JAVA PROGRAMMING LANGUAGE

OOP :-

Java :-

- Java is a high-level, object-oriented, platform-independent programming language used to build applications for computers, mobile devices.
- The Java Development Kit is a software development kit that required to develop java applications.
- The Java Runtime Environment is a software package that provides libraries and other components needed to run Java programs.
- The JVM is a virtual machine that responsible for runs Java bytecode and converts it into machine code that your computer understands.

◆ Examples :-

```
public class HelloWorld {
   public static void main(String[ ] args) {
      System.out.print("one");    <!-- Use when 2 or more output print in single line -->
      System.out.println("Hello, Java!");  <!-- Use when 2 or more output print in multi-line -->
   }
}
```

Variable :-

- A variable is like a container in your computer's memory where you store data.

◆ Examples :-

```
public class HelloWorld {
   public static void main(String[ ] args) {
      int age = 20 ; // local variable
      System.out.println("Age is: " + age) ;
   }
}
```

Data Type :-

- A data type is a way of telling the computer what kind of data a variable will store and how much memory it needs for that data.
- They are mainly divided into two categories :-

1. **Primitive Data Types :-**

- These are the most basic data types, directly stored in memory.

Type	Example	Description
byte	byte age = 25 ;	Stores small integers (-128 to 127).
short	short num = 32000 ;	Stores medium integers (-32,768 to 32,767).
int	int salary = 50000 ;	Stores large integers (-2,147,483,648 to 2,147,483,647).
long	long dist = 123456789L ;	Stores very large integers.
float	float pi = 3.14f ;	Stores decimal numbers (up to 7 decimal digits).
double	double price = 99.99 ;	Stores decimal numbers (up to 15 decimal digits).
char	char grade = 'A' ;	Stores a single character.
boolean	boolean flag = true ;	Stores true or false.

2. **Non-Primitive Data Types :-**

- Non-primitive data types are not built into the Java language by default like primitive types.

Type	Example	Description
String	String name = "Java";	Stores a sequence of characters (text).
Array	int[ ] arr = {1, 2, 3};	Stores multiple values of the same type.
Class	class Student { }	Blueprint for creating objects.
Interface	interface Drawable { }	Collection of abstract methods for classes to implement.
Object	Object obj = new Object();	Instance of a class; can store any type of object.

◆ **Examples :-**

```
public class DataTypeExample {
   public static void main(String[ ] args) {
      int age = 21;              // primitive int
      double price = 99.99;      // primitive double
      char grade = 'A';          // primitive char
      boolean isJavaFun = true;  // primitive boolean
      String name = "John";      // non-primitive String

      System.out.println("Age : " + age);   // Output :- Age : 21
      System.out.println("Price : " + price);  // Output :- Price : 99.99
      System.out.println("Grade : " + grade);  // Output :- Grade : A
      System.out.println("Is Java Fun? " + isJavaFun);  // Output :- Is Java Fun? true
      System.out.println("Name : " + name);  // Output :-  Name : John
   }
}
```

Naming Conversion :-

1. **Camel case :-**

- Camel case is a style of writing where the first word starts with a lowercase letter.
- Every subsequent word starts with a capital letter, with no spaces or underscores between words.
- This format is commonly used for variable names and method names in Java.
- **Example :-** myVariableName.

2. **Snake case :-**

- Snake case is a style of writing where all letters are lowercase. Words are separated by underscores (_) instead of spaces or capital letters.
- **Example :-** max_speed_limit.

3. **Kebab-case :-**

- Kebab case is a style of writing where all letters are lowercase. Words are separated by hyphens (-) instead of spaces or capital letters.
- This style is generally used in URLs, file names, or CSS class names, not in Java variable or method names.
- **Example :-** my-variable-name.

Keyword In Java :-

- keywords are reserved words that have a predefined meaning in the language syntax.
- You cannot use them as identifiers (like variable names, class names, method names) because they are part of Java's grammar rules.

List of Java Keywords :-

Category	Keywords
Data types	byte, short, int, long, float, double, char, boolean
Control statements	if, else, switch, case, default, while, do, for, break, continue, return
Access modifiers	public, private, protected
Non-access modifiers	static, final, abstract, synchronized, native, transient, volatile, strictfp
Class-related	class, interface, enum, extends, implements, new
Object-related	this, super, instanceof
Exception handling	try, catch, finally, throw, throws
Packages	package, import
Literals	true, false, null
Reserved (unused)	goto, const
Var handling	var, record, sealed, permits, non-sealed

Rules for Naming Identifiers :-

- It Can only contain Letters (A-Z, a-z), Digits (0-9), Underscore (_), Dollar sign ($).
- Java identifiers cannot be keywords, meaning reserved words like int, class, or for cannot be used as names.
- Identifier must start with a letter, an underscore (_), or a dollar sign ($), but cannot start with a digit.
- Java Identifier are Case-sensitive. Example :- total and Total are different identifiers.
- An identifier cannot contain spaces; for example, first name is invalid, but firstName is valid.
- Identifier can be of any length, but meaningful and readable names are recommended for better understanding.

User Input :-

- user input means taking data from the user during program execution.

Read user input using methods like :-

- nextInt( ) ⟶ This method reads an integer value entered by the user.
- nextFloat( ) ⟶ This method reads a floating-point number (decimal value) entered by the user.
- nextLine( ) ⟶ This method reads an entire line of text, including spaces, entered by the user.
- nextDouble( ) ⟶ This method reads a double-precision floating-point number entered by the user.
- nextBoolean( ) ⟶ This method reads a boolean value (true or false) entered by the user.

◆ **Example :-**

```
import java.util.Scanner;
public class MultiInput {
    public static void main(String[ ] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter roll number: ");
        int roll = sc.nextInt( );
        System.out.print("Enter marks: ");
        float marks = sc.nextFloat( );
        System.out.println("Roll : " + roll);
        System.out.println("Marks : " + marks);
    }
}
```

Challenges :-

1. Create a program to input name of the person and respond with "welcome NAME to Coding".

```
import java.util.Scanner;
public class WelcomeProgram {
public static void main(String[ ] args) {
Scanner sc = new Scanner(System.in);
System.out.print("Enter your name : ");
String name = sc.nextLine( ); // Reads complete name with spaces
System.out.println("Welcome " + name + " to Coding");
}
}
```

Output :-

```
Enter your name : Zeel Patel
Welcome Zeel Patel to Coding
```

---

2. Create a program to add two numbers.

```
import java.util.Scanner;
public class AddTwoNumbers {
public static void main(String[ ] args) {
Scanner sc = new Scanner(System.in);

System.out.print("Enter first number : ");
double num1 = sc.nextDouble( );
System.out.print("Enter second number : ");
double num2 = sc.nextDouble( );

double sum = num1 + num2;
System.out.println("The sum is : " + sum);
}
}
```

Output :-

```
Enter first number : 5
Enter second number : 7
The sum is : 12.0
```

Type Conversion :-

- Type conversion means changing one data type into another. In Java, this can happen automatically or manually.
- There are Two main types of type conversion :-

1. Widening Conversion (Automatic / Implicit) :-

- It Happens automatically when you assign a smaller data type to a bigger data type.
- Smallest to Largest :- [byte->short->int->long->float->double]

2. Narrowing Conversion (Manual / Explicit) :-

- It Happens when you assign a bigger data type to a smaller data type.
- Largest to smallest :- [byte<-short<-int<-long<-float<-double]

Example :-

```
public class TypeConversionCasting {
public static void main(String[ ] args) {
      // 1. Widening Conversion ( Automatic / Implicit ) :-
      int intNum = 100 ;          // int variable
      double doubleNum = intNum ;  // int ⟶ double
      System.out.println(" Implicit Casting ( int ⟶ double ) :- " + doubleNum) ;

      // 2. Narrowing Conversion ( Manual / Explicit ) :-
      double doubleValue = 9.79 ;
      int intValue = (int) doubleValue ; // double ⟶ int
      System.out.println(" Explicit Casting ( double ⟶ int ) :- " + intValue) ;
   }
}
```

Output :-

```
Widening Conversion ( int ⟶ double ) :- 100.0
Narrowing Conversion ( double ⟶ int ) :- 9
```

Types of Operators in Java :-

- They are used for calculations, comparisons, assignments, and logical operations.

