1. Write a for loop that prints the even numbers from 1 to 20.

```
public class EvenNumbers {
   public static void main(String[] args) {
      System.out.println("Even numbers from 1 to 20:");
      for (int i = 1; i <= 20; i++) {
         if (i % 2 == 0) {
            System.out.println(i);
         }
      }
   }
}
```

```
o\p=  Even numbers from 1 to 20:
2
4
6
8
10
12
14
16
18
20
```

2.Create a while loop that prompts the user for their flight choice until a valid number is entered.

```
import java.util.Scanner;

public class FlightChoice {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      int flightNumber = -1; // Initialize to an invalid value

      while (flightNumber <= 0) {
         System.out.print("Enter a valid flight number (positive number): ");
```

```
        if (scanner.hasNextInt()) { // Check if the input is an integer
            flightNumber = scanner.nextInt();
            if (flightNumber > 0) {
                System.out.println("You entered a valid flight number: " + flightNumber);
            } else {
                System.out.println("Invalid flight number. Please enter a positive number.");
            }
        } else {
            System.out.println("Invalid input. Please enter a numeric value.");
            scanner.next(); // Consume invalid input
        }
    }
    scanner.close();
  }
}
```

0\p= Enter a valid flight number (positive number): 123
You entered a valid flight number: 123

3. Discuss the pros and cons of using different types of loops for iterating through an array of numbers.

Summary Table of Pros and Cons

Loop Type	Pros	Cons
`for`	Fine-grained control, index access, reverse iteration possible	Verbose, prone to off-by-one errors
`while`	Flexible for unknown iteration counts	Requires careful index management, risk of infinite loops
Enhanced `for`	Simplified, concise, reduces chance of errors	No index access, not suitable for modifying array during iteration

For ex -

```
public class ForLoopExample {
    public static void main(String[] args) {
        int[] numbers = {10, 20, 30, 40, 50};

        System.out.println("Using for loop:");
        for (int i = 0; i < numbers.length; i++) {
            System.out.println("Index " + i + ": " + numbers[i]);
```

```
        }
    }
}
```

While ex-

```
public class WhileLoopExample {
    public static void main(String[] args) {
        int[] numbers = {10, 20, 30, 40, 50};
        int i = 0;

        System.out.println("Using while loop:");
        while (i < numbers.length) {
            System.out.println("Index " + i + ": " + numbers[i]);
            i++;
        }
    }
}
```

4.	Write a Java program that uses a for loop to print the first 10 numbers of the Fibonacci sequence.

```
public class FibonacciSequence {
    public static void main(String[] args) {
        int first = 0, second = 1;

        System.out.println("The first 10 numbers of the Fibonacci sequence are:");
        for (int i = 1; i <= 10; i++) {
            System.out.print(first + " "); // Print the current Fibonacci number
            int next = first + second; // Calculate the next number
            first = second; // Update first
            second = next; // Update second
```

```
        }
      }
    }
```

o\p= The first 10 numbers of the Fibonacci sequence are:

0 1 1 2 3 5 8 13 21 34

5.	Create a Java program using a while loop to calculate the sum of integers from 1 to 100.

```
public class SumOfIntegers {
  public static void main(String[] args) {
    int sum = 0;
    int num = 1;

    while (num <= 100) {
      sum += num; // Add the current number to the sum
      num++;     // Increment the number
    }

    System.out.println("The sum of integers from 1 to 100 is: " + sum);
  }
}
```

o/p= The sum of integers from 1 to 100 is: 5050

6.	Implement a do-while loop that prompts the user to enter a number until they enter a negative number.

```
import java.util.Scanner;

public class DoWhileNegativeNumber {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    int number;

    do {
```

```
            System.out.print("Enter a number (negative to quit): ");
            number = scanner.nextInt();
        } while (number >= 0);

        System.out.println("You entered a negative number. Program terminated.");
        scanner.close();
    }
}
```

o/p=Enter a number (negative to quit): 5

Enter a number (negative to quit): 10

Enter a number (negative to quit): 0

Enter a number (negative to quit): -3

You entered a negative number. Program terminated.

7.	Write a Java program that demonstrates the use of the continue statement in a loop.
8.	Initialize and print a 2D array of integers in Java.

```
7= public class ContinueStatementExample {
    public static void main(String[] args) {
        System.out.println("Numbers from 1 to 10, skipping even numbers:");
        for (int i = 1; i <= 10; i++) {
            if (i % 2 == 0) {
                continue; // Skip the current iteration if the number is even
            }
            System.out.println(i); // Print odd numbers only
        }
    }
}
```

o/p= Numbers from 1 to 10, skipping even numbers:

1

3

5

7

9

8=public class TwoDArrayExample {

```
    public static void main(String[] args) {
        // Initialize a 2D array
        int[][] numbers = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };

        System.out.println("2D Array:");
        // Print the 2D array
        for (int i = 0; i < numbers.length; i++) {
            for (int j = 0; j < numbers[i].length; j++) {
                System.out.print(numbers[i][j] + " "); // Print each element
            }
            System.out.println(); // Newline after each row
        }
    }
}
```

o/p=2D Array:

1 2 3

4 5 6

7 8 9

9.	Compare and contrast the use of for loops and while loops. When would you prefer one over the other?
10.	Analyze the impact of using a break statement in nested loops. What considerations should be made?

