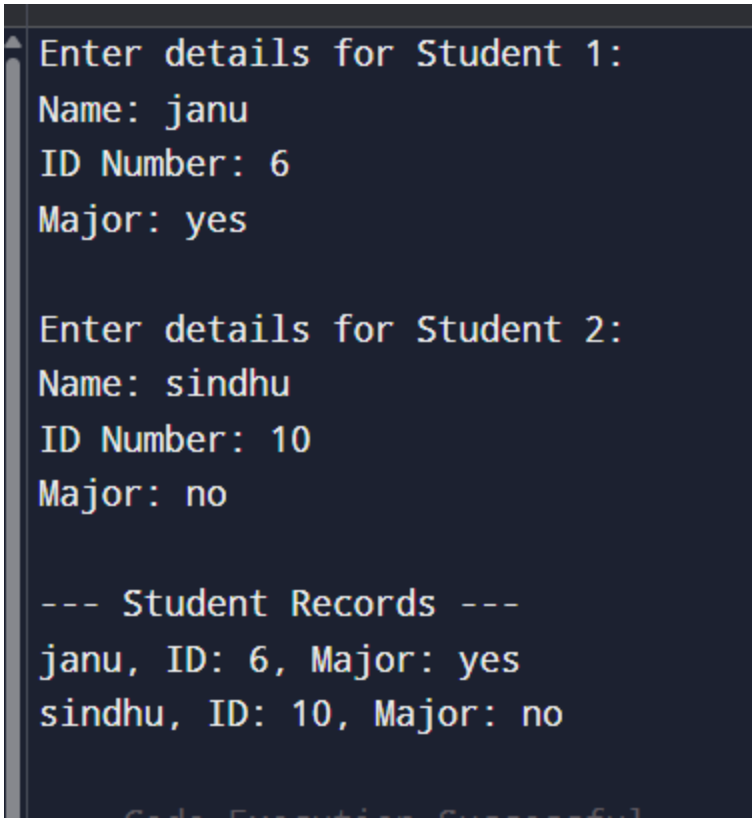
Test cases	Input	Expected Output	Actual Output	Status
TC1	Enter details for Student 1: Name: john ID Number: 1 Major: yes Enter details for Student 2: Name: ram ID Number: 2 Major: no Enter details for Student 3: Name: raju ID Number: 3 Major: yes	--- Student Records --- john, ID: 1, Major: yes ram, ID: 2, Major: no raju, ID: 3, Major: yes	--- Student Records --- john, ID: 1, Major: yes ram, ID: 2, Major: no raju, ID: 3, Major: yes	pass

TC2	Enter details for Student 1: Name: janu ID Number: 6 Major: yes Enter details for Student 2: Name: sindhu ID Number: 10 Major: no	--- Student Records --- janu, ID: 6, Major: yes sindhu, ID: 10, Major: no	--- Student Records --- janu, ID: 6, Major: yes sindhu, ID: 10, Major: no	pass
TC3	Enter details for Student 1: Name: anu ID Number: 19 Major: no	--- Student Records --- anu, ID: 19, Major: no	--- Student Records --- anu, ID: 19, Major: no	pass

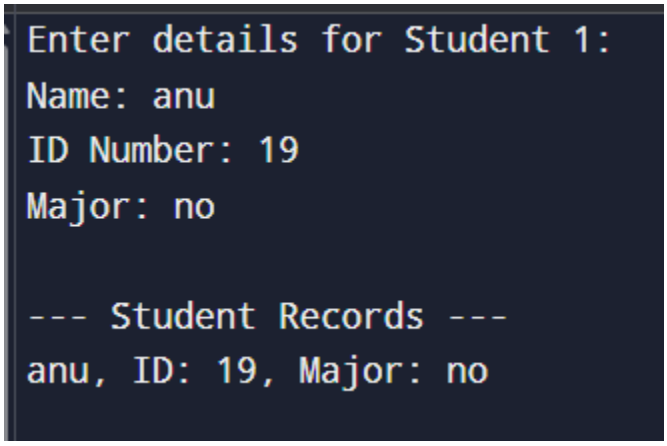
TC1 :


Enter details for Student 1:
Name: john
ID Number: 1
Major: yes
Enter details for Student 2:
Name: ram
ID Number: 2
Major: no
Enter details for Student 3:
Name: raju
ID Number: 3
Major: yes
--- Student Records ---
john, ID: 1, Major: yes
ram, ID: 2, Major: no
raju, ID: 3, Major: yes

TC2 :



TC3 :



Observation: The Student program demonstrates how to use constructors and arrays of objects in Java. It takes user input to create multiple Student objects and prints their details using a method. It also shows how to use this keyword to assign values properly and emphasizes object-oriented programming concepts like encapsulation and data grouping.

SECTION 3 :

Problem Statement 1 : (Bank Account) Create a different method for BankAccount class with checking the balance, deposit the amount and withdraw and also when the user enters the negative value for deposit or withdrawal it will show invalid operations.