1. Arithmetic Operators :-

- Arithmetic operators in Java are used to perform basic mathematical calculations on numeric values.

Operator	Meaning	Example
+	Addition	a + b
-	Subtraction	a - b
*	Multiplication	a * b
/	Division	a / b
%	Modulus (remainder)	a % b

Example :-

```
public class Arithmetic {
   public static void main(String[ ] args) {
   int a = 20, b = 6;
   // Addition
   int sum = a + b;
   int difference = a - b;
   int product = a * b;
   int quotient = a / b;
   int remainder = a % b;
   System.out.println("Addition ( a + b ) :- " + sum);
  System.out.println("Subtraction ( a – b ) :- " + difference);
  System.out.println("Multiplication ( a * b ) :- " + product);
  System.out.println("Division ( a / b ) :- " + quotient);
  System.out.println("Modulus ( a % b ) :- " + remainder);
 }  }
```

Output :-

Addition (a + b) :- 26

Subtraction (a ? b) :- 14

Multiplication (a * b) :- 120

Division (a / b) :- 3

Modulus (a % b) :- 2

2. Assignment (Shorthand) Operators :-

- Assignment operators are used to assign values to variables in Java.

Operator	Meaning	Example
=	Assign value	a = 10
+=	Add and assign	a += 5 (same as a = a + 5)
-=	Subtract and assign	a -= 3
*=	Multiply and assign	a *= 2
/=	Divide and assign	a /= 2
%=	Modulus and assign	a %= 3

Example :-

```
   public class AssignmentOperators {
   public static void main(String[ ] args) {
         int score = 50;
         System.out.println("Initial Score : " + score);
// Using += ( Add and assign )
         score += 10; // score = score + 10
         System.out.println("After += 10 : " + score);
```

```
// Using -= ( Subtract and assign )
        score -= 5; // score = score - 5
        System.out.println("After -= 5 :" + score);
// Using *= ( Multiply and assign )
        score *= 2; // score = score * 2
        System.out.println("After *= 2 :" + score);
// Using /= ( Divide and assign )
        score /= 4; // score = score / 4
        System.out.println("After /= 4 :" + score);
// Using %= ( Modulus and assign )
        score %= 3; // score = score % 3
        System.out.println("After %= 3 :" + score);
    }
  }
```

Output :- Initial Score :- 50

After += 10 :- 60

After -= 5 :- 55

After *= 2 :- 110

After /= 4 :- 27

After %= 3 :- 0

Challenges :-

1. Write a Program to Swap Two Numbers.

```
import java.util.Scanner ;
public class SwapNumbers {
public static void main(String[ ] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter first number (a) : ");
        int a = sc.nextInt( );
        System.out.print("Enter second number (b) : ");
        int b = sc.nextInt( );
        System.out.println("\nBefore Swap :-");
        System.out.println("a = " + a + ", b = " + b);
        int temp = a;
        a = b;
        b = temp;
        System.out.println("\nAfter Swap :-");
        System.out.println("a = " + a + ", b = " + b);
}  }
```

Output :-

Enter first number (a) :- 5

Enter second number (b) :- 10

Before Swap :- a = 5, b = 10

After Swap :- a = 10, b = 5

3. Unary Operators :-

- A Unary operators in Java are operators that work on only one operand to perform an operation.

Operator	Meaning	Example
+	Positive sign	+a
-	Negative sign	-a
++	Increment by 1	a++ or ++a
--	Decrement by 1	a-- or --a
!	Logical NOT	!true

Example :-

```
public class UnaryOperatorsExample {
public static void main(String[ ] args) {
    int a = 5;

    // 1. Unary Plus (+) and Minus (-)
    System.out.println("Unary plus (+a) : " + (+a)); // keeps the value same
    System.out.println("Unary minus (-a) : " + (-a)); // changes sign

    // 2. Increment Operators (++a, a++)
    System.out.println("Pre-increment (++a) : " + (++a)); // increases first, then prints
    System.out.println("Post-increment (a++) : " + (a++)); // prints first, then increases
    System.out.println("Value of a after post-increment : " + a);

    // 3. Decrement Operators (--a, a--)
    System.out.println("Pre-decrement (--a) : " + (--a)); // decreases first, then prints
    System.out.println("Post-decrement (a--) : " + (a--)); // prints first, then decreases
    System.out.println("Value of a after post-decrement : " + a);
  }
}
```

Output :-

```
Unary plus (+a) : 5
Unary minus (-a) : -5
Pre-increment (++a) : 6
Post-increment (a++) : 6
Value of a after post-increment : 7
Pre-decrement (--a) : 6
Post-decrement (a--) : 6
Value of a after post-decrement: 5
```

Challenges :-

1. Create a program to calculate products of two floating point numbers.

```
import java.util.Scanner;
public class ProductOfNumbers {
public static void main(String[ ] args) {
Scanner sc = new Scanner(System.in);
        System.out.print("Enter first number: ");
        float num1 = sc.nextFloat( );
        System.out.print("Enter second number: ");
        float num2 = sc.nextFloat( );
        float product = num1 * num2;
        System.out.println("Product : " + product);
    }
  }
```

Output :-

Enter first number : 2.5

Enter second number : 4.2

Product : 10.5

2. create a program to calculate the area of triangle. Area= 1/2*B*H.

```
 import java.util.Scanner;
public class AreaOfTriangle {
public static void main(String[ ] args) {
Scanner sc = new Scanner(System.in);
        System.out.print("Enter base of the triangle: ");
        float base = sc.nextFloat( );
        System.out.print("Enter height of the triangle: ");
        float height = sc.nextFloat( );
        float area = 0.5f * base * height;
        System.out.println("Area of the triangle : " + area);
    }
}
```

Output :-

Enter base of the triangle: 5

Enter height of the triangle: 10

Area of the triangle: 25.0

3. create a program to calculate simple interest. interest=(P x T x R)/100

```
import java.util.Scanner;
public class SimpleInterest {
public static void main(String[ ] args) {
Scanner sc = new Scanner(System.in);
        System.out.print("Enter Principal amount (P) : ");
        float principal = sc.nextFloat( );
        System.out.print("Enter Time in years (T) : ");
        float time = sc.nextFloat( );
        System.out.print("Enter Rate of Interest (R) : ");
        float rate = sc.nextFloat( );
        float interest = (principal * time * rate) / 100;
        System.out.println("Simple Interest : " + interest);
    }
}
```

Output :-

```
Enter Principal amount (P) : 10000
Enter Time in years (T) : 2
Enter Rate of Interest (R) : 5
Simple Interest : 1000.0
```

4. Create program to convert Fahrenheit to Celsius.

```
import java.util.Scanner;
public class FahrenheitToCelsius {
public static void main(String[ ] args) {
Scanner sc = new Scanner(System.in);
        System.out.print("Enter temperature in Fahrenheit : ");
        float fahrenheit = sc.nextFloat();
        float celsius = (fahrenheit - 32) * 5 / 9;
        System.out.println("Temperature in Celsius : " + celsius);
    }
}
```

Output :-

```
Enter temperature in Fahrenheit : 98.6
Temperature in Celsius : 37.0
```

IF – Else :-

-> if and else are decision-making statements in Java.