11.	How do arrays improve the organization and management of data in Java? Discuss with examples.
12.	Examine the differences in memory allocation for single-dimensional and multidimensional arrays.
13.	Discuss the potential pitfalls of using uninitialized arrays in Java.

9=

Comparison:

Aspect	For Loop	While Loop
Purpose	Used when the number of iterations is known.	Used when the number of iterations is unknown.
Structure	Initialization, condition, and increment are part of the loop header.	Condition is in the loop header; initialization and increment are outside.
Readability	More concise and better for counter-controlled loops.	More flexible for loops driven by dynamic conditions.
Example Use Case	Iterating over an array of fixed size.	Waiting for a condition to be met dynamically.

10 =

```
public class NestedLoopBreak {
   public static void main(String[] args) {
      for (int i = 1; i <= 3; i++) {
         for (int j = 1; j <= 3; j++) {
            if (j == 2) {
               break; // Exits the inner loop only
            }
            System.out.println("i = " + i + ", j = " + j);
         }
      }
   }
}
```

o/p= i = 1, j = 1

i = 2, j = 1

i = 3, j = 1

11 =

Benefits of Arrays:

1. **Efficient Data Storage:**
    - Arrays store multiple elements in contiguous memory locations.
    - Example:

java

Copy code

```
int[] numbers = {10, 20, 30};
System.out.println(numbers[1]); // Output: 20
```

2. **Ease of Access:**
    - Elements can be accessed using indices.
    - Example: numbers[2] gives the 3rd element.
3. **Structured Management:**
    - Arrays organize data of the same type, which improves readability and maintenance.
    - Example:

java

Copy code

```
int[] marks = {85, 90, 78};
for (int mark : marks) {
   System.out.println(mark);
}
```

4. **Reduces Redundancy:**
    - Eliminates the need to create multiple variables for similar data.

12 =

Single-Dimensional Arrays:

- **Memory Layout:**
    - Contiguous memory blocks are allocated.
- **Access:**
    - Faster as there is only one level of indexing.
- **Example:**

java

Copy code

```
int[] numbers = {1, 2, 3};

System.out.println(numbers[2]); // Output: 3
```

Multi-Dimensional Arrays:

- **Memory Layout:**
    - Memory is allocated for each row, and rows may or may not be contiguous (in Java, it is array of arrays).
- **Access:**
    - Slightly slower due to additional indexing overhead.
- **Example:**

java

Copy code

```
int[][] matrix = {
  {1, 2},
  {3, 4}
};

System.out.println(matrix[1][1]); // Output: 4
```

Key Difference:

- Single-dimensional arrays are simpler and faster to access.
- Multi-dimensional arrays are suitable for tabular data but consume more memory due to extra references.

13 =

Problem:

- If an array is declared but not initialized, attempting to use it results in a NullPointerException.

```
public class UninitializedArray {
  public static void main(String[] args) {
    int[] numbers = null; // Declared but not initialized
    try {
```

```
            System.out.println(numbers[0]); // Accessing causes NullPointerException
        } catch (NullPointerException e) {
            System.out.println("Array is not initialized.");
        }
    }
}
```

o/p = Array is not initialized.

Solution = always initialized

```
int[] numbers = new int[5]; // Initializes an array with default values (0 for int)
System.out.println(numbers[0]); // Output: 0
```

14.	Create a method that accepts an array and returns the maximum value using a for loop.
15.	Write a Java program that finds the average of numbers stored in an integer array.
16.	Write a Java program that sums the elements of a 2D array.

14 =

```
public class MaxValueInArray {
    public static int findMax(int[] array) {
        int max = array[0]; // Assume the first element is the largest
        for (int i = 1; i < array.length; i++) {
            if (array[i] > max) {
                max = array[i]; // Update max if a larger value is found
            }
        }
        return max;
    }
    public static void main(String[] args) {
        int[] numbers = {3, 7, 2, 9, 5};
        System.out.println("Maximum value in the array: " + findMax(numbers));
    }
}
```

o/p= Maximum value in the array: 9

15 =

```
public class ArrayAverage {
    public static void main(String[] args) {
        int[] numbers = {10, 20, 30, 40, 50};
        int sum = 0;

        for (int number : numbers) {
            sum += number; // Add each element to the sum
        }

        double average = (double) sum / numbers.length; // Calculate average
        System.out.println("Average of the numbers: " + average);
    }
}
```

o/p=

Average of the numbers: 30.0

16 =

```
public class SumOf2DArray {
    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };
        int sum = 0;

        for (int i = 0; i < matrix.length; i++) {
            for (int j = 0; j < matrix[i].length; j++) {
```

```
            sum += matrix[i][j]; // Add each element to the sum
        }
    }

    System.out.println("Sum of all elements in the 2D array: " + sum);
  }
}
```

o/p=

Sum of all elements in the 2D array: 45

17.	Demonstrate how to find the minimum and maximum values in a given array.
18.	Discuss the benefits and drawbacks of using static arrays versus dynamic arrays in Java.
19.	Design a Java program that merges two sorted arrays into a single sorted array.

