Assignment – 6th (Operators & Loops)

1. What are the Conditional operators in Java?

Ans: - Conditional operators in Java are special symbols used to make decisions in your code. The two types of conditional operators in Java are:

1. **Ternary operator (?:) -** a shorthand way to write an if-else statement that returns one of two values based on the result of a condition.
2. **Instanceof operator -** used to check if an object belongs to a certain class or if it's an instance of a subclass of that class.

These operators allow you to write more concise and efficient code by evaluating conditions and making decisions based on the result.

2. What are the types of operators based on the number of operands?

Ans: - Operators in programming languages are categorized based on the number of operands they take. The following are the types of operators based on the number of operands:

- **Unary operators: -** Take only one operand. For example, the increment (++) and decrement (--) operators. Work with only one operand.
- **Binary operators: -** Take two operands. For example, the arithmetic operators (+, -, *, /, %) and the comparison operators (==, !=, >, <, >=, <=). Work with two operands
- **Ternary operators: -** Take three operands. For example, the conditional operator (?:) in Java. Work with three operands.

3. What is the use of switch case in Java programming?

Ans: - The ***`switch`*** case statement in Java is a decision-making tool used to execute different code blocks based on the value of an expression. It provides a clean and efficient way to handle multiple conditions in a program.

The ***`switch`*** statement evaluates an expression, and then matches its value against several cases. If a match is found, the corresponding code block is executed. If no match is found, the code in the ***`default`*** case is executed (if specified).

The ***`switch`*** statement is useful when you have many conditions to check, and it provides a more organized and readable alternative to multiple ***`if-else`*** statements. It is commonly used in menu-driven programs, where a user makes a selection from a list of options, and you want to perform a specific task based on their selection.

4. What are the priority levels of arithmetic operation in Java?

Ans: - In Java, the priority or precedence of arithmetic operations determines the order in which operations are carried out. Operations with higher precedence are executed before operations with lower precedence. The following is a list of the priority levels of arithmetic operations in Java, from highest to lowest:

1. Post-increment **(x++)** and post-decrement **(x--)**
2. Pre-increment **(++x)** and pre-decrement **(--x)**
3. Unary plus **(+x)** and unary minus **(-x)**
4. Exponentiation **(x^y)**
5. Multiplication **(x * y)**, division **(x / y)**, and modulo **(x % y)**
6. Addition **(x + y)** and subtraction **(x - y)**

5. What are the conditional statements and use of conditional statements in Java?

Ans: - Conditional statements in Java are a way to control the flow of a program based on specific conditions. They allow the program to make decisions and execute different code blocks depending on the result of a condition. There are two main types of conditional statements in Java: if and switch.

1. **`if` statement: -** This statement is used to execute a block of code if a particular condition is met. The basic syntax is:

```
if (condition) {
        // code to be executed if condition is true
}
```

   You can also use an if statement along with an else statement to execute different code blocks depending on whether the condition is true or false. The basic syntax is:

```
if (condition) {
        // code to be executed if condition is true
} else {
        // code to be executed if condition is false
}
```

2. **'switch' statement: -** This statement is used to execute different code blocks based on the value of an expression. The basic syntax is:

```
switch (expression) {
        case value1:
                // code to be executed if expression is equal to value1
                break;
        case value2:
                // code to be executed if expression is equal to value2
                break;
        ...
        default:
                // code to be executed if expression doesn't match any of the values
}
```

Conditional statements play an important role in programming as they allow for decision-making based on specific conditions, making the code more dynamic and flexible.

6. What is the syntax of if else statement?

Ans: - The syntax of an if-else statement in Java is as follows:

```
if (condition) {
        // code to be executed if condition is true
} else {
        // code to be executed if condition is false
}
```

Here, **`condition`** is a boolean expression that evaluates to either **`true`** or **`false`**. If the condition is **`true`**, the code within the first set of curly braces **`{}`** will be executed. If the condition is **`false`**, the code within the second set of curly braces **`{}`** will be executed instead

7. What are the three types of iterative statements in java?

Ans: - The three types of iteration statements in Java are ***`for`***, ***`while`***, and ***`do-while`***.

- **`for` loop: -** This type of loop is used when you know the number of times you want to execute a certain block of code. The code block inside the loop will be executed repeatedly until the condition is met. The basic syntax is:

```
for (initialization; condition; increment/decrement)
{
        // code to be executed
}
```

- **`while` loop: -** This type of loop is used when you want to execute a certain block of code until a certain condition is met. If the condition is ***`true`***, the code block will be executed and the process repeats until the condition becomes ***`false`***. The basic syntax is:

```
while (condition)
{
        // code to be executed
}
```

- **`do-while` loop: -** This type of loop is similar to a while loop, but with a key difference: the code block within the loop is guaranteed to be executed at least once. If the condition is ***`true`***, the code block will be executed again, repeating until the condition becomes ***`false`***. The basic syntax is:

```
do {
        // code to be executed
} while (condition);
```

8. Write the difference between for loop and do-while loop?

Ans: - The key difference between for and do-while loops is the timing of when the loop condition is evaluated.

In a **`for loop`**, the condition is evaluated before each iteration. If the condition is false from the start, the loop won't run at all. The structure of a for loop includes an initialization, a condition, and an increment/decrement step.

On the other hand, a **`do-while loop`** evaluates the condition after each iteration. The code block inside the loop is guaranteed to run at least once, since the condition is only checked after the first run. The loop will continue to repeat as long as the condition is true.

Another difference is that **`for loops`** *are best suited for situations where the number of iterations is known beforehand, while* **`do-while loops`** *are used when the number of iterations is unknown.*

9. Write a program to print numbers from 1 to 10?

Ans: -

```
public class Main {
        public static void main(String[] args) {
                int i;
                for (i=1; i<=10; i++) {
                        System.out.println(i);
                }
        }
}
```