-> They let your program choose which code block to run based on a condition.

Syntax :-

```
if (condition) {
    // Code runs if condition is true
} else {
    // Code runs if condition is false
}
```

Example :-

```
import java.util.Scanner;
public class VotingEligibility {
    public static void main(String[ ] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter your age :- ");
        int age = sc.nextInt( );
        if (age >= 18) {
            System.out.println("You are eligible to vote.");
        } else {
            System.out.println("You are NOT eligible to vote.");
        }
    }
}
```

Output :-

```
Enter your age :- 19
You are eligible to vote.
```

---

4. Relational Operators :-

- Relational operators in Java are used to compare two values.
- They are mostly used in conditional statements such as if, while, for, and do-while loops.
- It Mostly used in conditions inside if, while, for, etc.

Operator	Meaning	Example
==	Equal to	a == b
!=	Not equal to	a != b
>	Greater than	a > b
<	Less than	a < b
>=	Greater than or equal to	a >= b
<=	Less than or equal to	a <= b

Example :-

```
import java.util.Scanner;
public class RelationalExample {
    public static void main(String[ ] args) {
        Scanner sc = new Scanner(System.in); // Create Scanner object
        System.out.print("Enter first number : ");
        int num1 = sc.nextInt( );
        System.out.print("Enter second number : ");
        int num2 = sc.nextInt();

        System.out.println(num1 + " == " + num2 + " : " + (num1 == num2));
        System.out.println(num1 + " != " + num2 + " : " + (num1 != num2));
        System.out.println(num1 + " > " + num2  + " : " + (num1 > num2));
        System.out.println(num1 + " < " + num2  + " : " + (num1 < num2));
        System.out.println(num1 + " >= " + num2 + " : " + (num1 >= num2));
        System.out.println(num1 + " <= " + num2 + " : " + (num1 <= num2));
    }
}
```

Output :-

```
Enter first number : 10
Enter second number : 20
10 == 20 : false
10 != 20 : true
10 > 20  : false
10 < 20  : true
10 >= 20 : false
10 <= 20 : true
```

5. Logical Operators :-

- Logical operators are used to combine two or more conditions.

Operator	Meaning	Example
&&	Logical AND	(a > b) && (a < c)
||	Logical OR	(a > b) || (a < c)
!	Logical NOT	!(a > b)

Example :-

```
import java.util.Scanner;
public class LogicalOperators {
    public static void main(String[ ] args) {
        Scanner sc = new Scanner(System.in);
```

```
System.out.print("Enter your age : ");
int age = sc.nextInt( );
System.out.print("Do you have a Voter ID? (true/false) : ");
boolean hasVoterID = sc.nextBoolean( );

// Logical AND (&&) ⟶ both conditions must be true
if (age >= 18 && hasVoterID) {
    System.out.println("You are eligible to vote.");
} else {
    System.out.println(“You are NOT eligible to vote.");
}

// Logical OR (||) ⟶ at least one condition must be true
    if (age >= 18 || hasVoterID) {
        System.out.println(" 🙂 You meet at least one voting requirement.");
    } else {
        System.out.println(" 😟 You meet no voting requirements.");
    }

    // Logical NOT (!) ⟶ reverses result
    boolean isAdult = age >= 18;
    System.out.println("Is adult? : " + isAdult);
    System.out.println("Not adult? : " + !isAdult);
  }
}
```

Output :-

```
Enter your age : 20
Do you have a Voter ID? (true/false) : true
You are eligible to vote.
🙂 You meet at least one voting requirement.
Is adult? : true
Not adult? : false
```

Challenges :-

1. Create a program that determines if a number is positive , negative, or zero.

```
import java.util.Scanner;
public class NumberCheck {
    public static void main(String[ ] args) {
        Scanner sc = new Scanner(System.in);
```

```
        System.out.print("Enter a number: ");
        double num = sc.nextDouble( );
        if (num > 0) {
            System.out.println("The number is Positive.");
        } else if (num < 0) {
            System.out.println("The number is Negative.");
        } else {
            System.out.println("The number is Zero.");
        }
    }
}
```

Output :-

Enter a number : 25

The number is Positive.

2. create a program that determines if a number is odd or even.

```
import java.util.Scanner;
public class OddEvenCheck {
    public static void main(String[ ] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number : ");
        int num = sc.nextInt( );
        if (num % 2 == 0) {
            System.out.println("The number is Even.");
        } else {
            System.out.println("The number is Odd.");
        }
    }
}
```

Output :-

Enter a number : 10

The number is Even.

3. create a program that determines if a given year is a leap year.

```
import java.util.Scanner;
public class LeapYearCheck {
    public static void main(String[ ] args) {
        Scanner sc = new Scanner(System.in);
```

```
        // Take year input
        System.out.print("Enter a year: ");
        int year = sc.nextInt( );
        // Leap year rules :
        // 1. Year divisible by 4 ⟶ possible leap year
        // 2. If divisible by 100 ⟶ must also be divisible by 400

        if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {
            System.out.println(year + " is a Leap Year.");
        } else {
            System.out.println(year + " is NOT a Leap Year.");
        }
    }
}
```

4. create a program that calculate grades based on marks. A--> above 90% B--> above 75% C --> above 60% D --> above 30% F --> Below 30%.

```
import java.util.Scanner;
public class GradeCalculator {
    public static void main(String[ ] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter your marks (0-100): ");
        int marks = sc.nextInt();
        if (marks > 90) {
            System.out.println("Grade : A");
        } else if (marks > 75) {
            System.out.println("Grade : B");
        } else if (marks > 60) {
            System.out.println("Grade : C");
        } else if (marks > 30) {
            System.out.println("Grade : D");
        } else {
            System.out.println("Grade : F");
        }
    }
}
```

Output :-

```
Enter your marks (0-100) : 95
Grade : A
```

5. Create a program that categorize a person into different age groups. Child -> below 13 Teen -> below 20 Adult -> below 60 Senior -> above 60.

```
import java.util.Scanner;
public class AgeGroupCategorizer {
    public static void main(String[ ] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter your age :- ");
        int age = sc.nextInt( );
        if (age < 13) {
            System.out.println("Category : Child");
        } else if (age < 20) {
            System.out.println("Category : Teen");
        } else if (age < 60) {
            System.out.println("Category : Adult");
        } else {
            System.out.println("Category : Senior");
        }
    }
}
```

Output :-

Enter your age : 20

Category : Adult

---

6. Bitwise Operators :-

- Bitwise operators in Java are used to perform operations directly on the binary (bit-level) representation of numbers.