17 =

```
public class MinMaxInArray {
  public static void main(String[] args) {
    int[] numbers = {8, 3, 1, 9, 7, 4, 2};

    int min = numbers[0]; // Assume first element is minimum
    int max = numbers[0]; // Assume first element is maximum

    for (int number : numbers) {
      if (number < min) {
        min = number; // Update min if a smaller value is found
      }
      if (number > max) {
        max = number; // Update max if a larger value is found
      }
    }
```

```
        System.out.println("Minimum value: " + min);
        System.out.println("Maximum value: " + max);
    }
}
```

o/p=

Minimum value: 1

Maximum value: 9

18 =

Static Arrays

Benefits:

1. **Predictability:** Fixed size ensures memory allocation is straightforward.
2. **Simplicity:** Easy to use and requires no additional libraries.

Drawbacks:

1. **Inflexibility:** Size must be declared upfront and cannot be changed during runtime.
2. **Memory Waste:** Unused slots in an array lead to wasted memory.

Dynamic Arrays (e.g., ArrayList)

Benefits:

1. **Resizability:** Can grow or shrink as needed during runtime.
2. **Convenience:** Provides built-in methods for common operations like adding, removing, or sorting elements.

Drawbacks:

1. **Overhead:** Slightly slower due to internal resizing and dynamic allocation.
2. **Dependency:** Requires Java Collections framework.

19 =

```
import java.util.Arrays;

public class MergeSortedArrays {
    public static void main(String[] args) {
        int[] array1 = {1, 3, 5};
```

```
        int[] array2 = {2, 4, 6};

        int[] mergedArray = new int[array1.length + array2.length];

        int i = 0, j = 0, k = 0;

        // Merge arrays
        while (i < array1.length && j < array2.length) {
            if (array1[i] < array2[j]) {
                mergedArray[k++] = array1[i++];
            } else {
                mergedArray[k++] = array2[j++];
            }
        }

        // Add remaining elements
        while (i < array1.length) {
            mergedArray[k++] = array1[i++];
        }
        while (j < array2.length) {
            mergedArray[k++] = array2[j++];
        }

        System.out.println("Merged Sorted Array: " + Arrays.toString(mergedArray));
    }
}
```

o/p =

Merged Sorted Array: [1, 2, 3, 4, 5, 6]

20.	Write a Java program to reverse the array
21.	Find the Second largest element in Java

22.	Find the first even number in a list and breaks the loop when it finds.
23.	Prints all odd numbers from 1 to 20, using continue to skip even numbers.
24.	Prompts the user to enter numbers until they enter a negative number.

20 =

```
import java.util.Arrays;

public class ReverseArray {
    public static void main(String[] args) {
        int[] array = {10, 20, 30, 40, 50};
        System.out.println("Original Array: " + Arrays.toString(array));

        for (int i = 0, j = array.length - 1; i < j; i++, j--) {
            int temp = array[i];
            array[i] = array[j];
            array[j] = temp;
        }

        System.out.println("Reversed Array: " + Arrays.toString(array));
    }
}
```

o/p =

```
Original Array: [10, 20, 30, 40, 50]
Reversed Array: [50, 40, 30, 20, 10]
```

21 =

```
public class SecondLargest {
    public static void main(String[] args) {
        int[] numbers = {10, 20, 4, 8, 25};
        int largest = Integer.MIN_VALUE, secondLargest = Integer.MIN_VALUE;

        for (int number : numbers) {
```

```
            if (number > largest) {
                secondLargest = largest;
                largest = number;
            } else if (number > secondLargest && number != largest) {
                secondLargest = number;
            }
        }

        System.out.println("Second Largest Element: " + secondLargest);
    }
}
```

o/p =

```
Second Largest Element: 20
```

22 =

```
public class FirstEvenNumber {
    public static void main(String[] args) {
        int[] numbers = {3, 7, 9, 12, 15};

        for (int number : numbers) {
            if (number % 2 == 0) {
                System.out.println("First Even Number: " + number);
                break; // Exit the loop after finding the first even number
            }
        }
    }
}
```

o/p =

```
First Even Number: 12
```

23 =

```
public class PrintOddNumbers {
    public static void main(String[] args) {
        System.out.println("Odd Numbers from 1 to 20:");

        for (int i = 1; i <= 20; i++) {
            if (i % 2 == 0) {
                continue; // Skip even numbers
            }
            System.out.println(i);
        }
    }
}
```

o/p =

```
Odd Numbers from 1 to 20:
1
3
5
7
9
11
13
15
17
19
```

24=

```
import java.util.Scanner;

public class PromptUntilNegative {
    public static void main(String[] args) {
```

```
        Scanner scanner = new Scanner(System.in);
        int number;

        do {
            System.out.print("Enter a number (negative to quit): ");
            number = scanner.nextInt();
        } while (number >= 0);

        System.out.println("You entered a negative number. Program terminated.");
        scanner.close();
    }
}
```

o/p =

```
Enter a number (negative to quit): 10
Enter a number (negative to quit): 5
Enter a number (negative to quit): -1
You entered a negative number. Program terminated.
```

25.	Prints a multiplication table but skips the multiplication by 5.
26.	Program counts from 1 to 10 but breaks when it reaches 6.
27.	Program prints numbers from 1 to 10 but skips the number 5.
28.	Develop a program that checks whether a given number is prime or not. Use a for loop to test divisibility. If the number is found to be divisible by any number other than 1 and itself, it is not prime.

