

TASK 2: Variables, Data Types & Console Input Application.

Variables :- A variable in Java is a named memory location used to store data values that can be changed during program execution.

Variable = name + data type + value

Why do we need variables?

Variables are used to:

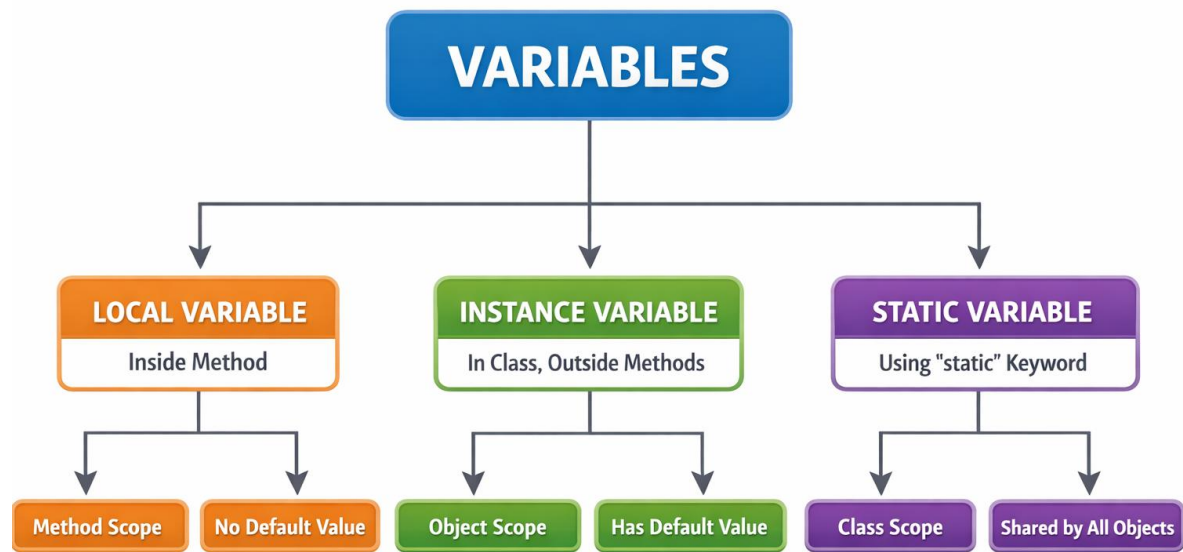
- Store user input
- Store calculation results
- Hold program state
- Reuse data efficiently

Without variables, programs cannot perform meaningful operations.

Syntax of Variable Declaration:- `dataType variableName = value;`

- Example:- `int age = 20;`
- `int` → data type
- `age` → variable name
- `20` → value

Types of Variables in Java:-



Local Variable:- A local variable is declared inside a method, constructor, or block and is accessible only within that block.

Characteristics:

- No default value
- Must be initialized before use
- Stored in stack memory
- Short lifetime

Example Code:

```
class Example
{
    public void show()
    {
        int x = 10; // local variable
        System.out.println(x);
    }
}
```

Instance Variable:-An instance variable is declared inside a class but outside any method. Each object gets its own copy of instance variables.

Characteristics:

- Has default value
- Stored in heap memory
- Lifetime = object lifetime
- Accessed using object

Example Code:

```
class Student {  
  
    int age; // instance variable  
  
    public void display() {  
        System.out.println(age);  
    }  
  
    public static void main(String[] args) {  
        Student s1 = new Student();  
        s1.display(); // prints 0 (default value)  
    }  
}
```

Static Variable (Class Variable):-A static variable is declared using the static keyword. It is shared among all objects of the class.

Characteristics:

- Single copy
- Memory efficient
- Stored in method area
- Accessed using class name

Example Code:

```
class Company {  
  
    static String companyName = "Tech Corp";  
  
    public static void main(String[] args) {
```

```
        System.out.println(Company.companyName);
    }
}
```

What is a Data Type in Java?

A data type specifies:

- What kind of data a variable can store
- How much memory it occupies
- What operations can be performed on it

Without data types, Java cannot manage memory efficiently.