Operator	Meaning	Example
&	Bitwise AND	a & b
|	Bitwise OR	a | b
^	Bitwise XOR	a ^ b
~	Bitwise NOT	~a
<<	Left shift	a << 2
>>	Right shift	a >> 2
>>>	Unsigned right shift	a >>> 2

1. Create a program that shows bitwise AND , OR, XOR, NOT, Left Shift, Right Shift.

```
public class BitwiseExample {
    public static void main(String[ ] args) {
        int a = 5; // Binary : 0101
        int b = 3; // Binary : 0011
```

```
        System.out.println("a & b  = " + (a & b)); // AND  // 0101 & 0011 = 0001 (1)
        System.out.println("a | b  = " + (a | b)); // OR   // 0101 | 0011 = 0111 (7)
        System.out.println("a ^ b  = " + (a ^ b)); // XOR  // 0101 ^ 0011 = 0110 (6)
        System.out.println("~a     = " + (~a));    // NOT  // ~0101 = 1010 (two's complement)
        System.out.println("a << 1 = " + (a << 1));// Left Shift  // 0101 << 1 = 1010 (10)
        System.out.println("a >> 1 = " + (a >> 1));// Right Shift // 0101 >> 1 = 0010 (2)
    }
}
```

Output :-

```
a & b  = 1     // 0101 & 0011 = 0001
a | b  = 7     // 0101 | 0011 = 0111
a ^ b  = 6     // 0101 ^ 0011 = 0110
~a     = -6    // ~0101 = 1010 (two's complement)
a << 1 = 10    // 0101 << 1 = 1010
a >> 1 = 2     // 0101 >> 1 = 0010
```

Example :-

4 & 1 ⟶ 100 & 001 ⟶ 000 ⟶ Even

7 & 1 ⟶ 111 & 001 ⟶ 001 ⟶ Odd

1. Write a program to check given number is even or odd using bitwise operator.

```
import java.util.Scanner;
public class EvenOddBitwise {
    public static void main(String[ ] args) {
        Scanner sc = new Scanner(System.in);
        // Taking input from the user
        System.out.print("Enter a number : ");
        int num = sc.nextInt( );
        // Using bitwise AND to check even or odd
        if ((num & 1) == 0) {
            System.out.println(num + " is Even.");
        } else {
            System.out.println(num + " is Odd.");
        }
    }
}
```

Output :-

```
Enter a number : 8
8 is Even.
```

Java Comments :-

- It is used to add notes in java code.

Syntax :-

1. Single Line :- //

2. Multi-Line :- /* */

3. Java Docs :- /***/

Example :-

```
public class CommentExample {
    public static void main(String[ ] args) {
        // This is a single-line comment
        int a = 10; // You can also write it beside code
        /* This is a
           multi-line comment
           explaining multiple steps */
        int b = 20;
        /**
         * This is a JavaDoc comment
         * It is used for generating official documentation
         */
        int sum = a + b;
        System.out.println("Sum :- " + sum);
    }
}
```

Loop in Java :-

- A loop is a programming structure that allows you to repeat a block of code multiple times until a certain condition is met.
- They make programs shorter and easier to maintain. Type of Loops are while, for , do-while.

While Loop :-

- It Used when the number of iterations is unknown and we check the condition before running the code.

Example :-

```
public class WhileExample {
    public static void main(String[ ] args) {
        int i = 1; // Initial value
        while (i <= 5) { // Condition
            System.out.println("Number : " + i);
            i++; // Increment
        }
    }
}
```

Output :-

Number : 1

Number : 2

Number : 3

Number : 4

Number : 5

do-While Loop :-

- The do-while loop is used to execute a block of code at least once, and then repeatedly execute the block as long as the specified condition is true.

Example :-

```
import java.util.Scanner;
public class DoWhileInputExample {
    public static void main(String[ ] args) {
        Scanner scanner = new Scanner(System.in);
        int number;
        do {
            System.out.print("Enter a number (0 to exit) : ");
            number = scanner.nextInt( );
            System.out.println("You entered : " + number);
        } while (number != 0);
        System.out.println("Loop ended because you entered 0.");
    }
}
```

Output :-

Enter a number (0 to exit) : 5

You entered : 5

Enter a number (0 to exit) : 3

You entered : 3

Enter a number (0 to exit) : 8

You entered : 8

Enter a number (0 to exit) : 0

You entered : 0

Loop ended because you entered 0.

For Loop :-

- The for loop is used to run a block of code a specific number of times.

Syntax :-

```
for (initialization; condition; update) {
    // Code to execute repeatedly
}
```

Example :- If start = 1, pivot = 5, and end = 9, the output should be :- 543216789.

```
public class PivotPrintWhileFunction {
   // Function to print from pivot down to start.
   public static void printDown(int start, int pivot) {
      int i = pivot;
      while (i >= start) {
         System.out.print(i);
         i--;
      }
   }
   // Function to print from pivot+1 up to end.
   public static void printUp(int pivot, int end) {
      int i = pivot + 1;
      while (i <= end) {
         System.out.print(i);
         i++;
      }
   }
   public static void main(String[ ] args) {
      int start = 1;
      int end = 9;
      int pivot = 5;
      printDown(start, pivot);
      printUp(pivot, end);
   }
}
```

Output :- 543216789.

Functions in Java :-

- A function is a block of code that performs a specific task.
- In Java, functions cannot exist outside of a class; they must be inside a class.
- A function that is declared as static can be called using the class name without creating an object.

Syntax :-

```
modifier static returnType functionName(parameter1Type parameter1Name, parameter2Type parameter2Name, ...)
{
   // Body of the function
   // Statements to perform the task
   return value; // if returnType is not void
}
```

modifier ⟶ Access level of the function (public, private, protected, or default).

static ⟶ Makes the function belong to the class (can be called without creating an object).

returnType ⟶ The type of value the function will return (int, double, String, or void if nothing is returned).

functionName ⟶ Name of the function (should follow Java naming rules).

parameters ⟶ Input values the function accepts (optional; can be zero or more).

body ⟶ The code block inside { } that performs the task.

Example 1 :-

```
import java.util.Scanner;
public class Demo {
    public static void checkNumber(int num) {
        if (num > 0) {
            System.out.println(num + " is Positive");
        } else if (num < 0) {
            System.out.println(num + " is Negative");
        } else {
            System.out.println("Number is Zero");
        }   }
    public static void main(String[ ] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number : ");
        int number = sc.nextInt( );
        checkNumber(number);
    }
}
```

Output :- Enter a number : 5

5 is Positive

Example 2 :-

```
import java.util.Scanner;
public class PatternProgram {
    public static void printFirstPattern( ) {
    int rows = 0 ;  // Start with row 0
    while (rows < 5) {  // Outer loop for number of rows
        System.out.print("*");  // Print first star for the row

        int i = 0 ;  // Inner loop counter
        while (i < rows) {  // Inner loop for extra stars
            System.out.print(" *"); // Space + star
            i++;
        }
```

```
        System.out.println( ); // Move to the next line
        rows++; // Increase row count
    }
}
    public static void main(String[ ] args) {
        printFirstPattern( );
    }
}
```

Output :-

```
*
**
***
****
*****
```

Return Statement :-

- The return statement is used to exit from a method and optionally send a value back to the method caller.

Example :-

```
public class Numbers{
public static void main(String[ ]args){
    int first = readNumber( );
    int second = readNumber( );
    int sum = first + second;
    System.out.println("Sum of the number is : " +sum);
}
public static int readNumber( ){
    Scanner input = new Scanner(System.in);
    System.out.println("Please enter number");
    int number = input.nextInt();
    return number;
}
}
```

Output :-

```
Please enter number : 5
Please enter number : 7
Sum of the number is : 12
```

Argument and Parameters :-

- A parameter is a variable that is declared in the method definition to receive input values.
- An argument is the actual value or expression provided to a method when it is called.