25 =

```
public class MultiplicationTableSkip5 {
    public static void main(String[] args) {
        int number = 7; // Multiplication table for 7
        System.out.println("Multiplication Table for " + number + " (skipping multiples of 5):");
```

```
        for (int i = 1; i <= 10; i++) {
            if (i == 5) {
                continue; // Skip multiplication by 5
            }
            System.out.println(number + " x " + i + " = " + (number * i));
        }
    }
}
```

o/p=

```
Multiplication Table for 7 (skipping multiples of 5):
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
```

26=

```
public class BreakAt6 {
    public static void main(String[] args) {
        System.out.println("Counting from 1 to 10, but breaking at 6:");
      for (int i = 1; i <= 10; i++) {
            if (i == 6) {
                break; // Exit loop when reaching 6           }
            System.out.println(i);
        }
    }
}
```

o/p = Counting from 1 to 10, but breaking at 6:

1

2

3

4

5

27 =

```
public class SkipNumber5 {
    public static void main(String[] args) {
        System.out.println("Printing numbers from 1 to 10, skipping 5:");

        for (int i = 1; i <= 10; i++) {
            if (i == 5) {
                continue; // Skip the number 5
            }
            System.out.println(i);
        }
    }
}
```

o/p =

Printing numbers from 1 to 10, skipping 5:

1

2

3

4

6

7

8

9

10

28 =

```
import java.util.Scanner;

public class PrimeNumberCheck {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number to check if it's prime: ");
        int number = scanner.nextInt();

        boolean isPrime = true;

        if (number <= 1) {
            isPrime = false; // Numbers less than or equal to 1 are not prime
        } else {
            for (int i = 2; i <= number / 2; i++) {
                if (number % i == 0) {
                    isPrime = false; // Found a divisor, not a prime number
                    break;
                }
            }
        }

        if (isPrime) {
            System.out.println(number + " is a prime number.");
        } else {
            System.out.println(number + " is not a prime number.");
        }

        scanner.close();
    }
}
```

o/p=

Enter a number to check if it's prime: 7

7 is a prime number.

29.	Create a program that reverses the digits of a given integer. Use a while loop to extract each digit and build the reversed number.

30.	Write a program that prints the multiplication table for a given number. The user should input the number and the range (e.g., up to 10 or 20). Use a for loop to generate the table.
31.	Write a program that counts the number of vowels and consonants in a given string. Use a for loop to iterate through the string and keep track of the counts.
32.	Print the Pattern as per given etc 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

29 =

```
import java.util.Scanner;


public class ReverseDigits {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter an integer to reverse its digits: ");
        int number = scanner.nextInt();
        int reversed = 0;


        while (number != 0) {
            int digit = number % 10; // Extract last digit
            reversed = reversed * 10 + digit; // Build the reversed number
            number /= 10; // Remove the last digit
        }
```

```
        System.out.println("Reversed Number: " + reversed);
        scanner.close();
    }
}
```

o/p=

```
Enter an integer to reverse its digits: 12345
Reversed Number: 54321
```

30 =

```
import java.util.Scanner;

public class MultiplicationTable {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number for the multiplication table: ");
        int number = scanner.nextInt();

        System.out.print("Enter the range for the table: ");
        int range = scanner.nextInt();

        System.out.println("Multiplication Table for " + number + ":");
        for (int i = 1; i <= range; i++) {
            System.out.println(number + " x " + i + " = " + (number * i));
        }

        scanner.close();
    }
}
```

o/p=

Enter the number for the multiplication table: 7

Enter the range for the table: 10

Multiplication Table for 7:

7 x 1 = 7

7 x 2 = 14

7 x 3 = 21

7 x 4 = 28

7 x 5 = 35

7 x 6 = 42

7 x 7 = 49

7 x 8 = 56

7 x 9 = 63

7 x 10 = 70

31 =

```
import java.util.Scanner;

public class VowelConsonantCount {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a string: ");
        String input = scanner.nextLine().toLowerCase();

        int vowels = 0, consonants = 0;

        for (char c : input.toCharArray()) {
            if (c >= 'a' && c <= 'z') { // Check if the character is a letter
                if ("aeiou".indexOf(c) != -1) {
                    vowels++; // Count vowels
```

```
            } else {
                consonants++; // Count consonants
            }
        }
    }

    System.out.println("Number of vowels: " + vowels);
    System.out.println("Number of consonants: " + consonants);
    scanner.close();
  }
}
```

o/p =

```
Enter a string: Hello World
Number of vowels: 3
Number of consonants: 7
```

32 =

```
public class PatternPrinting {
  public static void main(String[] args) {
    for (int i = 5; i >= 1; i--) {
      for (int j = 1; j <= i; j++) {
        System.out.print("1 ");
      }
      System.out.println();
    }
  }
}
```

33.	As part of a feedback collection system, you want to gather customer ratings for a product. Design a program that prompts customers to rate the product from 1 to 5. Use a labeled while loop to continue collecting ratings until a customer enters 0. After collecting all ratings, compute and display the average rating and the number of ratings received.
34.	You are tasked with developing a program that tracks a user's monthly expenses. The program should repeatedly ask the user to input their expenses for different categories (like food, transportation, etc.) until they type "done". After the user is finished, display the total expenses for the month
35.	Develop a password validation system that prompts users to create a password. The program should check if the password meets certain criteria (length, special characters, etc.). If it doesn't meet the criteria, it should continue prompting the user until a valid password is entered.

