**Q1. What are the conditional operators in Java?**

Answer: The conditional operators in Java are:

"&&" (logical AND) - returns true if both operands are true.

"||" (logical OR) - returns true if either of the operands is true.

"!" (logical NOT) - returns the reverse of the operand.

For example:

int x = 10;

int y = 20;

if (x == 10 && y == 20) {

System.out.println("Both conditions are true.");

}

if (x == 10 || y == 30) {

System.out.println("One of the conditions is true.");

}

if (!(x == 10)) {

System.out.println("The condition is false.");

}

It's worth noting that the operands in a conditional expression must evaluate to either true or false, so they are often used in combination with

comparison operators such as "==" (equal to), "!=" (not equal to), ">" (greater than), "<" (less than), ">=" (greater than or equal to), and "<=" (less than or equal to).

**Q2. What are the types of operators based on the number of operands?**

Answer: Operators in Java are classified based on the number of operands they take:

Unary operators - take a single operand. Examples include the unary minus operator (-), the increment operator (++), and the decrement operator (--).

Binary operators - take two operands. Examples include the addition operator (+), the subtraction operator (-), the multiplication operator (\*), and the division operator (/).

Ternary operator - take three operands. The only ternary operator in Java is the conditional operator (? :). It allows for a simple and compact way to express a conditional expression.

For example:

int x = 10;

int y = 20;

int result = (x > y) ? x : y;

**Q3. What is the use of switch case in Java Programming?**

Answer:

The "switch" statement in Java is a control statement that allows you to execute a block of code based on the value of a specified expression. It provides an alternative to using a series of "if...else if" statements, and can make your code more readable and maintainable.

The basic syntax of the "switch" statement is as follows:

switch (expression) {

case value1:

// code to be executed if expression == value1

break;

case value2:

// code to be executed if expression == value2

break;

...

case valueN:

// code to be executed if expression == valueN

break;

default:

// code to be executed if expression doesn't match any case value

}

The expression in the "switch" statement is evaluated, and the resulting value is compared to the values in each "case" statement. If a match is found, the code inside the

corresponding case block is executed. If no match is found, the code in the "default" block is executed. The "break" statement at the end of each case block is used to exit the "switch"

statement and prevent execution from falling through to the next case.

**Q4. What are the priority levels of arithmetic Operation in Java?**

Answer: The six arithmetic operators in Java are:

Addition (+)

Subtraction (-)

Multiplication (\*)

Division (/)

Modulus (%)

Increment (++) and decrement (–)

There are two important points to note when using Java operators:

Multiplication, division, and modulo operations have precedence over addition and subtraction operations.

Plus, and minus operators have the same precedence. Operators with the same precedence are evaluated from left to right.

**Q5. What are the conditional statements and use of conditional statements in Java?**

Answer: Conditional statements are statements in Java that allow you to control the flow of your program based on whether a certain condition is true or false.

If statement: The "if" statement is used to execute a block of code only if a certain condition is true. The basic syntax of an "if" statement is as follows:

if (condition) {

// code to be executed if condition is true

}

If...else statement: The "if...else" statement is used to execute one block of code if a certain condition is true and another block of code if the condition is false. The basic syntax of an "if...else" statement is as follows:

if (condition) {

// code to be executed if condition is true

} else {

// code to be executed if condition is false

}

If...else if...else statement: The "if...else if...else" statement is used to execute one of several blocks of code based on multiple conditions. The basic syntax of an "if...else if...else" statement is as follows:

if (condition1) {

// code to be executed if condition1 is true

} else if (condition2) {

// code to be executed if condition1 is false and condition2 is true

} ...

else if (condition) {

// code to be executed if all previous conditions are false and condition is true

} else {

// code to be executed if all previous conditions are false

}

Switch: In Java, Switch statements are similar to if-else-if statements. The switch statement contains multiple blocks of code called cases and a single case is executed based

on the variable which is being switched. The switch statement is easier to use instead of if-else-if statements. It also enhances the readability of the program.

**Q6. What is the syntax of if else statement?**

Answer: If...else if...else statement: The "if...else if...else" statement is used to execute one of several blocks of code based on multiple conditions. The basic syntax of an "if...else if...else" statement is as follows:

if (condition1) {

// code to be executed if condition1 is true

} else if (condition2) {

// code to be executed if condition1 is false and condition2 is true

} ...

else if (conditionN) {

// code to be executed if all previous conditions are false and conditionN is true

} else {

// code to be executed if all previous conditions are false

}

**Q7. What are the 3 types of iterative statements in java?**

Answer:

There are three types of iteration statements in Java:

For loop: The "for" loop is used to repeat a block of code a specified number of times. The basic syntax of a "for" loop is as follows:

for (initialization; condition; increment/decrement) {

// code to be executed

}

While loop: The "while" loop is used to repeat a block of code while a certain condition is true. The basic syntax of a "while" loop is as follows:

while (condition) {

// code to be executed

}

Unlike for and while loops, which test the loop condition at the top of the loop, the do...while loop in C programming checks its condition at the bottom of the loop.

A do...while loop is similar to a while loop, except the fact that it is guaranteed to execute at least one time.

do {

statement(s);

} while( condition );

**Q8. What are the differences between for loop and do-while loop?**

Answer:

The difference between a "for" loop and a "do...while" loop is that a "for" loop tests the condition before executing the loop, while a "do...while" loop tests the condition after executing the loop. This means that a "for" loop may not execute at all if the condition is false, while a "do...while" loop is guaranteed to execute at least once, even if the condition is false.

int i = 5;

for (; i < 5;) {

System.out.println("This is a for loop");

}

do {

System.out.println("This is a do...while loop");

i++;

} while (i < 5);

**Q9. Write a program to print from 1 to 10.**

Answer: public class Main {

public static void main (String [] args) {

for (int i = 1; i <= 10; i++) {

System.out.println(i);

}

}

}

This program uses a for loop to iterate from 1 to 10. On each iteration, the value of i is printed to the console using the System.out.println method. The result will be the following:

1

2

3

4

5

6

7

8

9

10