Q1. What are the Conditional Operators in Java?

Answer: In Java, conditional operators check the condition and decides the desired result on the basis of both conditions. In this section, we will discuss the conditional operator in Java.

Types of Conditional Operator

There are three types of the conditional operator in Java:

- Conditional AND
- Conditional OR
- Ternary Operator

Operator	**Symbol**
Conditional or Logical AND	&&
Conditional or Logical OR	||
Ternary Operator	?:

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

Answer: In Java, operators can be categorized based on the number of operands they work with. Here are the different types of operators in Java:

1.Unary Operators: Unary operators work with a single operand.

- Increment and Decrement Operators: ++ (increment), -- (decrement)
- Unary Plus and Minus Operators: + (unary plus), - (unary minus)
- Logical Complement Operator: ! (logical NOT)
- Bitwise Complement Operator: ~ (bitwise NOT)

2.Binary Operators: Binary operators work with two operands.

- Arithmetic Operators: + (addition), - (subtraction), * (multiplication), / (division), % (modulus)
- Relational Operators: < (less than), > (greater than), <= (less than or equal to), >= (greater than or equal to), == (equal to), != (not equal to)
- Logical Operators: && (logical AND), || (logical OR)
- Bitwise Operators: & (bitwise AND), | (bitwise OR), ^ (bitwise XOR)
- Assignment Operators: = (simple assignment), += (add and assign), -=, *=, /=, %=, &=, |=, ^=

3.Ternary Operator: The ternary operator is the only operator in Java that takes three operands.

- Conditional Operator: condition? expression1: expression2

Q3. What is the use of Switch case in Java programming?

Answer: The switch case in java is used to select one of many code blocks for execution.

Break keyword: As java reaches a break keyword, the control breaks out of the switch block. The execution of code stops on encountering this keyword, and the case testing inside the block ends as the match is found. A lot of execution time can be saved because it ignores the rest of the code's execution when there is a break.

Default keyword: The keyword is used to specify the code executed when the expression does not match any test case.

Q4. What are the conditional Statements and use of conditional statements in Java?

Answer: Conditional statements in Java allow you to control the flow of your program based on certain conditions. They help you make decisions and execute different blocks of code depending on whether a condition is true or false. The most commonly used conditional statements in Java are:

1.if statement:

The if statement allows you to execute a block of code if a given condition is true. It has the following syntax:

```
If(condition) {
        // Code to be executed if the condition is true
}
```

2.if-else Statement:

The if-else statement extends the if statement by allowing you to execute different code blocks based on whether a condition is true or false. It has the following syntax:

```
If (condition) {
        // Code to be executed if the condition is true
} else {
        // Code to be executed if the condition is false
}
```

3. if-else if-else Statement:

The if-else if-else statement allows you to check multiple conditions and execute different code blocks based on the conditions. It has the following syntax:

```
If (condition1) {
        // Code to be executed if condition1 is true
} else if (condition2) {
        // Code to be executed if condition2 is true
} else {
        // Code to be executed if all conditions are false
}
```

Q5. What is the syntax of if else statement?

Answer: if-else Statement:

The if-else statement extends the if statement by allowing you to execute different code blocks based on whether a condition is true or false. It has the following syntax:

```
If (condition) {

        // Code to be executed if the condition is true

} else {

        // Code to be executed if the condition is false

}
```

Q6. How do you compare two strings in Java?

Answer: String is a sequence of characters. In Java, objects of String are immutable which means they are constant and cannot be changed once created.
Below are 5 ways to compare two Strings in Java:

1.Using user-defined function: Define a function to compare values with following conditions:

1. if (string1 > string2) it returns a **positive value**.
2. if both the strings are equal lexicographically i.e. (string1 == string2) it returns **0**.
3. if (string1 < string2) it returns a **negative value**.

The value is calculated as **(int)str1.charAt(i) – (int)str2.charAt(i)**

2. Using String.equals(): In java, string equals () method compares the two given strings based on the data/content of the string. If all the contents of both the strings are same then it return true.If any character does not match, then it return false

Syntax:

```
Str1.equals(str2);
```

Here str1 and str2 both are the strings which are to be compared.

3.Using String.equalslgnoreCase() : The String.equalsIgnoreCase() method compares two strings regardless of the case (lower or upper) of the string. This method returns true if the argument is not null and the contents of both the Strings are same ignoring case, else false.

Syntax:

```
str2.equalsIgnoreCase(str1);
```

Here str1 and str2 both are the strings which are to be compared.

4. Using Objects.equals(): Object.equals(Object a, Object b) method returns true if the arguments are equal to each other and false otherwise. Consequently, if both arguments are null, true is returned and if exactly one argument is null, false is returned. Otherwise, equality is determined by using the equals () method of the first argument.

Syntax:

```
public static boolean equals (Object a, Object b)
```

Here a and b both are the string objects which are to be compared.

5.Using String.compareTo():

Syntax:

```
int str1.compareTo(String str2)
```

Working:
It compares and returns the following values as follows:

1.if (string1 > string2) it returns a **positive value**.
2.if both the strings are equal lexicographically
i.e. (string1 == string2) it returns **0**.
3.if (string1 < string2) it returns a **negative value**.

Q7. What is Mutable String in Java Explain with an example

Answer: Mutable means changing over time or that can be changed. In a mutable string, we can change the value of the string and JVM doesn't create a new object. In a mutable string, we can change the value of the string in the same object.
To create a mutable string in java, Java has two classes StringBuffer _and StringBuilder_where the String class is used for the immutable string**.**

Q8. Write a program to sort a String Alphabetically

Answer:

// Java program to sort a string in alphabetical order

```
import java.util.*;

public class Main
{
    public static void main (String[] args)
    {
        String str;
        Scanner sc = new Scanner (System.in);
  System.out.println("Enter the string: ");
        str = sc.nextLine();
        int j = 0;
        char temp = 0;
        char[] chars = str.toCharArray();
        for (int i = 0; i < chars.length; i++) {
            for (j = 0; j < chars.length; j++) {
                if (chars[j] > chars[i]) {
                    temp = chars[i];
                    chars[i] = chars[j];
                    chars[j] = temp;
                }
            }
        }
    System.out.println("The sorted string is : ");
    for (int i = 0; i < chars.length; i++) {
        System.out.print(chars[i]);
```

```
            }

        }
}
```

Q9. Write a program to check if the letter 'e' is present in the word 'Umbrella'.

Answer:

```
public class LetterCheck {
    public static void main(String[] args) {
        String word = "Umbrella";
        boolean isLetterPresent = false;

        for (int i = 0; i < word.length(); i++) {
            if (word.charAt(i) == 'e') {
                isLetterPresent = true;
                break;
            }
        }

        if (isLetterPresent) {
            System.out.println("The letter 'e' is present in the word.");
        } else {
            System.out.println("The letter 'e' is not present in the word.");
        }
    }
}
```

Q10.Where exactly is the string constant pool located in the memory?

Answer:

In Java, the string constant pool is a part of the runtime constant pool, which is a data structure that resides in the Java Virtual Machine (JVM) memory. The JVM memory is divided into different regions, and the string constant pool is typically located in the method area or non-heap memory.

The method area is a part of the JVM memory where class structures, method bytecode, constant pool, and other metadata related to classes and methods are stored. The string constant pool is a specific section within the method area that holds the literal string values used in the program.

It's important to note that the exact implementation of the JVM memory layout can vary across different JVM implementations. However, regardless of the specific implementation details, the string constant pool is always a part of the JVM memory and is typically located in the method area or non-heap memory.