33 =

```
import java.util.Scanner;


public class FeedbackCollection {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int totalRatings = 0;
        int ratingCount = 0;


        System.out.println("Rate the product (1-5). Enter 0 to finish:");


        while (true) {
            System.out.print("Enter your rating: ");
            int rating = scanner.nextInt();


            if (rating == 0) {
                break; // Exit loop if customer enters 0
            }


            if (rating >= 1 && rating <= 5) {
```

```
            totalRatings += rating;
            ratingCount++;
        } else {
            System.out.println("Invalid rating! Please enter a number between 1 and 5.");
        }
    }

    if (ratingCount > 0) {
        double averageRating = (double) totalRatings / ratingCount;
        System.out.println("Total ratings received: " + ratingCount);
        System.out.println("Average rating: " + averageRating);
    } else {
        System.out.println("No ratings received.");
    }

    scanner.close();
  }
}
```

o/p =

```
Rate the product (1-5). Enter 0 to finish:
Enter your rating: 4
Enter your rating: 5
Enter your rating: 3
Enter your rating: 0
Total ratings received: 3
Average rating: 4.0
```

34 =

```
import java.util.Scanner;
```

```
public class ExpenseTracker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        double totalExpenses = 0;

        System.out.println("Enter your monthly expenses by category (type 'done' to finish):");

        while (true) {
            System.out.print("Enter category expense: ");
            String input = scanner.nextLine();

            if (input.equalsIgnoreCase("done")) {
                break; // Exit loop if user types "done"
            }

            try {
                double expense = Double.parseDouble(input);
                if (expense < 0) {
                    System.out.println("Expense cannot be negative. Try again.");
                } else {
                    totalExpenses += expense;
                }
            } catch (NumberFormatException e) {
                System.out.println("Invalid input. Please enter a valid number.");
            }
        }

        System.out.println("Total expenses for the month: $" + totalExpenses);
        scanner.close();
    }
}
```

o/p=

```
Enter your monthly expenses by category (type 'done' to finish):
Enter category expense: 200
Enter category expense: 150.75
Enter category expense: done
Total expenses for the month: $350.75
```

35 =

```
import java.util.Scanner;

public class PasswordValidation {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.print("Create a password: ");
            String password = scanner.nextLine();

            if (isValidPassword(password)) {
                System.out.println("Password created successfully!");
                break; // Exit loop when a valid password is entered
            } else {
                System.out.println("Invalid password. It must have at least 8 characters, " +
                        "one uppercase letter, one lowercase letter, one digit, and one special character.");
            }
        }

        scanner.close();
    }

    private static boolean isValidPassword(String password) {
```

```
        if (password.length() < 8) return false;

        boolean hasUppercase = false, hasLowercase = false, hasDigit = false, hasSpecialChar = false;

        for (char c : password.toCharArray()) {
            if (Character.isUpperCase(c)) hasUppercase = true;
            else if (Character.isLowerCase(c)) hasLowercase = true;
            else if (Character.isDigit(c)) hasDigit = true;
            else if (!Character.isLetterOrDigit(c)) hasSpecialChar = true;
        }

        return hasUppercase && hasLowercase && hasDigit && hasSpecialChar;
    }
}
```

o/p =

Create a password: pass123

Invalid password. It must have at least 8 characters, one uppercase letter, one lowercase letter, one digit, and one special character.

Create a password: Password@1

Password created successfully!

36.	Create a fitness app that allows users to log their daily steps. The user should be prompted to enter their steps for each day of the week. Use a loop to collect this data, and at the end of the week, calculate and display the total steps taken and the average steps per day.
37.	Develop a temperature conversion tool that allows users to convert temperatures between Celsius and Fahrenheit. Use a loop to continue asking for temperature values until the user chooses to exit. After each conversion, display the result and prompt the user again.
38.	Implement a simple banking system where users can deposit and withdraw money. Use a loop to allow the user to perform transactions until they choose to exit. After exiting, display the final account balance and transaction history.

39.	Create a program that allows a teacher to input grades for students in a class. The program should continue to prompt for grades until the teacher enters -1 to stop. After all grades have been entered, calculate and display the average grade, the highest grade, and the number of students who passed (e.g., scored above a certain threshold). share code

36 =

```
import java.util.Scanner;


public class FitnessApp {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      int totalSteps = 0;
      int days = 7;


      System.out.println("Enter your daily steps for each day of the week:");


      for (int i = 1; i <= days; i++) {
         System.out.print("Day " + i + ": ");
         int steps = scanner.nextInt();
         totalSteps += steps;
      }


      double averageSteps = (double) totalSteps / days;
      System.out.println("Total steps for the week: " + totalSteps);
      System.out.println("Average steps per day: " + averageSteps);


      scanner.close();
   }
}
```

o/p =

Enter your daily steps for each day of the week:

