Q1. What are the Conditional Operators in Java?

ANS

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

Conditional AND (&&): This operator returns true if both operands are true. Otherwise, it returns false.
Conditional OR (||): This operator returns true if either operand is true. Otherwise, it returns false.

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

ANS

There are three types of operators in Java based on the number of operands they require:

Unary operators operate on a single operand. Examples of unary operators in Java include +, -, !, ++, and --.

Binary operators operate on two operands. Examples of binary operators in Java include +, -, *, /, %, <, >, ==, !=, &&, ||, ^, |, and &.

Ternary operators operate on three operands. The only ternary operator in Java is the conditional operator, which is written as ?:.

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

ANS

The switch case statement in Java is used to select one of many code blocks for execution. The switch statement works by first evaluating an expression. The value of the expression is then compared to a series of case values. If the value of the expression matches one of the case values, the corresponding code block is executed. If the value of the expression does not match any of the case values, the default code block is executed.

EX

```
int day = 3;

switch (day) {
  case 1:
    System.out.println("The day is Monday");
    break;
  case 2:
    System.out.println("The day is Tuesday");
    break;
  case 3:
    System.out.println("The day is Wednesday");
    break;
  default:
    System.out.println("The day is unknown");
}
```

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

ANS

Conditional statements are used to make decisions in Java code. They allow you to execute different bloc ks of code depending on whether a certain condition is true or false.

There are three main types of conditional statements in Java:

If statement: The if statement is the most basic conditional statement. It allows you to execute a block of c ode if a certain condition is true.
Else if statement: The else if statement allows you to execute a different block of code if the first condition is false.
Switch statement: The switch statement allows you to execute one of many blocks of code depending on t he value of a variable.

EX

```
int number = 0;

if (number > 0) {
  System.out.println("The number is positive");
} else if (number < 0) {
  System.out.println("The number is negative");
} else {
  System.out.println("The number is zero");
}
```

Q5.What is the syntax of if else statement?

ANS

The condition is an expression that evaluates to a boolean value. If the value of the expression is true, the code in the if block is executed. Otherwise, the code in the else block is executed.

The if block and the else block are optional. If you only want to execute code if the condition is true, you c an omit the else block.

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

EX

```
int number = 10;

if (number > 0) {
  System.out.println("The number is positive");
} else {
  System.out.println("The number is not positive");
}
```

Q6.How do you compare two strings in Java?

ANS

There are three ways to compare two strings in Java:

Using the equals() method: The equals() method compares two strings based on their content. If the two s trings have the same content, the equals() method returns true. Otherwise, it returns false.

EX
```
String string1 = "Hello, world!";
String string2 = "Hello, world!";
boolean equal = string1.equals(string2);
```

Using the compareTo() method: The compareTo() method compares two strings based on their Unicode v alues. If the first string has a lower Unicode value than the second string, the compareTo() method return s a negative value. If the first string has a higher Unicode value than the second string, the compareTo() method returns a positive value. If the two strings have the same Unicode value, the compareTo() method returns 0.

EX
```
String string1 = "Hello, world!";
String string2 = "Hello, world!";
int comparisonResult = string1.compareTo(string2);
```

Using the == operator: The == operator compares two strings based on their reference. If the two strings r efer to the same object, the == operator returns true. Otherwise, it returns false.

EX
```
String string1 = "Hello, world!";
String string2 = "Hello, world!";
boolean equal = string1 == string2;
```

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

ANS

In Java, a mutable string is a string object whose content can be changed after it is created. This is in con trast to an immutable string, whose content cannot be changed after it is created.

The String class in Java is immutable, while the StringBuffer and StringBuilder classes are mutable.

Here is an example of how to create a mutable string in Java:

```
StringBuffer mutableString = new StringBuffer("Hello, world!");
mutableString.append("!");
```

In this example, a StringBuffer object is created with the value "Hello, world!". The append() metho d is then used to add the character ! to the end of the string. The mutableString object now has the value " Hello, world!".

Q8.Write a program to sort a String Alphabetically

ANS

```
import java.util.Arrays;

public class SortStringAlphabetically {
```

```
    public static void main(String[] args) {
        String[] strings = {"hello", "world", "java", "programming"};

        // Sort the strings alphabetically
        Arrays.sort(strings);

        // Print the sorted strings
        for (String string : strings) {
            System.out.println(string);
        }
    }
}
```

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

ANS

```
public class CheckLetter {

    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’");
        }
    }
}
```

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

ANS

The string constant pool is located in the permanent generation area of the Java heap.