Example :-

```
public class Demo {
   // 'x' and 'y' are parameters
   public static int add(int x, int y) {
      return x + y;
   }
   public static void main(String[ ] args) {
      int result = add(5, 3);  // 5 and 3 are arguments
      System.out.println(result);  // Output : 8
   }
}
```

Challenges :-

1. Develop a program that print the multiplication table of given number.

```
import java.util.Scanner;
public class Multiplication {
   public static void printMultiplication(int num) {
      int i = 1;
      while(i <= 10){
         System.out.println(num + " X " + i + " = " + (num * i));
         i++;
      }
   }
   public static void main(String[ ] args) {
      Scanner input = new Scanner(System.in);
      System.out.println("Please enter number");
      int num = input.nextInt( );
      printMultiplication(num);
      input.close();
   }
}
```

Output :-

```
Please enter number : 5
5 X 1 = 5
5 X 2 = 10
5 X 3 = 15
5 X 4 = 20
5 X 5 = 25
5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45
5 X 10 = 50
```

2. Create a program to sum all odd numbers from 1 to a specified number N.

```
import java.util.Scanner;
public class Multiplication {
   public static int oddSum(int num) {
      int sum = 0;
```

Output :-

```
Please enter number : 100
Odd Sum till 100 is 2500
```

```
    int i = 1;
    while(i<=num){
        sum = sum + i;
        i += 2;
    }
    return sum;
}
public static void main(String[ ] args) {
    Scanner input = new Scanner(System.in);
    System.out.println("Please enter number : ");
    int num = input.nextInt();
    int sum = oddSum(num);
    System.out.println(" Odd Sum till " + num + " is " + sum);
  }
}
```

3. write a function that calculates the factorial of a given number.

```
import java.util.Scanner;
public class Factorial {
   public static long factorial(int num) {
       if(num < 2) {   return 1;   }
       long fact = 1;
       int i = 2;
       while (i<=num){
           fact *= i;
           i++;
       }
      return fact;
   }
   public static void main(String[ ] args) {
       Scanner input = new Scanner(System.in);
       System.out.println("Please enter number : ");
       int num = input.nextInt( );
       long fact = factorial(num);
       System.out.println("Factorial is : " +fact);
   }
}
```

Output :-

Please enter number : 4

Factorial is : 24

4. create a program that computes the sum of the digits of an integer.

```
import java.util.Scanner;
public class Sum {
   public static int sumDigit(int num) {
      int sum = 0;
      while (num>0){
         sum += num % 10;
         num /= 10;
      }
      return sum;
   }
   public static void main(String[ ] args) {
      Scanner input = new Scanner(System.in);
      System.out.println("Please enter number : ");
      int num = input.nextInt( );
      int sum = sumDigit(num);
      System.out.println("Sum of digits is : " +sum);

   }
}
```

Output :-

Please enter number : 123456

Sum of digit is : 21

5. create a program to find the least common multiple (LCM) of two numbers.

```
import java.util.Scanner;
public class LCM {
   public static int lcm(int first, int second) {
      int i = 1;
      while(true){
         int factor = first * i;
         if(factor % second == 0){
            return factor;
         }
         i++;
      }
}
   public static void main(String[ ] args) {
      Scanner input = new Scanner(System.in);
      System.out.println("Please enter number 1 : ");
      int first = input.nextInt( );
      System.out.println("Please enter number 2 : ");
      int second = input.nextInt( );
```

Output :-

Please enter number 1 : 2

Please enter number 2 : 4

lcm of the two number is : 4

```
        int lcm = lcm(first, second);
        System.out.println("lcm of the two number is : " +lcm);
    }
}
```

6. create a program to find the Greatest common divisor (GCD) of two numbers.

```
import java.util.Scanner;
public class GCD {
    public static int gcd(int num1, int num2) {
        int gcd = 1;
        int i = 2;
        int least = least(num1, num2);
        while (i <= least) {
            if (num1 % i == 0 && num2 % i == 0) {
                gcd = i;
            }
            i++;
        }
        return gcd;
    }
    public static int least(int num1, int num2) {
        if (num1 < num2) {
            return num1;
        } else {
            return num2;
        }
    }
    public static void main(String[ ] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter number 1: ");
        int num1 = input.nextInt();
        System.out.println("Please enter number 2: ");
        int num2 = input.nextInt();
        int gcdValue = gcd(num1, num2);
        System.out.println("GCD of " + num1 + " and " + num2 + " is: " + gcdValue);
    }
}
```

Output :-

```
Please enter number 1 :  15
Please enter number 2 :  20
GCD of 15 and 20 is : 5
```

7. Create a program to check whether given number is prime.

```
import java.util.Scanner;
public class Prime {
    public static boolean isPrime(int num) {
        int i = 2;
        while(i < num){
            if(num % i == 0){
                return false;
            }
            i++;
        }
        return true;
    }
    public static void main(String[ ] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter number 1: ");
        int num1 = input.nextInt();
        boolean prime = isPrime(num1);
        if(prime){
            System.out.println("Your number is prime");
        }
        else{
            System.out.println("Your number is not prime");
        }
    }
}
```

Output :-

```
Please enter number 1 : 23
Your number is prime
```

8. create a program to reverse the digits of a number.

```
import java.util.Scanner;
public class ReverseTheDigits {
    public static int reverse(int num) {
        int newNum = 0;
        while(num > 0){
            int digit = num % 10;
            newNum = newNum * 10 + digit;
            num /= 10;
        }
        return newNum;
    }
```

Output :-

```
Please enter number : 249
Reverse of number is : 942
```

```
    public static void main(String[ ] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter number : ");
        int num = input.nextInt();
        int reverse = reverse(num);
        System.out.println("Reverse of number is : " +reverse);
}
}
```

9. create a program to print the Fibonacci series up to a certain numbers.

```
import java.util.Scanner;
public class Fibonacci {
    public static void printFibonacci(int num) {
        if (num < 0) return;
        System.out.print("0 ");
        if (num == 0) return;
        System.out.print("1 ");

        int first = 0, second = 1;
        while (first + second <= num) {
            int third = first + second;
            System.out.print(third + " ");
            first = second;
            second = third;
        }
}
public static void main(String[] args) {
    Scanner input = new Scanner(System.in);
        System.out.println("Please enter number: ");
        int num = input.nextInt();
        System.out.println("Here is the Fibonacci series:");
        printFibonacci(num);
    }
}
```

Output :-

Please enter number : 200

Here is the Fibonacci series : 0 1 1 2 3 5 8 13 21 34 55 89 144

10. create a program to check if number is an Armstrong number. Example :- 153 is an Armstrong number because 1^3+5^3+3^3 = 153

```
import java.util.Scanner;
public class ArmstrongChecker {

    public static boolean isArmstrong(int num) {
        int noOfDigits = noDigits(num);
        System.out.println("Number of digits : "+noOfDigits);
        int originalNum = num;
        int finalNumber = 0;

        while (num > 0) {
            int digit = num % 10;
            num /= 10;
            finalNumber += power(digit, noOfDigits);
        }
        System.out.println("Final Number is : "+finalNumber);
        return finalNumber == originalNum;
    }

    public static int power(int base, int exp) {
    int result = 1;
    int i = 0;
    while (i < exp) {
        result *= base;
        i++;
    }
    System.out.println("Power of " + base + " is " +result);
    return result;
    }

    public static int noDigits(int num) {
        int digits = 0;
        while (num > 0) {
            digits++;
            num /= 10;
        }
        return digits;
    }
```

Output :-

```
Enter a number : 1
1 is an Armstrong number.
```

```
    public static void main(String[ ] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter a number : ");
        int number = input.nextInt();
        if (isArmstrong(number)) {
            System.out.println(number + " is an Armstrong number.");
        } else {
            System.out.println(number + " is NOT an Armstrong number.");
        }
    }
}
```