Pseudo code :

Start

Create bank account with balance

Define deposit(), withdraw(), getbalance()

In main():

loop:

show menu

get user choice

if 1 → show balance

if 2 → get deposit amount → deposit()

if 3 → get withdrawal amount → withdraw()

if 4 → exit

else → invalid option

End

Algorithm: steps

1. Start
2. Create account with ₹2000
3. Repeat:
    - Show menu
    - Get user input
    - Perform action (check, deposit, withdraw, exit)
4. End when user selects Exit

Code :

```
import java.util.Scanner;
```

```
public class BankAccount {
    private double balance;

    public BankAccount(double initialBalance) {
        if (initialBalance >= 0)
            this.balance = initialBalance;
        else
            this.balance = 0;
    }

    public double getBalance() {
        return balance;
    }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited: " + amount);
        } else {
            System.out.println("Invalid deposit amount. Must be positive.");
        }
    }

    public void withdraw(double amount) {
        if (amount <= 0) {
            System.out.println("Invalid withdrawal amount.");
        } else if (amount > balance) {
            System.out.println("Insufficient balance.");
        } else {
            balance -= amount;
            System.out.println("Withdrew: " + amount);
        }
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        BankAccount account = new BankAccount(2000);

        while (true) {
            System.out.println("--Bank Menu--:");
            System.out.println("1. Check Balance");
            System.out.println("2. Deposit");
```

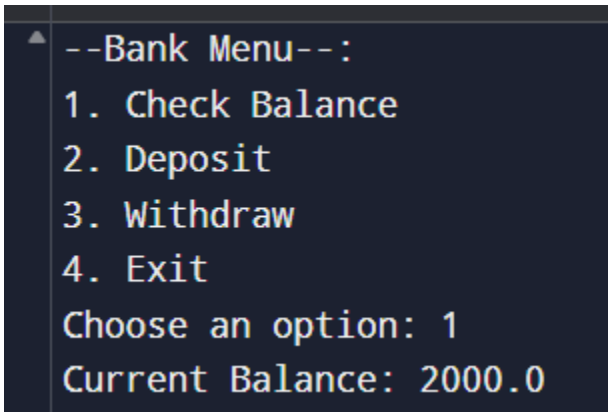
```
            System.out.println("3. Withdraw");
            System.out.println("4. Exit");

            System.out.print("Choose an option: ");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.println("Current Balance: " + account.getBalance());
                    break;

                case 2:
                    System.out.print("Enter deposit amount: ");
                    double depositAmount = scanner.nextDouble();
                    account.deposit(depositAmount);
                    break;

                case 3:
                    System.out.print("Enter withdrawal amount: ");
                    double withdrawAmount = scanner.nextDouble();
                    account.withdraw(withdrawAmount);
                    break;

                case 4:
                    System.out.println("Thank you! Exiting.");
                    return;

                default:
                    System.out.println("Invalid option. Please try again.");
            }
        }
    }
}
```

Test Cases:

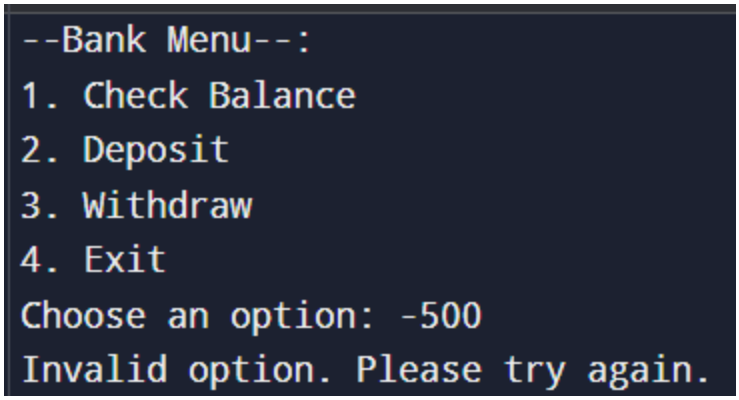
Test cases	Input	Expected Output	Actual Output	Status
TC1	--Bank Menu--: 1. Check Balance	Current Balance: 2000.0	Current Balance: 2000.0	pass

	2. Deposit 3. Withdraw 4. Exit Choose an option: 1			
TC2	--Bank Menu--: 1. Check Balance 2. Deposit 3. Withdraw 4. Exit Choose an option: -500	Invalid option. Please try again.	Invalid option. Please try again.	pass
TC3	--Bank Menu--: 1. Check Balance 2. Deposit 3. Withdraw 4. Exit Choose an option: 2	--Bank Menu--: 1. Check Balance 2. Deposit 3. Withdraw 4. Exit Choose an option: 2 Enter deposit amount: 2000 Deposited: 2000.0	--Bank Menu--: 1. Check Balance 2. Deposit 3. Withdraw 4. Exit Choose an option: 2 Enter deposit amount: 2000 Deposited: 2000.0	pass