Classification of Data Types in Java

- 1.) Primitive Data Types
- 2.) Non-Primitive Data Types

Example Code (Primitive Types)

```
class PrimitiveDemo
{
    public static void main(String[] args) {
        int age = 20;
        double salary = 25000.50;
        char grade = 'A';
        boolean isActive = true;

        System.out.println(age);
        System.out.println(salary);
        System.out.println(grade);
        System.out.println(isActive);
    }
}
```

Non-Primitive Data Types:-

- String

- Array
- Class
- Interface
- Object

String Example

```
class StringDemo
{
    public static void main(String[] args) {
        String name = "Akash";
        System.out.println(name);
    }
}
```

Array Example

```
class ArrayDemo
{
    public static void main(String[] args) {
        int[] marks = {85, 90, 88};
        System.out.println(marks[0]);
    }
}
```

What is Type Casting?

Converting one data type into another.

- Implicit: small → large (int → double)
- Explicit: large → small (double → int)

What is a Console Input Application?

A console input application is a program that:

- Takes input from the keyboard (console)
- Processes the data
- Displays output in the console window
- Canner Class (Core of Console Input)

- Step 1:
- Import Scanner

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

Step 2:

Create Scanner Object

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

Step 3:

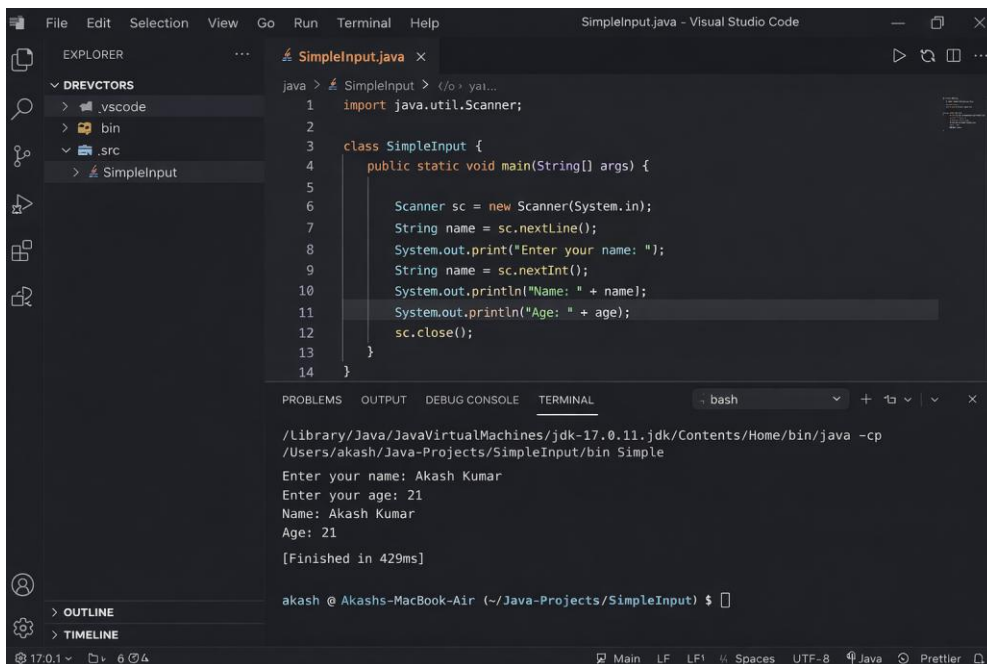
Read Input

```
int a = sc.nextInt();
```

```
double b = sc.nextDouble();
```

```
String name = sc.next();
```

Example :- Simple Console Input Program



The screenshot displays the Visual Studio Code interface with a Java file named `SimpleInput.java` open. The code defines a `SimpleInput` class with a `main` method that uses a `Scanner` to read input from `System.in`. The terminal window at the bottom shows the command to compile and run the program, followed by the user's input for name and age, and the program's output.