Day 1: 5000

Day 2: 7000

Day 3: 8000

Day 4: 6000

Day 5: 9000

Day 6: 10000

Day 7: 4000

Total steps for the week: 49000

Average steps per day: 7000.0

37 =

```
import java.util.Scanner;

public class TemperatureConverter {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("\nTemperature Conversion Tool");
            System.out.println("1. Convert Celsius to Fahrenheit");
            System.out.println("2. Convert Fahrenheit to Celsius");
            System.out.println("3. Exit");
            System.out.print("Enter your choice: ");
            int choice = scanner.nextInt();

            if (choice == 3) {
                System.out.println("Exiting the tool. Goodbye!");
                break;
            }
```

```
            System.out.print("Enter the temperature: ");
            double temperature = scanner.nextDouble();
            double converted;

            if (choice == 1) {
                converted = (temperature * 9 / 5) + 32;
                System.out.println("Temperature in Fahrenheit: " + converted);
            } else if (choice == 2) {
                converted = (temperature - 32) * 5 / 9;
                System.out.println("Temperature in Celsius: " + converted);
            } else {
                System.out.println("Invalid choice. Try again.");
            }
        }

        scanner.close();
    }
}
```

o/p =

```
Temperature Conversion Tool
1. Convert Celsius to Fahrenheit
2. Convert Fahrenheit to Celsius
3. Exit
Enter your choice: 1
Enter the temperature: 25
Temperature in Fahrenheit: 77.0

Enter your choice: 3
Exiting the tool. Goodbye!
```

38 =

```
import java.util.Scanner;

public class BankingSystem {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        double balance = 0.0;
        String transactionHistory = "";

        while (true) {
            System.out.println("\nSimple Banking System");
            System.out.println("1. Deposit");
            System.out.println("2. Withdraw");
            System.out.println("3. Exit");
            System.out.print("Enter your choice: ");
            int choice = scanner.nextInt();

            if (choice == 3) {
                System.out.println("Exiting. Final balance: $" + balance);
                System.out.println("Transaction History:\n" + transactionHistory);
                break;
            }

            System.out.print("Enter the amount: ");
            double amount = scanner.nextDouble();

            if (choice == 1) {
                balance += amount;
                transactionHistory += "Deposited: $" + amount + "\n";
                System.out.println("Deposited $" + amount + ". New balance: $" + balance);
            } else if (choice == 2) {
```

```
            if (amount > balance) {
                System.out.println("Insufficient funds. Withdrawal failed.");
            } else {
                balance -= amount;
                transactionHistory += "Withdrew: $" + amount + "\n";
                System.out.println("Withdrew $" + amount + ". New balance: $" + balance);
            }
        } else {
            System.out.println("Invalid choice. Try again.");
        }
    }

    scanner.close();
  }
}
```

o/p =Simple Banking System

1. Deposit

2. Withdraw

3. Exit

Enter your choice: 1

Enter the amount: 500

Deposited $500. New balance: $500.0

Enter your choice: 2

Enter the amount: 200

Withdrew $200. New balance: $300.0

Enter your choice: 3

Exiting. Final balance: $300.0

Transaction History:

Deposited: $500.0

Withdrew: $200.0

39 =

```
import java.util.Scanner;

public class GradeManagementSystem {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int gradeCount = 0, totalGrades = 0, highestGrade = Integer.MIN_VALUE, passCount = 0;
        final int PASS_THRESHOLD = 50;

        System.out.println("Enter grades for students (enter -1 to stop):");

        while (true) {
            System.out.print("Enter grade: ");
            int grade = scanner.nextInt();

            if (grade == -1) {
                break; // Exit loop if -1 is entered
            }

            if (grade >= 0 && grade <= 100) {
                totalGrades += grade;
                gradeCount++;
                if (grade > highestGrade) {
                    highestGrade = grade;
                }
                if (grade >= PASS_THRESHOLD) {
                    passCount++;
```

```
            }
        } else {
            System.out.println("Invalid grade. Please enter a value between 0 and 100.");
        }
    }

    if (gradeCount > 0) {
        double averageGrade = (double) totalGrades / gradeCount;
        System.out.println("Total students: " + gradeCount);
        System.out.println("Average grade: " + averageGrade);
        System.out.println("Highest grade: " + highestGrade);
        System.out.println("Number of students who passed: " + passCount);
    } else {
        System.out.println("No grades were entered.");
    }

    scanner.close();
  }
}
```

o/p=

```
Enter grades for students (enter -1 to stop):
Enter grade: 75
Enter grade: 85
Enter grade: 45
Enter grade: -1
Total students: 3
Average grade: 68.33333333333333
Highest grade: 85
Number of students who passed: 2
```

40.	Design a shopping cart application that allows users to add items to their cart. The program should ask the user for item names and prices in a loop until the user types "checkout". After checking out, display the total amount due and a list of items purchased.
41.	Write a program that calculates the total sales and commission for a group of salespeople. Prompt the user to enter sales figures for each salesperson in a loop. The loop should continue until a negative number is entered, indicating the end of input. Calculate and display the total sales and the average sales per salesperson.
42.	Write a Java program to reverse a String.
43.	How would you check if a String is a palindrome in Java?
44.	How would you identify and count the occurrences of each character in a String?

