💡 Q1. What are the Conditional Operators in Java?

Conditional operators in Java are used to evaluate Boolean expressions and return a value based on the outcome of the evaluation. There are three conditional operators in Java:

- **Conditional AND (&&):** This operator returns true if both operands are true. If either operand is false, the operator returns false.
- **Conditional OR (||):** This operator returns true if either operand is true. If both operands are false, the operator returns false.
- **Ternary Operator (? :):** This operator is also known as the conditional operator. It takes three operands and returns the value of the second operand if the first operand is true, and the value of the third operand if the first operand is false.

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

- Unary operators require one operand. They operate on a single operand and return a single value. For example, the - operator is a unary operator that negates the value of its operand.
- Binary operators require two operands. They operate on two operands and return a single value. For example, the + operator is a binary operator that adds the two operands together.
- Ternary operators require three operands. They operate on three operands and return a single value. For example, the ?: operator is a ternary operator that returns one value if its first operand is true, and another value if its first operand is false.

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

The switch case statement in Java is used to select one of many code blocks for execution. It is a decision-making statement that allows you to execute different blocks of code based on the value of a variable.

The switch statement works by evaluating the value of a variable and then comparing it to a list of values. If the value of the variable matches one of the values in the list, the corresponding code block is executed. If the value of the variable does not match any of the values in the list, the default code block is executed.

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

There are three main types of conditional statements in Java:

- **if statement:** The if statement is the simplest type of conditional statement. It allows you to execute a block of code if a condition is true.
- **if-else statement:** The if-else statement allows you to execute different blocks of code depending on whether a condition is true or false.
- **switch statement:** The switch statement allows you to execute different blocks of code based on the value of a variable.

💡 Q5.What is the syntax of if else statement?

The syntax of the if else statement is as follows:

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

} else {

// code block to be executed if condition is false

}

💡 Q6.How do you compare two strings in Java?

There are two ways to compare two strings in Java:

1) Using equals() Method:

The String class equals() method compares the original content of the string. It compares values of string for equality. String class provides the following two methods:

- **public boolean equals(Object another)** compares this string to the specified object.
- **public boolean equalsIgnoreCase(String another)** compares this string to another string, ignoring case.

```
class Test{

public static void main(String args[]){
  String s1="Smith";
  String s2="Smith";
  String s3=new String("Smith");
  String s4="John";
```

```
  String s5="Sachin";
  String s6="SACHIN";
  System.out.println(s1.equals(s2));//true
  System.out.println(s1.equals(s3));//true
  System.out.println(s1.equals(s4));//false
  System.out.println(s5.equals(s6));//false
  System.out.println(s5.equalsIgnoreCase(s6));//true
 }
}
```

2) Using == operator

The == operator compares references not values.

```
class Test{
 public static void main(String args[]){
  String s1="Smith";
  String s2="Smith";
  String s3=new String("Smith");
//because both refer to same instance
  System.out.println(s1==s2);//true
//because s3 refers to instance created in nonpool
  System.out.println(s1==s3);//false
 }
}
```

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

With Mutable string, we can change the contents of an existing object, which does not create a new object. Therefore mutable strings are those strings whose content can be changed without creating a new object. StringBuffer and StringBuilder are mutable versions of String in java, whereas the java String class is immutable. Immutable objects are those objects whose contents cannot be modified once created. Whenever an immutable object's content is changed, there will be a creation of a new object in the heap area.

Example:

```
public class MutableStringDemo{

public static void main(String args[]){

StringBuffer sBuffer1=new StringBuffer("Welcome");
```

```
System.out.println("Original String is ::: " + sBuffer1 + ":: having length " +
sBuffer1.length());

 //using append method

 sBuffer1.append(" To Java"); System.out.println("Modified String after append is :: " +
sBuffer1 + " :: having length " + sBuffer1.length());

//using reverse method sBuffer1.reverse(); System.out.println("Modified String after
Reverse is :: " + sBuffer1);

 }

}
```

Output:

Original String is ::: Welcome:: having length 7

Modified String after append is :: Welcome To Java :: having length 15

Modified String after Reverse is :: avaJ oT emocleW

💡 **Q8.Write a program to sort a String Alphabetically**

```
import java.util.Arrays;

public class SortStringAlphabetically {

    public static void main(String[] args) {

        String string = "hero";

        char[] characters = string.toCharArray();

        Arrays.sort(characters);

        String sortedString = new String(characters);

        System.out.println(sortedString);

    }

}
```

Output: ehor

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

```
public class CheckLetterPresent {
    public static void main(String[] args) {
        String word = "Umbrella";
        char letter = 'e';
        boolean isPresent = false;

        for (int i = 0; i < word.length(); i++) {
            if (word.charAt(i) == letter) {
                isPresent = true;
                break;
            }
        }
        if (isPresent) {
            System.out.println("The letter 'e' is present in the word 'Umbrella'");
        } else {
            System.out.println("The letter 'e' is not present in the word 'Umbrella'");
        }
    }
}
```

Output: The letter 'e' is present in the word 'Umbrella'

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

The string constant pool is located in the heap memory of a Java program. The heap is a region of memory that is used to store objects that are created by the program. The string constant pool is a special area of the heap that is used to store strings that are literals in the program.

When a Java program is compiled, the compiler checks to see if the string literals that are used in the program have already been defined in the string constant pool. If they have, the compiler simply references the existing string in the pool. If they have not, the compiler creates a new string in the pool and stores it there.

The string constant pool is used to improve the performance of Java programs. By storing strings in the pool, the compiler can avoid having to create new strings every time they are used. This can save a significant amount of time, especially in programs that use a lot of strings.