11. create a program to verify if a number is palindrome.

```
import java.util.Scanner;
public class Palindrome {
    public static boolean palindrome(int num) {
        return num == reverse(num);
    }

    public static int reverse(int num) {
        int newNum = 0;
        while(num > 0){
            int digit = num % 10;
            newNum = newNum * 10 + digit;
            num /= 10;
        }
        return newNum;
    }
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int number = input.nextInt();
        boolean palindrome = palindrome(number);
        if(palindrome){
            System.out.println("Your number is palindrome number");
        }
        else{
            System.out.println("Your number is not a palindrome number");
        }
} }
```

Output :-

Enter a number : 32123

Your number is palindrome number

12. Create a program that print patterns.

Right Half Pyramid pattern :-

```
*
* *
* * *
* * * *
* * * * *
```

Reverse Right Half Pyramid pattern :-

```
* * * * *
* * * *
* * *
* *
*
```

Left Half Pyramid pattern :-

```
    *
   * *
  * * *
 * * * *
* * * * *
```

Program :-

```
import java.util.Scanner;
public class RightHalfPyramid {

    public static void printRightHalfPyramid(int maxRows) {
        System.out.println("Right Half Pyramid :-");
        int rows = 0;
        while (rows < maxRows) {
            int i = 0;
            while (i < rows) {
                System.out.print("*  ");
                i++;
            }
            System.out.println( );
            rows++;
        }
    }
    public static void printReverseRightHalfPyramid(int maxRows) {
        System.out.println("Reverse Right Half Pyramid :-");
        int rows = maxRows;
```

```
    while (rows > 0) {
        int i = 0;
        while (i < rows) {
            System.out.print(" *");
            i++;
        }
        System.out.println( );
        rows--;
    }
}
public static void printLeftHalfPyramid(int maxRows) {
    System.out.println("Left Half Pyramid :-");
    int rows = 1;
while (rows <= maxRows) {
    // Print spaces
    int spaces = maxRows - rows;
    int j = 0;
    while (j < spaces) {
        System.out.print("  ");  // two spaces for alignment
        j++;
    }
    // Print stars
    int i = 0;
    while (i < rows) {
        System.out.print("* ");
        i++;
    }
    System.out.println( );
        rows++;
    }
}
public static void main(String[ ] args) {
    Scanner input = new Scanner(System.in);
    System.out.print("Enter a number of Rows : ");
    int rows = input.nextInt( );
    printRightHalfPyramid(rows);
    printReverseRightHalfPyramid(rows);
    printLeftHalfPyramid(rows);
} }
```

Array In Java :-

- An Array is a collection or container that stores multiple values of the same data type in a single variable.
- Every element has a specific position called an index, which starts from 0.
- Arrays are useful when you want to store lists of data, like scores, names, or any repeated items.
- The size of an array is fixed when it is created and cannot be changed later.

Declaration and Creation :-

Syntax :- dataType[ ] arrayName = new dataType[size];

Example :- int[ ] numbers = new int[5]; // Array to hold 5 integers

- dataType is the type of elements the array will hold (e.g., int, String).
- arrayName is the name you give to the array.
- size is the number of elements the array will hold.

Initializing an Array with Values (Directly) :-

Syntax :- dataType[ ] arrayName = {value1, value2, value3, ...};

Example :- int[ ] numbers = {10, 20, 30, 40, 50};

Accessing Array Elements :-

Syntax :- arrayName[index]

Example :- int firstNumber = numbers[0]; // Access first element

Changing Array Elements :-

Syntax :- arrayName[index] = newValue;

Example :- numbers[2] = 100; // Change the third element to 100

Example :-

```
public class ArrayExample {
    public static void main(String[ ] args) {
        // 1. Declare and create an array of integers with size 5 :-
        int[ ] numbers = new int[5];
        // 2. Initialize elements of the array :-
        numbers[0] = 10;
        numbers[1] = 20;
        numbers[2] = 30;
        numbers[3] = 40;
        numbers[4] = 50;
        // 3. Access and print elements using their indexes :-
        System.out.println("First element :- " + numbers[0]);  // 10
        System.out.println("Third element :- " + numbers[2]);  // 30

        // 4. Update an element in the array :-
        numbers[2] = 100;
        System.out.println("Updated third element :- " + numbers[2]);  // 100
        // 5. You can also declare and initialize an array in one step :-
```

```
String[ ] fruits = {"Apple", "Banana", "Cherry"};
System.out.println("First element :- " + fruits[0]);

// 6. Access value of Array using Loop :-
int index = 0;
while(index<5){
    System.out.println("Array of " + index + " index : " +numbers[index]);
    index++;

// 7. Array Traversal in Loop :-
int index = 0;
while(index < numbers.length){
    System.out.println("Array of " + index + " index : " +numbers[index]);
    index++;
        }
      }
   }
}
```

Output :-

```
First element : 10
Third element : 30
Updated third element : 100
First element : Apple
Array of 0 index : 10
Array of 1 index : 20
Array of 2 index : 100
Array of 3 index : 40
Array of 4 index : 50
```

---

1. Write A program of Array Searching.

```
import java.util.Scanner;
public class ArraySearching {
   public static void main(String[ ] args) {
      Scanner input = new Scanner(System.in);
      int[ ] arr = {3, 6, 8, 87, 65, 4, 68, 23, 9, 98, 34};
      System.out.println("Enter the number you want to search:");
      int num = input.nextInt( );

      boolean isFound = isFound(arr, num);
```

Output :-

```
Enter the number you want to search : 98
Your number was found in the array.
```

```
    if (isFound) {
        System.out.println("Your number was found in the array.");
    } else {
        System.out.println("Your number was not found in the array.");
    }
}

// Method to search for number in array using while loop
public static boolean isFound(int[ ] arr, int num) {
    int index = 0;
    while (index < arr.length) {
        if (arr[index] == num) {
            return true;  // Found the number, return true
        }
        index++;
    }
    return false;  // Number not found after checking all elements
  }
}
```

2D Array :-

- A 2D array is like a table or matrix with rows and columns.

Declaration and Creation :-

Syntax :- arrayName = new dataType[rows][columns];

Example :- int[ ][ ] matrix = new int[3][4]; // 3 rows and 4 columns

Initializing a 2D Array with Values :-

```
int[ ][ ] matrix = {
    {1, 2, 3, 4},
    {5, 6, 7, 8},
    {9, 10, 11, 12}
};
```

Example 1 :- Declaration and Creation Array.

```
public class TwoDArrayWhile {
    public static void main(String[ ] args) {
        // Declare and create 2D array
        int[ ][ ] matrix = new int[3][4];
        // Assign some values
        matrix[0][0] = 10;
```

Output :-

```
10  20  0   0
30  0   0   0
0   0   0   40
```

```
matrix[0][1] = 20;
matrix[1][0] = 30;
matrix[2][3] = 40;

// Print using while loops
int i = 0;
while (i < matrix.length) {
    int j = 0;
    while (j < matrix[i].length) {
        System.out.print(matrix[i][j] + " ");
        j++;
    }
    System.out.println( );
    i++;
  }
 }
}
```

Example : 2 :- Initializing, and Printing a 2D Array.

```
public class TwoDArrayExample {
    public static void main(String[ ] args) {
        // Declare and initialize 2D array
        int[ ][ ] matrix = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };
        int i = 0;  // row index
        while (i < matrix.length) {
            int j = 0;  // column index
            while (j < matrix[i].length) {
                System.out.print(matrix[i][j] + " ");
                j++;
            }
            System.out.println( );  // move to next line after each row
            i++;
        }
    }
}
```

