

## Java Input

Java provides different ways to get input from the user.

To get input from user using the object of Scanner class.

In order to use the object of Scanner, we need to import java.util.Scanner package.

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

Then, we need to create an object of the Scanner class. We can use the object to take input from the user.

```
// create an object of Scanner
```

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

```
// take input from the user
```

```
int number = input.nextInt();
```

## Example: Get Integer Input From the User

```
import java.util.Scanner;

class Input {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter an integer: ");
        int number = input.nextInt();
        System.out.println("You entered " + number);
        // closing the scanner object
        input.close();
    }
}
```

Output:

Enter an integer: 2

You entered 2

In the above example, we have created an object named input of the Scanner class. We then call the nextInt() method of the Scanner class to get an integer input from the user.

Similarly, we can use nextLong(), nextFloat(), nextDouble(), and next() methods to get long, float, double, and string input respectively from the user.

**Note:** We have used the close() method to close the object. It is recommended to close the scanner object once the input is taken.

# Java Expressions, Statements and Blocks

## Java Expressions

A Java expression consists of variables, operators, literals, and method calls.

```
int score;  
score = 90;
```

Here, `score = 90` is an expression that returns an `int`. Consider another example,

```
Double a = 2.2, b = 3.4, result;  
result = a + b - 3.4;
```

Here, `a + b - 3.4` is an expression.

```
if (number1 == number2)
```

```
    System.out.println("Number 1 is larger than number 2");
```

Here, `number1 == number2` is an expression that returns a boolean value. Similarly, `"Number 1 is larger than number 2"` is a string expression.

## Java Statements

In Java, each statement is a complete unit of execution. For example,

```
int score = 9*5;
```

Here, we have a statement. The complete execution of this statement involves multiplying integers 9 and 5 and then assigning the result to the variable score.

In the above statement, we have an expression  $9 * 5$ . In Java, expressions are part of statements.

## Expression statements

We can convert an expression into a statement by terminating the expression with a ;. These are known as expression statements. For example,

```
// expression  
number = 10  
// statement  
number = 10;
```

In the above example, we have an expression `number = 10`. Here, by adding a semicolon (;), we have converted the expression into a statement (`number = 10;`).

Consider another example,

```
// expression  
++number  
// statement  
++number;
```

## Declaration Statements

In Java, declaration statements are used for declaring variables.

For example,

```
Double tax = 9.5;
```

The statement above declares a variable `tax` which is initialized to 9.5.

## Java Blocks

A block is a group of statements (zero or more) that is enclosed in curly braces { }. For example,

```
class Main {  
    public static void main(String[] args) {  
  
        String band = "Beatles";  
  
        if (band == "Beatles") { // start of block  
            System.out.print("Hello ");  
            System.out.print("Bye");  
        } // end of block  
    }  
}
```

### Output:

Hello Bye

In the above example, we have a block if {....}.

Here, inside the block we have two statements:

However, a block may not have any statements.

## Java Comments

In computer programming, comments are a portion of the program that are completely ignored by Java compilers. They are mainly used to help programmers to understand the code. For example,

```
// declare and initialize two variables
```

```
int a =1;
```

```
int b = 3;
```

```
// print the output
```

```
System.out.println("This is output");
```

Here, we have used the following comments,

- declare and initialize two variables
- print the output



## Types of Comments in Java

In Java, there are two types of comments:

1. single-line comment
2. multi-line comment

## Single-line Comment

A single-line comment starts and ends in the same line. To write a single-line comment, we can use the `//` symbol. For example,

```
// "Hello, World!" program example
class Main {
    public static void main(String[] args) {
    {
        // prints "Hello, World!"
        System.out.println("Hello, World!");
    }
}
```

Here, we have used two single-line comments:

"Hello, World!" program example  
prints "Hello World!"

The Java compiler ignores everything from `//` to the end of line. Hence, it is also known as **End of Line comment**.

## Multi-line Comment

When we want to write comments in multiple lines, we can use the multi-line comment. To write multi-line comments, we can use the `/*....*/` symbol. For example,

```
/* This is an example of multi-line comment.  
 * The program prints "Hello, World!" to the standard output.  
 */
```

```
class HelloWorld {  
    public static void main(String[] args) {  
    {  
        System.out.println("Hello, World!");  
    }  
}
```

1. Java Program to Print an Integer (Entered by the User)
2. Java Program to Add Two Integers
3. Java Program to Multiply two Floating Point Numbers
4. Java Program to Find ASCII Value of a character
5. Java Program to Compute Quotient and Remainder
6. Java Program to Swap Two Numbers
7. Java Program to Check Whether a Number is Even or Odd
8. Java Program to Check Whether an Alphabet is Vowel or Consonant
9. Java Program to Find the Largest Among Three Numbers
10. Java Program to Find all Roots of a Quadratic Equation
11. Java Program to Find the Frequency of Character in a String
12. Java Program to Remove All Whitespaces from a String
13. Java Program to Round a Number to n Decimal Places
14. Java Program to Check if a String is Empty or Null



# Java Flow Control

## Java if...else Statement

There are various forms of if...else statements in Java.

if statement

if...else statement

if...else if...else statement

Nested if...else statement

# 1. Java if (if-then) Statement

The syntax of a if-then statement:

```
if (condition) {  
    // statements  
}
```

Here, condition is a boolean expression. It returns either true or false.

if condition evaluates to true, statements inside the body of if are executed  
if condition evaluates to false, statements inside the body of if are skipped

## Condition is true

```
int number = 10;
```

```
if (number > 0) {  
    // code  
}
```

```
// code after if
```

## Condition is false

```
int number = 10;
```

```
if (number < 0) {  
    // code  
}
```

```
// code after if
```



```
class IfStatement {  
    public static void main(String[] args) {  
  
        int number = 10;  
  
        // checks if number is greater than 0  
        if (number > 0) {  
            System.out.println("The number is positive.");  
        }  
  
        System.out.println("Statement outside if block");  
    }  
}
```

```
class Main {  
    public static void main(String[] args) {  
        // create a string variable  
        String language = "Java";  
  
        // if statement  
        if (language == "Java") {  
            System.out.println("Best Programming Language");  
        }  
    }  
}
```

## 2. Java if...else (if-then-else) Statement

The syntax of the if...else statement is:

```
if (condition) {  
    // codes in if block  
}  
else {  
    // codes in else block  
}
```

Here, the program will do one task (codes inside if block) if the condition is true and another task (codes inside else block) if the condition is false.

