03-Class and Object Practical

We will first write a simple program to add two numbers in the Demo class. Then, we will create a separate Calculator class to handle the addition functionality. This will make our code modular and reusable.

Simple Addition in Demo Class

```
public class Demo {
    public static void main(String[] args) {
        int num1 = 4;
        int num2 = 5;

        int result = num1 + num2;
        System.out.println(result);
    }
}
```

Creating the Calculator Class

We will create a Calculator class with an add method to handle the addition. This will allow us to reuse the Calculator class for various addition operations.

```
public class Calculator {

    // Method to add two numbers
    public int add(int n1, int n2) {
        System.out.println("In add method");
        int res = n1 + n2;
        return res;
    }
}
```

Accessing the Calculator Class Method in Demo Class

To use the add method from the Calculator class in the Demo class, we need to create an object of the Calculator class. This is done using the new keyword.

Complete Code

```
Calculator Class
public class Calculator {


    // Method to add two numbers
    public int add(int n1, int n2) {
        System.out.println("In add method");
        int res = n1 + n2;
        return res;
    }
}
```

Demo Class

```
public class Demo {
    public static void main(String[] args) {
        int num1 = 4;
        int num2 = 5;
        // Creating an object of the Calculator class
        Calculator calc = new Calculator();


        // Calling the add method of Calculator class
        int result = calc.add(num1, num2);
        // Printing the result
        System.out.println("The sum is: " + result);
    }
}
```

TELUSKO

Explanation of Code Using Steps

1. Define the Calculator Class
    - Class Definition: public class Calculator { }
    - Method to Add Two Numbers:

```
public int add(int n1, int n2) {
    System.out.println("In add method");
    int res = n1 + n2;
    return res;
}
```

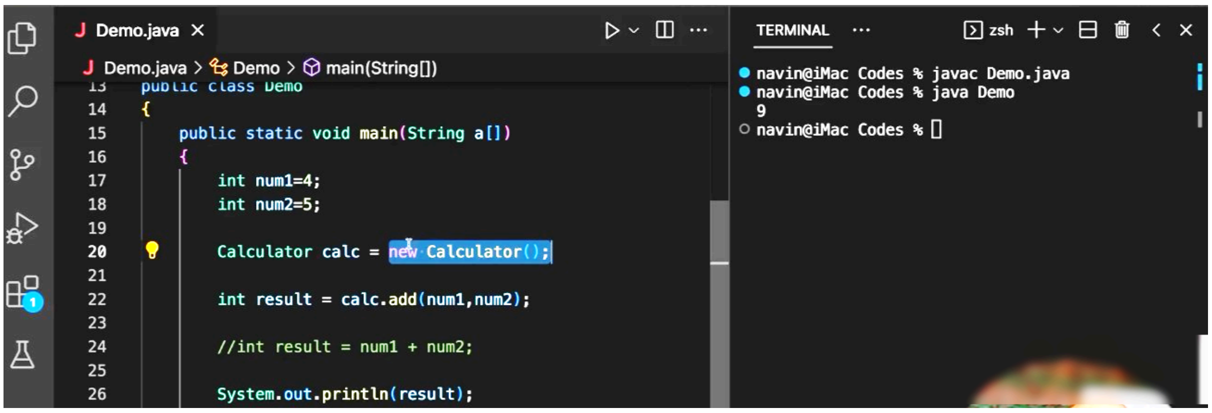
2. Create the Demo Class
    - Class Definition: public class Demo { }
    - Main Method: public static void main(String[] args) { }
3. Initialize Variables in Main Method
    - Define Variables:

```
int num1 = 4;
int num2 = 5;
```

4. Create an Object of the Calculator Class
    - Object Creation: Calculator calc = new Calculator();
        - Explanation:
            - Calculator is the class name.
            - calc is a reference variable (similar to how num1 and num2 are variables, but it references an object).
            - new Calculator() creates a new object of the Calculator class.
            - The JVM handles the creation of the object and allocates memory for it.
5. Call the Add Method Using the Calculator Object

- Method Call: int result = calc.add(num1, num2);
  - Explanation:
    - calc.add(num1, num2) calls the add method of the Calculator class using the calc object.
    - The method adds num1 and num2, prints "In add method", and returns the result.

6. Print the Result
   - Output: System.out.println("The sum is: " + result);
     - This line prints the sum of num1 and num2.



```java
public class Demo
{
    public static void main(String a[])
    {
        int num1=4;
        int num2=5;

        Calculator calc = new Calculator();

        int result = calc.add(num1,num2);

        //int result = num1 + num2;

        System.out.println(result);
```

```
navin@iMac Codes % javac Demo.java
navin@iMac Codes % java Demo
9
navin@iMac Codes %
```

Summary

By separating the addition functionality into the Calculator class, we make our code more modular and reusable. The Demo class then creates an object of the Calculator class and calls the add method to perform the addition. This approach follows the principles of object-oriented programming and promotes better code organization and reuse.