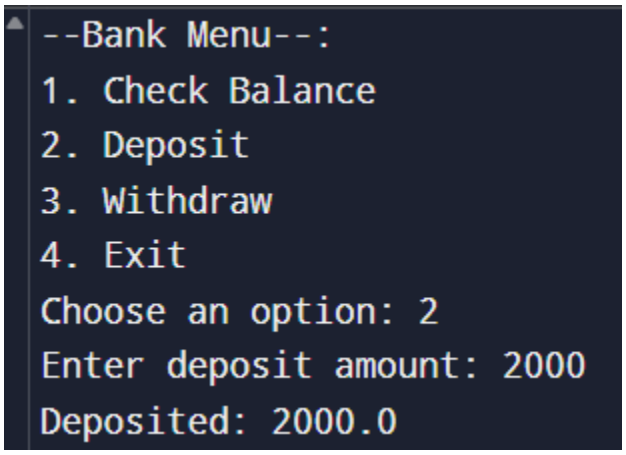
TC1 :



TC2 :



TC3 :



Observation: The Bank Account program demonstrates basic object-oriented programming concepts like encapsulation, method calling, and user input handling. It ensures valid transactions by checking for negative deposits and excessive withdrawals, making the program safe, interactive, and user-friendly.

Problem Statement 2 : (Product Inventory) Create a product Use getters and setters Validate constraints (like no negative price or quantity) Calculate total value.

Pseudo code :

Start

Ask user for product name, price, and quantity

Create product object with these values

Inside product class:

store name, price, quantity

setprice(): allow only positive price

setquantity(): allow zero or more

gettotalvalue(): return price × quantity

Print product name, price, quantity, and total value

End

Algorithm: steps

1. Start
2. Prompt user to enter product name, price, and quantity
3. Create a Product object with given input
4. In Product class:
    - Use constructor to initialize fields using setters
    - Validate price and quantity
5. Display:
    - Name
    - Price
    - Quantity
    - Total value (price × quantity)
6. End

Code :

```
import java.util.Scanner;

public class Product {
   private String name;
   private double price;
   private int quantity;

   public Product(String name, double price, int quantity) {
      this.name = name;
      setPrice(price);
      setQuantity(quantity);
   }

   public String getName() {
      return name;
   }
```

```
public double getPrice() {
    return price;
}

public int getQuantity() {
    return quantity;
}

public void setPrice(double price) {
    if (price > 0) {
        this.price = price;
    } else {
        System.out.println("Invalid price! Must be positive.");
    }
}

public void setQuantity(int quantity) {
    if (quantity >= 0) {
        this.quantity = quantity;
    } else {
        System.out.println("Invalid quantity! Must be zero or more.");
    }
}
public double getTotalValue() {
    return price * quantity;
}

public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    System.out.print("Enter product name: ");
    String name = scanner.nextLine();

    System.out.print("Enter product price: ");
    double price = scanner.nextDouble();

    System.out.print("Enter product quantity: ");
    int quantity = scanner.nextInt();

    Product product = new Product(name, price, quantity);

    System.out.println("--- Product Info ---");
    System.out.println("Name: " + product.getName());
    System.out.println("Price: " + product.getPrice());
```

```
        System.out.println("Quantity: " + product.getQuantity());
        System.out.println("Total Value: " + product.getTotalValue());
    }
}
```

Test Cases:

Test cases	Input	Expected Output	Actual Output	Status
TC1	Enter product name: ice cream Enter product price: 30 Enter product quantity: 5	--- Product Info --- Name: ice cream Price: 30.0 Quantity: 5 Total Value: 150.0	--- Product Info --- Name: ice cream Price: 30.0 Quantity: 5 Total Value: 150.0	pass
TC2	Enter product name: pen Enter product price: -10 Enter product quantity: 5 Invalid price! Must be positive.	--- Product Info --- Name: pen Price: 0.0 Quantity: 5 Total Value: 0.0	--- Product Info --- Name: pen Price: 0.0 Quantity: 5 Total Value: 0.0	pass
TC3	Enter product name: rose Enter product price: 5.5 Enter product quantity: 10	--- Product Info --- Name: rose Price: 5.5 Quantity: 10 Total Value: 55.0	--- Product Info --- Name: rose Price: 5.5 Quantity: 10 Total Value: 55.0	pass

TC1 :

```
Enter product name: ice cream
Enter product price: 30
Enter product quantity: 5
--- Product Info ---
Name: ice cream
Price: 30.0
Quantity: 5
Total Value: 150.0
```

TC2 :

```
Enter product name: pen
Enter product price: -10
Enter product quantity: 5
Invalid price! Must be positive.
--- Product Info ---
Name: pen
Price: 0.0
Quantity: 5
Total Value: 0.0
```

TC3 :

```
Enter product name: rose
Enter product price: 5.5
Enter product quantity: 10
--- Product Info ---
Name: rose
Price: 5.5
Quantity: 10
Total Value: 55.0
```

Observation:

The Product program demonstrates encapsulation by using private fields with getters and setters. It validates user input to ensure the price is positive and quantity is non-negative, ensuring data integrity. The program also calculates the total value (price × quantity) and displays all product details interactively using user input.