40 =

```
import java.util.ArrayList;
import java.util.Scanner;

public class ShoppingCart {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        ArrayList<String> items = new ArrayList<>();
        ArrayList<Double> prices = new ArrayList<>();
        double total = 0.0;

        System.out.println("Shopping Cart Application");
        System.out.println("Enter item names and prices. Type 'checkout' to finish.");

        while (true) {
            System.out.print("Enter item name (or 'checkout' to finish): ");
            String itemName = scanner.nextLine();

            if (itemName.equalsIgnoreCase("checkout")) {
                break;
```

```
            }

            System.out.print("Enter price for " + itemName + ": ");
            double price = scanner.nextDouble();
            scanner.nextLine(); // Consume the newline character

            items.add(itemName);
            prices.add(price);
            total += price;
        }

        System.out.println("\nItems Purchased:");
        for (int i = 0; i < items.size(); i++) {
            System.out.println(items.get(i) + " - $" + prices.get(i));
        }
        System.out.println("Total Amount Due: $" + total);

        scanner.close();
    }
}
```

o/p =

```
Shopping Cart Application
Enter item names and prices. Type 'checkout' to finish.
Enter item name (or 'checkout' to finish): Apple
Enter price for Apple: 1.5
Enter item name (or 'checkout' to finish): Banana
Enter price for Banana: 0.8
Enter item name (or 'checkout' to finish): checkout

Items Purchased:
```

Apple - $1.5

Banana - $0.8

Total Amount Due: $2.3

41 =

```
import java.util.Scanner;

public class SalesCommission {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        double totalSales = 0.0;
        int salesCount = 0;

        System.out.println("Enter sales figures for each salesperson (enter a negative number to stop):");

        while (true) {
            System.out.print("Enter sales amount: ");
            double sales = scanner.nextDouble();

            if (sales < 0) {
                break;
            }

            totalSales += sales;
            salesCount++;
        }

        if (salesCount > 0) {
            double averageSales = totalSales / salesCount;
            System.out.println("Total Sales: $" + totalSales);
```

```
            System.out.println("Average Sales per Salesperson: $" + averageSales);
        } else {
            System.out.println("No sales figures were entered.");
        }

        scanner.close();
    }
}
```

o/p =

```
Enter sales figures for each salesperson (enter a negative number to stop):
Enter sales amount: 500
Enter sales amount: 700
Enter sales amount: 300
Enter sales amount: -1
Total Sales: $1500.0
Average Sales per Salesperson: $500.0
```

42 =

```
public class ReverseString {
    public static void main(String[] args) {
        String original = "Hello, World!";
        String reversed = "";

        for (int i = original.length() - 1; i >= 0; i--) {
            reversed += original.charAt(i);
        }
        System.out.println("Original String: " + original);
        System.out.println("Reversed String: " + reversed);
    }
}
```

o/p =

Original String: Hello, World!

Reversed String: !dlroW ,olleH

43 =

```
public class PalindromeCheck {
    public static void main(String[] args) {
        String original = "radar";
        String reversed = "";

        for (int i = original.length() - 1; i >= 0; i--) {
            reversed += original.charAt(i);
        }

        if (original.equalsIgnoreCase(reversed)) {
            System.out.println(original + " is a palindrome.");
        } else {
            System.out.println(original + " is not a palindrome.");
        }
    }
}
```

o/p =

radar is a palindrome.

44 =

```
import java.util.HashMap;

public class CharacterCount {
    public static void main(String[] args) {
        String input = "hello world";
```

```
        HashMap<Character, Integer> charCount = new HashMap<>();

        for (char c : input.toCharArray()) {
            if (c != ' ') { // Ignore spaces
                charCount.put(c, charCount.getOrDefault(c, 0) + 1);
            }
        }

        System.out.println("Character occurrences:");
        for (char c : charCount.keySet()) {
            System.out.println(c + ": " + charCount.get(c));
        }
    }
}
```

o/p=

Character occurrences:

h: 1

e: 1

l: 3

o: 2

w: 1

r: 1

d: 1.

45.	Write a Java program to reverse a given String without using the built-in reverse method. Example: Input: "Hello" Output: "olleH"
46.	Create a method that checks if a given String is a palindrome (reads the same forwards and backwards). Input: "racecar" Output: true

45 =

```
import java.util.Scanner;

public class ReverseStringManual {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a string: ");
        String input = scanner.nextLine();
        String reversed = "";

        for (int i = input.length() - 1; i >= 0; i--) {
            reversed += input.charAt(i);
        }

        System.out.println("Reversed String: " + reversed);

        scanner.close();
    }
}
```

o/p =

```
Enter a string: Hello
Reversed String: olleH
```

46 =

```
import java.util.Scanner;

public class PalindromeChecker {
    public static boolean isPalindrome(String input) {
        int start = 0;
        int end = input.length() - 1;
```

```
        while (start < end) {
            if (input.charAt(start) != input.charAt(end)) {
                return false;
            }
            start++;
            end--;
        }
        return true;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a string: ");
        String input = scanner.nextLine();

        boolean result = isPalindrome(input);
        System.out.println("Is the string \"" + input + "\" a palindrome? " + result);

        scanner.close();
    }
}
```

o/p =

Enter a string: racecar

Is the string "racecar" a palindrome? True

47.	Write a program that counts the number of vowels and consonants in a given String. Input: "Hello World" Output: Vowels: 3, Consonants: 7
48.	Implement a method that capitalizes the first letter of each word in a given String. Input: "hello world" Output: "Hello World"

