

Pre-Experiment Exercise:

a) Explain various Data types in java.

In Java, data types define the type and size of values that can be stored in variables. Java has two categories of data types: primitive data types and reference data types. Here's an explanation of each data type:

Primitive Data Types:

Primitive data types are the basic building blocks of data in Java. They are predefined by the language and have a fixed size. There are eight primitive data types in Java:

- a. byte: 8-bit signed integer. Range: -128 to 127.
- b. short: 16-bit signed integer. Range: -32,768 to 32,767.
- c. int: 32-bit signed integer. Range: -2^{31} to $2^{31} - 1$.
- d. long: 64-bit signed integer. Range: -2^{63} to $2^{63} - 1$.
- e. float: 32-bit floating-point number. It can represent decimal values with single-precision.
- f. double: 64-bit floating-point number. It can represent decimal values with double-precision.
- g. char: 16-bit Unicode character. It can store a single character.
- h. boolean: Represents a boolean value (true or false).

Reference Data Types:

Reference data types refer to objects and provide a reference to the memory address where the data is stored. These data types do not store the actual data but rather point to the memory location where the data is kept. Some common reference data types include:

- a. Class types: References to objects created from classes.
- b. Interface types: References to objects implemented from interfaces.
- c. Array types: References to arrays.

b) Control Structures in Java.

i. if statement: It checks a Boolean condition: true or false. It executes a block of code if the specified condition is true.

ii. if-else statement: Extends the basic if statement by providing an alternative block of code to execute when the condition is false.

iii. if-else-if ladder: Allows you to check multiple conditions in sequence. It provides an alternative block of code for each condition.

iv. nested if statement: An if statement inside another if statement. It allows you to test for multiple conditions in a hierarchical manner.

c) Switch statement: The Java switch statement executes one statement from multiple conditions. It is like an if-else-if ladder statement.

d) Looping statements:

i. for loop: Used to iterate over a range of values or elements for a specific number of times.

ii. while loop: Executes a block of code repeatedly as long as the specified condition is true.

iii. do-while loop: Similar to the while loop, but guarantees the execution of the block of code at least once, even if the condition is false from the beginning

iv. for-each loop (enhanced for loop): Used to iterate over elements in an array or a collection without the need for explicit index management.

These control structures and looping statements are essential for controlling the flow of execution and handling repetitive tasks in Java programs.

Post Experiment Exercise

A. Extended Theory:

1. Explain entry controlled loop and exit controlled loop used in Java with example.

In Java, there are two types of loops based on the condition evaluation: entry-controlled loop and exit-controlled loop.

Entry-controlled loop (Pre-tested loop):

An entry-controlled loop evaluates the condition before entering the loop body. If the condition is true, the loop body is executed. If the condition is false from the beginning, the loop will not execute at all. The most common entry-controlled loop in Java is the "for" loop.

Example of entry-controlled loop (for loop):

```
// Entry-controlled loop (for loop)
for (int i = 1; i <= 5; i++) {
    System.out.println("Iteration: " + i);
}
```

In this example, the "for" loop will iterate from 1 to 5. Before each iteration, the condition $i \leq 5$ is checked. As long as the condition is true, the loop will continue to execute. Once the condition becomes false (i.e., when i becomes 6), the loop exits.

Exit-controlled loop (Post-tested loop):

An exit-controlled loop evaluates the condition after executing the loop body. The loop body is guaranteed to be executed at least once before checking the condition. The most common exit-controlled loop in Java is the "do-while" loop.

Example of exit-controlled loop (do-while loop):

```
// Exit-controlled loop (do-while loop)
int count = 1;
do {
    System.out.println("Count: " + count);
    count++;
} while (count <= 5);
```

In this example, the "do-while" loop will execute the loop body first and then check the condition $\text{count} \leq 5$. If the condition is true, the loop will execute again. If the condition is false (i.e., when count becomes 6), the loop exits.

The choice between entry-controlled and exit-controlled loops depends on the specific requirements of your program. If you want to execute the loop only if a certain condition is met, and the loop may not execute at all, then an entry-controlled loop like the "for" loop is appropriate. If you want to ensure that the loop body is executed at least once, then an exit-controlled loop like the "do-while" loop is suitable.