## Condition is true

```
int number = 5;
```

```
if (number > 0) {  
    // code
```

```
}  
else {  
    // code
```

```
}  
// code after if...else
```

## Condition is false

```
int number = 5;
```

```
if (number < 0) {  
    // code
```

```
}  
else {  
    // code
```

```
}  
// code after if...else
```

```
class Main {  
    public static void main(String[] args) {  
        int number = 10;  
  
        // checks if number is greater than 0  
        if (number > 0) {  
            System.out.println("The number is positive.");  
        }  
  
        // execute this block  
        // if number is not greater than 0  
        else {  
            System.out.println("The number is not positive.");  
        }  
  
        System.out.println("Statement outside if...else block");  
    }  
}
```

### 3. Java if...else...if Statement

In Java, we have an if...else...if ladder, that can be used to execute one block of code among multiple other blocks.

```
if (condition1) {  
    // codes  
}  
else if(condition2) {  
    // codes  
}  
else if (condition3) {  
    // codes  
}  
.  
.  
else {  
    // codes  
}
```

Talent Battle

### 1st Condition is true

```
int number = 2;  
if (number > 0) {  
    // code  
}  
else if (number == 0){  
    // code  
}  
else {  
    //code  
}  
//code after if
```

### 2nd Condition is true

```
int number = 0;  
if (number > 0) {  
    // code  
}  
else if (number == 0){  
    // code  
}  
else {  
    //code  
}  
//code after if
```

### All Conditions are false

```
int number = -2;  
if (number > 0) {  
    // code  
}  
else if (number == 0){  
    // code  
}  
else {  
    //code  
}  
//code after if
```

```
class Main {  
    public static void main(String[] args) {  
        int number = 0;  
        // checks if number is greater than 0  
        if (number > 0) {  
            System.out.println("The number is positive.");  
        }  
        // checks if number is less than 0  
        else if (number < 0) {  
            System.out.println("The number is negative.");  
        }  
        // if both condition is false  
        else {  
            System.out.println("The number is 0.");  
        }  
    }  
}
```



## 4. Java Nested if..else Statement

In Java, it is also possible to use if..else statements inside an if...else statement. It's called the nested if...else statement.

Here's a program to find the largest of 3 numbers using the nested if..else statement.

```
class Main {
    public static void main(String[] args) {
        // declaring double type variables
        Double n1 = -1.0, n2 = 4.5, n3 = -5.3, largest;
        // checks if n1 is greater than or equal to n2
        if (n1 >= n2) {
            // if..else statement inside the if block
            // checks if n1 is greater than or equal to n3
            if (n1 >= n3) {
                largest = n1;
            }
            else {
                largest = n3;
            }
        } else {
            // if..else statement inside else block
            // checks if n2 is greater than or equal to n3
            if (n2 >= n3) {
                largest = n2;
            }
            else {
                largest = n3;
            }
        }
        System.out.println("Largest Number: " + largest);
    }
}
```

# Java switch Statement

The switch statement allows us to execute a block of code among many alternatives.  
The syntax of the switch statement in Java is:

```
switch (expression) {  
    case value1:  
        // code  
        break;  
    case value2:  
        // code  
        break;  
    ...  
  
    default:  
        // default statements  
}
```

// Java Program to check the size using the switch...case statement

```
class Main {  
    public static void main(String[] args) {  
        int number = 44;  
        String size;  
        // switch statement to check size  
        switch (number) {  
            case 29:  
                size = "Small";  
                break;  
            case 42:  
                size = "Medium";  
                break;  
            // match the value of week  
            case 44:  
                size = "Large";  
                break;  
            case 48:  
                size = "Extra Large";  
                break;  
            default:  
                size = "Unknown";  
                break;  
        }  
        System.out.println("Size: " + size);  
    }  
}
```



# Java Program to Make a Simple Calculator Using switch..case

import java.util.Scanner;

```
class Main {
    public static void main(String[] args) {

        char operator;
        Double number1, number2, result;

        // create an object of Scanner class
        Scanner input = new Scanner(System.in);

        // ask users to enter operator
        System.out.println("Choose an operator: +, -, *, or /");
        operator = input.next().charAt(0);

        // ask users to enter numbers
        System.out.println("Enter first number");
        number1 = input.nextDouble();

        System.out.println("Enter second number");
        number2 = input.nextDouble();

        switch (operator) {

            // performs addition between numbers
            case '+':
                result = number1 + number2;
                System.out.println(number1 + " + " + number2 + " = " + result);
                break;

            // performs subtraction between numbers
            case '-':
                result = number1 - number2;
                System.out.println(number1 + " - " + number2 + " = " + result);
                break;

            // performs multiplication between numbers
            case '*':
                result = number1 * number2;
                System.out.println(number1 + " * " + number2 + " = " + result);
                break;

            // performs division between numbers
            case '/':
                result = number1 / number2;
                System.out.println(number1 + " / " + number2 + " = " + result);
                break;

            default:
                System.out.println("Invalid operator!");
                break;
        }

        input.close();
    }
}
```