Output :-

```
1 2 3
4 5 6
7 8 9
```

Challenges :-

1. Create a program to find the sum and average of all elements in an array.

```
import java.util.Scanner;
public class array {
    public static void main(String[ ] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter number of elements: ");
        int size = input.nextInt( );
        int[ ] nums = new int[size];

        // Taking array input using while loop
        int i = 0;
        while (i < size) {
            System.out.print("Please enter element no " + (i + 1) + ": ");
            nums[i] = input.nextInt();
            i++;
        }

        // Now calculate sum and average
        long sum = sum(nums);
        int avg = average(nums);
        System.out.println("Sum of elements: " + sum);
        System.out.println("Average of elements: " + avg);
    }

    public static long sum(int[ ] nums) {
        long sum = 0;
        int i = 0;
        while (i < nums.length) {
            sum += nums[i];
            i++;
        }
        return sum;
    }
    public static int average(int[ ] nums) {
        long sum = sum(nums);
        return (int) (sum / nums.length);
    }
}
```

Output :-

```
Please enter number of elements : 5
Please enter element no 1 : 1
Please enter element no 2 : 4
Please enter element no 3 : 7
Please enter element no 4 : 2
Please enter element no 5 : 9
Sum of elements : 23
Average of elements : 4
```

2. create a program to find number of occurrences of an element in an array.

```
import java.util.Scanner;
public class array {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter number of elements : ");
        int size = input.nextInt( );
        int[ ] numArr = new int[size];

        int i = 0;
        while (i < size) {
            System.out.print("Please enter element no " + (i + 1) + ": ");
            numArr[i] = input.nextInt( );
            i++;
        }
        System.out.println("Now enter number you want to find : ");
        int num = input.nextInt( );
        int occurrences = noOfOccurrences(numArr, num);
        System.out.println("Your element was found " + occurrences + " times in the array.");
}

public static int noOfOccurrences(int[] numArr, int num) {
    int occ = 0;
    int i = 0;
    while (i < numArr.length) {
        if (numArr[i] == num) {
            occ++;
        }
        i++;
    }
    return occ;
  }
}
```

Output :-

```
Please enter number of elements : 6
Please enter element no 1 : 1
Please enter element no 2 : 2
Please enter element no 3 : 3
Please enter element no 4 : 2
Please enter element no 5 : 4
Please enter element no 6 : 2
Now enter number you want to find : 2
Your element was found 3 times in the array.
```

3. create a program to find the maximum and minimum element in an array.

```
import java.util.Scanner;
public class array {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
```

```
System.out.println("Please enter number of elements: ");
int size = input.nextInt();
int[ ] numArr = new int[size];

int i = 0;
while (i < size) {
    System.out.print("Please enter element no " + (i + 1) + ": ");
    numArr[i] = input.nextInt();
    i++;
}
int max = max(numArr);
int min = min(numArr);
System.out.println("Max of the array : " + max);
    System.out.println("Min of the array : " + min);
  }
public static int max(int[ ] numArr) {
    if (numArr.length == 0) {
        return Integer.MIN_VALUE;
    }
    int max = numArr[0];
    int i = 1;
    while (i < numArr.length) {
        if (numArr[i] > max) {
            max = numArr[i];
        }
        i++;
    }
    return max;
 }
public static int min(int[ ] numArr) {
    if (numArr.length == 0) {
        return Integer.MAX_VALUE;
    }
    int min = numArr[0];
    int i = 1;
    while (i < numArr.length) {
        if (numArr[i] < min) {
            min = numArr[i];
        }  i++;
```

Output :-

```
Please enter number of elements :  5
Please enter element no 1 : 4
Please enter element no 2 : 8
Please enter element no 3 : 2
Please enter element no 4 : 98
Please enter element no 5 : -45
Max of the array : 98
Min of the array : -45
```

```
        }
        return min;
    }
}
```

4. create a program to check if the given array is sorted.

```
import java.util.Scanner;
public class array {
    public static void main(String[ ] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter number of elements: ");
        int size = input.nextInt( );
        int[ ] numArr = new int[size];

        int i = 0;
        while (i < size) {
            System.out.print("Please enter element no " + (i + 1) + ": ");
            numArr[i] = input.nextInt();
            i++;
        }
        boolean isInc = isIncreasing(numArr);
        boolean isDec = isDecreasing(numArr);
        if (isInc || isDec) {
            System.out.println("Your array is sorted");
        } else {
            System.out.println("Your array is not sorted");
        }
    }

    public static boolean isIncreasing(int[ ] numArr) {
        int i = 1;
        while (i < numArr.length) {
            if (numArr[i] < numArr[i - 1]) {
                return false;
            }
            i++;
        }
        return true;
    }
```

Output :-

```
Please enter number of elements : 6
Please enter element no 1 : 53
Please enter element no 2 : 47
Please enter element no 3 : 23
Please enter element no 4 : 5
Please enter element no 5 : 2
Please enter element no 6 : -8
Your array is sorted
```

```
    public static boolean isDecreasing(int[ ] numArr) {
        int i = 1;
        while (i < numArr.length) {
            if (numArr[i] > numArr[i - 1]) {
                return false;
            }
            i++;
        }
        return true;
    }
}
```

5. create a program to return a new array deleting specific element.

```
import java.util.Scanner;
public class array {
    public static void main(String[ ] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter number of elements: ");
        int size = input.nextInt();
        int[] numArr = new int[size];

        int i = 0;
        while (i < size) {
            System.out.print("Please enter element no " + (i + 1) + ": ");
            numArr[i] = input.nextInt();
            i++;
        }

    System.out.println("Now enter the number you want to delete: ");
    int numToDelete = input.nextInt( );
    int[ ] newArr = deleteNumber(numArr, numToDelete);
    System.out.println("Here is your new array :");
    i = 0; // reuse same variable
    while (i < newArr.length) {
        System.out.print(newArr[i] + " ");
        i++;
    }
}
```

```
public static int[ ] deleteNumber(int[] numArr, int numToDelete) {
    int occ = noOfOccurrences(numArr, numToDelete);
    if (occ == 0) {
        System.out.println("Number not found in array.");
        return numArr;
    }
    int newSize = numArr.length - occ;
    int[ ] newArr = new int[newSize];
    int i = 0, j = 0;
    while (i < numArr.length) {
        if (numArr[i] != numToDelete) {
            newArr[j] = numArr[i];
            j++;
        }
        i++;
    }
    return newArr;
}

public static int noOfOccurrences(int[ ] numArr, int num) {
    int occ = 0;
    int i = 0;
    while (i < numArr.length) {
        if (numArr[i] == num) {
            occ++;
        }
        i++;
    }
        return occ;
    }
}
```

Output :- Please enter number of elements : 5

Please enter element no 1 : 1

Please enter element no 2 : 2

Please enter element no 3 : 3

Please enter element no 4 : 4

Please enter element no 5 : 5

Now enter the number you want to delete : 2

Here is your new array : 1 3 4 5

6. create a program to reverse array.

```
import java.util.Scanner;
public class array {
    public static void main(String[ ] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter number of elements: ");
        int size = input.nextInt( );
        int[ ] numArr = new int[size];

        int i = 0;
        while (i < size) {
            System.out.print("Please enter element no " + (i + 1) + ": ");
            numArr[i] = input.nextInt();
            i++;
        }
        reverse(numArr); // reverse in place

        System.out.println("Here is your reverse array :");
        i = 0;
        while (i < numArr.length) {
            System.out.print(numArr[i] + " ");
            i++;
        }
    }

    public static void reverse(int[ ] arr) {
        int i = 0;
        while (i < arr.length / 2) {
            int swap = arr[i];
            arr[i] = arr[arr.length - 1 - i];
            arr[arr.length - 1 - i] = swap;
            i++;
        }
    }
}
```