2. Explain the use of break and continue statement and differentiate between them.

In Java, the break and continue statements are used to alter the flow of control in loops. They both provide ways to manage the execution of loops based on certain conditions.

Break statement:

The break statement is used to terminate the current loop prematurely. When encountered within a loop, it immediately exits the loop and continues executing the code after the loop. It is often used when a specific condition is met, and there is no need to continue the loop iterations.

Continue statement:

The continue statement is used to skip the current iteration of a loop and proceed with the next iteration. When encountered within a loop, it immediately jumps to the loop's next iteration, ignoring the remaining code in the loop body for the current iteration.

	Break	Continue
Usage	Used to exit the loop entirely and continue executing the code after the loop.	Used to skip the current iteration and proceed with the next iteration of the loop.
Effect of loop execution	Terminates the loop execution and moves to the code after the loop.	Skips the remaining code in the loop body for the current iteration and moves to the next iteration.
Loop control	Controls the termination of the loop.	Controls the termination of the loop.

In summary, both break and continue statements allow you to control the flow of execution within loops. break is used to terminate the loop entirely, while continue is used to skip the current iteration and move to the next one.

B. Questions/Programs:

1. Write a Java program that counts number of alphabets, words, digits, special symbols and blank spaces in a given string.

```
import java.util.Scanner;

public class StringAnalysis {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String input = scanner.nextLine();
        scanner.close();

        int alphabetCount = 0;
        int wordCount = 0;
        int digitCount = 0;
        int specialSymbolCount = 0;
        int blankSpaceCount = 0;

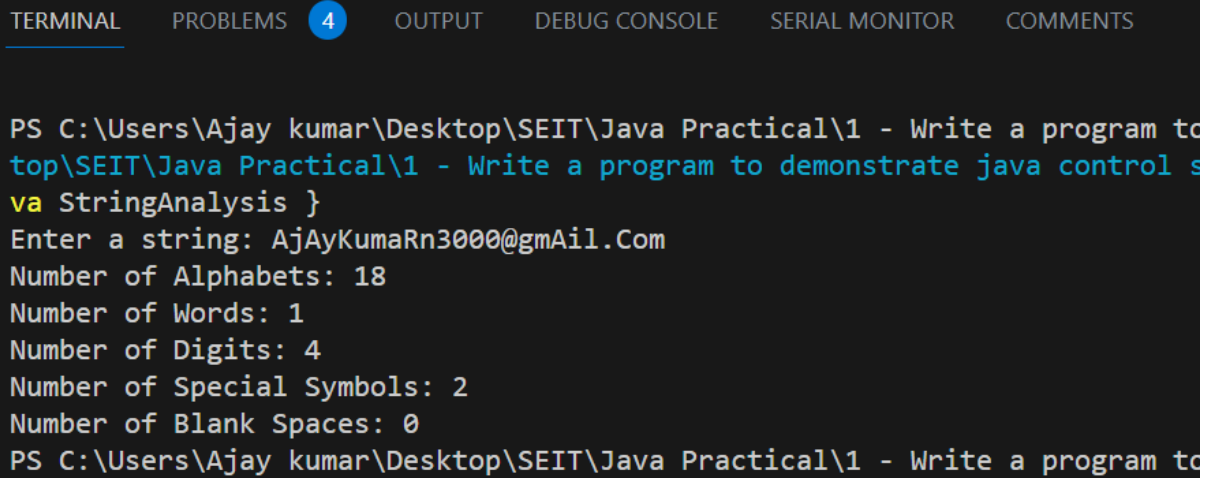
        for (int i = 0; i < input.length(); i++) {
            char ch = input.charAt(i);

            if (Character.isLetter(ch)) {
                alphabetCount++;
            } else if (Character.isDigit(ch)) {
                digitCount++;
            } else if (Character.isWhitespace(ch)) {
                blankSpaceCount++;
            } else {
                specialSymbolCount++;
            }
        }

        // Count words using split() method
        String[] words = input.split("\\s+");
        wordCount = words.length;

        System.out.println("Number of Alphabets: " + alphabetCount);
        System.out.println("Number of Words: " + wordCount);
        System.out.println("Number of Digits: " + digitCount);
        System.out.println("Number of Special Symbols: " +
specialSymbolCount);
        System.out.println("Number of Blank Spaces: " + blankSpaceCount);
    }
}
```