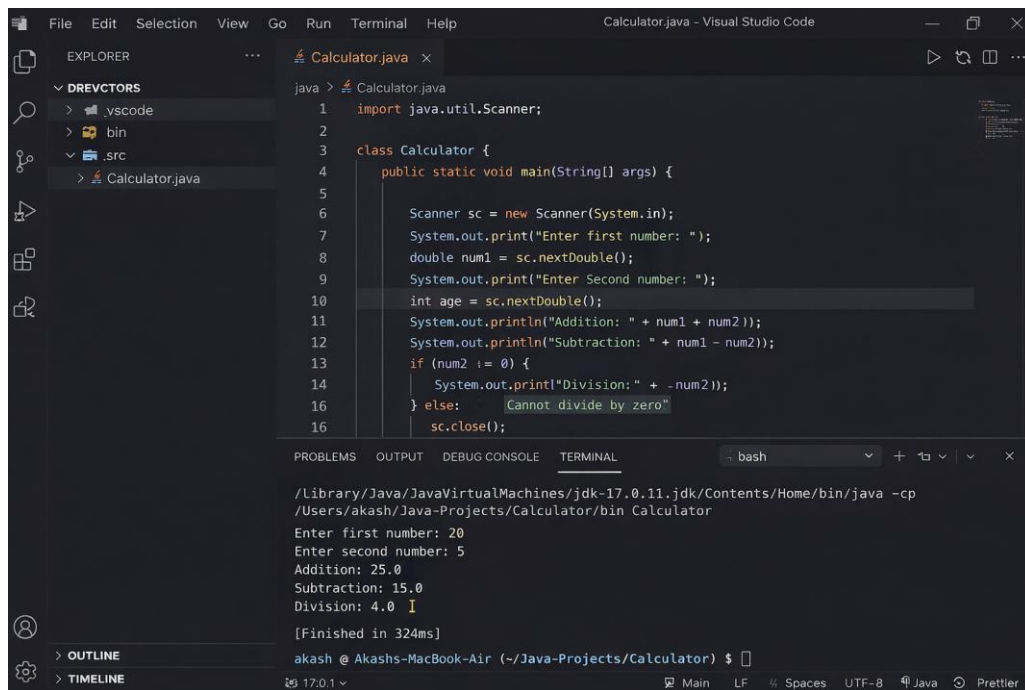
```
java > SimpleInput > </o> yat...
1  import java.util.Scanner;
2
3  class SimpleInput {
4      public static void main(String[] args) {
5
6          Scanner sc = new Scanner(System.in);
7          String name = sc.nextLine();
8          System.out.print("Enter your name: ");
9          String name = sc.nextInt();
10         System.out.println("Name: " + name);
11         System.out.println("Age: " + age);
12         sc.close();
13     }
14 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL - bash + - | X

```
/Library/Java/JavaVirtualMachines/jdk-17.0.11.jdk/Contents/Home/bin/java -cp
/Users/akash/Java-Projects/SimpleInput/bin Simple
Enter your name: Akash Kumar
Enter your age: 21
Name: Akash Kumar
Age: 21
[Finished in 429ms]

akash @ Akashs-MacBook-Air (~/.Java-Projects/SimpleInput) $
```

Example : Console-Based Calculator



The image shows a Visual Studio Code editor window titled "Calculator.java - Visual Studio Code". The Explorer sidebar on the left shows a project structure with folders ".vscode", "bin", and ".src", and a file "Calculator.java" under ".src". The main editor displays the code for "Calculator.java".

```
1 import java.util.Scanner;
2
3 class Calculator {
4     public static void main(String[] args) {
5
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter first number: ");
8         double num1 = sc.nextDouble();
9         System.out.print("Enter Second number: ");
10        int age = sc.nextDouble();
11        System.out.println("Addition: " + num1 + num2);
12        System.out.println("Subtraction: " + num1 - num2);
13        if (num2 != 0) {
14            System.out.println("Division: " + num1 / num2);
15        } else {
16            System.out.println("Cannot divide by zero");
17        }
18        sc.close();
19    }
20 }
```

The bottom panel shows the TERMINAL output for the command `java -cp /Users/akash/Java-Projects/Calculator/bin Calculator`. The output shows the program running and taking user input:

```
/Library/Java/JavaVirtualMachines/jdk-17.0.11.jdk/Contents/Home/bin/java -cp
/Users/akash/Java-Projects/Calculator/bin Calculator
Enter first number: 20
Enter second number: 5
Addition: 25.0
Subtraction: 15.0
Division: 4.0
[Finished in 324ms]
akash @ Akashs-MacBook-Air (~/.Java-Projects/Calculator) $
```

The status bar at the bottom indicates the file is "Main", uses "LF" line endings, "UTF-8" encoding, and "Java" language.