Output :-

```
Please enter number of elements : 5
Please enter element no 1 : 8
Please enter element no 2 : 5
Please enter element no 3 : 86
Please enter element no 4 : 1
Please enter element no 5 : 46
Here is your reverse array :46 1 86 5 8
```

7. create a program to check is the array is palindrome or not.

```
import java.util.Scanner;
public class PalindromeArray {
    public static void main(String[ ] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter number of elements: ");
        int size = input.nextInt();
        int[] arr = new int[size];

        int i = 0;
        while (i < size) {
            System.out.print("Please enter element no " + (i + 1) + ": ");
            arr[i] = input.nextInt();
            i++;
        }

        if (isPalindrome(arr)) {
            System.out.println("The array is a palindrome.");
        } else {
            System.out.println("The array is not a palindrome.");
        }
    }

    public static boolean isPalindrome(int[ ] arr) {
        int start = 0;
        while (start < arr.length / 2) {
            if (arr[start] != arr[arr.length-1-start]) {
                return false;
            }
            start++;
        }
        return true;
    }
}
```

Output :-

```
Please enter number of elements : 5
Please enter element no 1 : 1
Please enter element no 2 : 12
Please enter element no 3 : 3
Please enter element no 4 : 12
Please enter element no 5 : 1
The array is a palindrome.
```

8. create a program to merge two sorted array.

ArrayUtility.java :-

```
import java.util.Scanner;
public class ArrayUtility {
    public static int[ ] inputArray( ) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter number of elements : ");
        int size = input.nextInt();
        int[ ] num = new int[size];
        int i = 0;
        while (i < size) {
            System.out.print("Please enter element no " + (i + 1) + ": ");
            num[i] = input.nextInt();
            i++;
        }
        return num;
    }

    public static void displayArray(int[ ] numArray) {
        int i = 0;
        while (i < numArray.length) {
            System.out.print(numArray[i] + " ");
            i++;
        }
        System.out.println();
    }
}
```

```
import java.util.Scanner;
public class array {
    public static void main(String[ ] args) {
        System.out.println("Merging sorted arrays");
        int[ ] arr1 = ArrayUtility.inputArray( );
        int[ ] arr2 = ArrayUtility.inputArray( );
        int[ ] mergedArr = merge(arr1, arr2); // fixed method call
        System.out.println("Your merged array is:");
        ArrayUtility.displayArray(mergedArr);
    }
```

```
public static int[] merge(int[] arr1, int[] arr2) { // fixed name
    int newsize = arr1.length + arr2.length;
    int[ ] newArr = new int[newsize];
    int i = 0, j = 0, k = 0;
    while (i < arr1.length && j < arr2.length) {
        if (arr1[i] < arr2[j]) {
            newArr[k++] = arr1[i++];
        } else {
            newArr[k++] = arr2[j++];
        }
    }
    while (i < arr1.length) {
        newArr[k++] = arr1[i++];
        }
        while (j < arr2.length) {
            newArr[k++] = arr2[j++];
        }
        return newArr;
    }
}
```

Output :-

```
Merging sorted arrays
Please enter number of elements : 5
Please enter element no 1 : 2
Please enter element no 2 : 1
Please enter element no 3 : 3
Please enter element no 4 : 4
Please enter element no 5 : 5


Please enter number of elements : 5
Please enter element no 1 : 2
Please enter element no 2 : 7
Please enter element no 3 : 8
Please enter element no 4 : 9
Please enter element no 5 : 10
Your merged array is : 1 2 2 3 4 5 7 8 9 10
```

9. create a program to search an element in 2D-array.

ArrayUtility.java :-

```
import java.util.Scanner;
public class ArrayUtility {
    public static int[ ][ ] input2DArray( ) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please enter number of rows: ");
        int rows = input.nextInt( );
        System.out.println("Please enter number of columns: ");
        int columns = input.nextInt( );
        int[ ][ ] numArray = new int[rows][columns];

    int i = 0;
    while (i < rows) {
        int j = 0;
        while (j < columns) {
            System.out.print("Please enter element row " + (i + 1) + ", column " + (j + 1) + ": ");
            numArray[i][j] = input.nextInt();
            j++;
        }
        i++;
    }
    return numArray;
}
public static void display2DArray(int[ ][ ] numArray) {
    int i = 0;
    while (i < numArray.length) {
        int j = 0;
        while (j < numArray[i].length) {
            System.out.print(numArray[i][j] + " ");
            j++;
        }
        System.out.println( );
        i++;
    }
  }
}
```

```
import java.util.Scanner;
public class array {
    public static void main(String[ ] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Welcome to 2D Search\n");

        // Take 2D array input
        int[ ][ ] numArr = ArrayUtility.input2DArray( );

        System.out.print("Now enter the element you want to search: ");
        int num = input.nextInt( );

        boolean isFound = search(numArr, num);
        if (isFound) {
            System.out.println("Your number is found!");
        } else {
            System.out.println("Your number is not found.");
        }
}

public static boolean search(int[ ][ ] numArr, int num) {
    int i = 0;
    while (i < numArr.length) {
        int j = 0;
        while (j < numArr[i].length) {
            if (numArr[i][j] == num) {
                return true; // found
            }
            j++;
        }
        i++;
    }
    return false;
  }
}
```

Output :-

```
Please enter number of rows : 3
Please enter number of columns : 3
Enter element at row 1, column 1 : 1
Enter element at row 1, column 2 : 2
Enter element at row 1, column 3 : 3
Enter element at row 2, column 1 : 4
Enter element at row 2, column 2 : 5
Enter element at row 2, column 3 : 6
Enter element at row 3, column 1 : 7
Enter element at row 3, column 2 : 8
Enter element at row 3, column 3 : 9
Now enter the element you want to search : 6
Your number is found!
```

10. create a program to sum and average of all element in a 2D-array.

```
public class array {
    public static void main(String[ ] args) {
        System.out.println("Welcome to sum and Average of 2D Array\n");
        int[ ][ ] numArr = ArrayUtility.input2DArray( );
        long sum = sum(numArr);
        double avg = average(numArr);
        System.out.println("Your sum of array is : " + sum);
        System.out.println("Your average of array is : " + avg);
    }

    // Method to calculate average
    public static double average(int[ ][ ] numArr) {
        if (numArr.length == 0) {
            return 0;
        }
        int rows = numArr.length;
        int cols = numArr[0].length;
        double size = rows * cols;
        return (double) sum(numArr) / size; // casting for accurate average
    }

    // Method to calculate sum
    public static long sum(int[ ][ ] numArr) {
        long sum = 0;
        int i = 0;
        while (i < numArr.length) {
            int j = 0;
            while (j < numArr[i].length) {
                sum += numArr[i][j];
                j++;
            }
            i++;
        }
        return sum;
    }
}
```

Output :-

```
Welcome to sum and Average of 2D Array
Please enter number of rows : 2
Please enter number of columns : 2
Enter element at row 1, column 1 : 4
Enter element at row 1, column 2 : 7
Enter element at row 2, column 1 : 9
Enter element at row 2, column 2 : 2
Your sum of array is : 22
Your average of array is : 5.5
```