Output:



The screenshot shows an IDE interface with a terminal window. The terminal has tabs for 'TERMINAL', 'PROBLEMS', '4', 'OUTPUT', 'DEBUG CONSOLE', 'SERIAL MONITOR', and 'COMMENTS'. The 'TERMINAL' tab is active. The text in the terminal is as follows:

```
PS C:\Users\Ajay kumar\Desktop\SEIT\Java Practical\1 - Write a program to  
top\SEIT\Java Practical\1 - Write a program to demonstrate java control s  
va StringAnalysis }  
Enter a string: AjAyKumaRn3000@gmAil.Com  
Number of Alphabets: 18  
Number of Words: 1  
Number of Digits: 4  
Number of Special Symbols: 2  
Number of Blank Spaces: 0  
PS C:\Users\Ajay kumar\Desktop\SEIT\Java Practical\1 - Write a program to
```

2. Write a Java program to count vowels and consonants in a given string.

```
import java.util.Scanner;

public class VowelConsonantCounter {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String input = scanner.nextLine().toLowerCase();
        scanner.close();

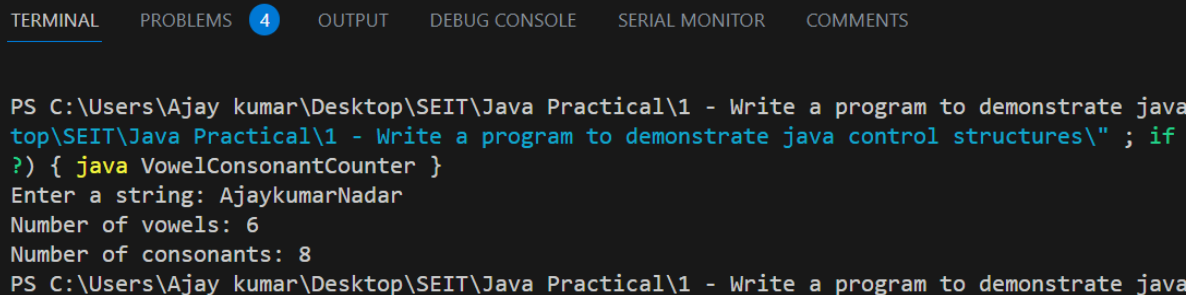
        int vowelCount = 0;
        int consonantCount = 0;

        for (int i = 0; i < input.length(); i++) {
            char ch = input.charAt(i);

            // Check if the character is a letter
            if (Character.isLetter(ch)) {
                // Check if the letter is a vowel
                if (isVowel(ch)) {
                    vowelCount++;
                } else {
                    consonantCount++;
                }
            }
        }
        System.out.println("Number of vowels: " + vowelCount);
        System.out.println("Number of consonants: " + consonantCount);
    }

    public static boolean isVowel(char ch) {
        return ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch ==
'u';
    }
}
```

Output:



The screenshot shows an IDE with a terminal window. The terminal has tabs for TERMINAL, PROBLEMS (with a blue circle containing the number 4), OUTPUT, DEBUG CONSOLE, SERIAL MONITOR, and COMMENTS. The terminal text is as follows:

```
PS C:\Users\Ajay kumar\Desktop\SEIT\Java Practical\1 - Write a program to demonstrate java
top\SEIT\Java Practical\1 - Write a program to demonstrate java control structures\" ; if
?) { java VowelConsonantCounter }
Enter a string: AjaykumarNadar
Number of vowels: 6
Number of consonants: 8
PS C:\Users\Ajay kumar\Desktop\SEIT\Java Practical\1 - Write a program to demonstrate java
```