47 =

```
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();

        int vowels = 0, consonants = 0;

        for (char c : input.toCharArray()) {
            if (Character.isLetter(c)) {
                if ("aeiou".indexOf(c) != -1) {
                    vowels++;
                } else {
                    consonants++;
                }
            }
        }
        System.out.println("Vowels: " + vowels);
        System.out.println("Consonants: " + consonants);
        scanner.close();
    }
}
```

o/p =

```
Enter a string: Hello World
Vowels: 3
Consonants: 7
```

48 =

```
import java.util.Scanner;

public class CapitalizeWords {
    public static String capitalize(String input) {
        String[] words = input.split("\\s+");
        StringBuilder result = new StringBuilder();

        for (String word : words) {
            if (!word.isEmpty()) {
                result.append(Character.toUpperCase(word.charAt(0)))
                    .append(word.substring(1).toLowerCase())
                    .append(" ");
            }
        }

        return result.toString().trim();
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a string: ");
        String input = scanner.nextLine();

        String capitalized = capitalize(input);
        System.out.println("Capitalized String: " + capitalized);

        scanner.close();
    }
}
```

o/p=

Enter a string: hello world

Capitalized String: Hello World

49.	Implement a method that checks if two Strings are anagrams of each other (contain the same characters in a different order). Input: "listen", "silent" Output: true
50.	Write a program to remove duplicate characters from a String while maintaining the original order of characters. Input: "programming" Output: "progamin"
51.	Create a method that finds the first non-repeating character in a String. Input: "swiss" Output: 'w'
52.	Implement a method to compress a String using the counts of repeated characters. If the compressed String is not smaller than the original, return the original String. Input: "aabcccccaaa" Output: "a2b1c5a3"

49 =

```
import java.util.*;


public class StringManipulations {


   // 49. Check if two Strings are anagrams
import java.util.Arrays;


public class AnagramCheck {
   public static boolean areAnagrams(String str1, String str2) {
      if (str1.length() != str2.length()) return false;
      char[] arr1 = str1.toCharArray();
      char[] arr2 = str2.toCharArray();
      Arrays.sort(arr1);
      Arrays.sort(arr2);
      return Arrays.equals(arr1, arr2);
```

```
    }

    public static void main(String[] args) {
        System.out.println("Are 'listen' and 'silent' anagrams? " + areAnagrams("listen", "silent"));
    }
}
```

o/p =

Are 'listen' and 'silent' anagrams? True

50 =

```
import java.util.HashSet;
import java.util.Set;

public class RemoveDuplicates {
    public static String removeDuplicates(String str) {
        Set<Character> seen = new HashSet<>();
        StringBuilder result = new StringBuilder();
        for (char c : str.toCharArray()) {
            if (!seen.contains(c)) {
                seen.add(c);
                result.append(c);
            }
     }
        return result.toString();
    }
    public static void main(String[] args)
      System.out.println("After removing duplicates: " + removeDuplicates("programming"));
    }}
```

o/p =

After removing duplicates: progamin

51 =

```
import java.util.LinkedHashMap;
import java.util.Map;

public class FirstNonRepeatingCharacter {
    public static char firstNonRepeatingChar(String str) {
        Map<Character, Integer> charCount = new LinkedHashMap<>();
        for (char c : str.toCharArray()) {
            charCount.put(c, charCount.getOrDefault(c, 0) + 1);
        }
        for (Map.Entry<Character, Integer> entry : charCount.entrySet()) {
            if (entry.getValue() == 1) {
                return entry.getKey();
            }
        }
        return '\0'; // Return null character if no unique character is found
    }

    public static void main(String[] args) {
        System.out.println("First non-repeating character in 'swiss': " + firstNonRepeatingChar("swiss"));
    }
}
```

o/p =

```
First non-repeating character in 'swiss': w
```

52 =

```
public class StringCompression {
    public static String compressString(String str) {
        StringBuilder compressed = new StringBuilder();
        int count = 1;
        for (int i = 1; i < str.length(); i++) {
```

```
            if (str.charAt(i) == str.charAt(i - 1)) {
                count++;
            } else {
                compressed.append(str.charAt(i - 1)).append(count);
                count = 1;
            }
        }
        compressed.append(str.charAt(str.length() - 1)).append(count);
        return compressed.length() < str.length() ? compressed.toString() : str;
    }

    public static void main(String[] args) {
        System.out.println("Compressed String: " + compressString("aabccccccaaa"));
    }
}
```

o/p =

Compressed String: a2b1c5a3

53.	Write a Java program that appends the string " World" to an existing StringBuffer containing "Hello". Input: "Hello"

=

```
public class AppendString {
    public static void main(String[] args) {
        StringBuffer stringBuffer = new StringBuffer("Hello");
        stringBuffer.append(" World");
        System.out.println("After appending: " + stringBuffer);
    }
}
```

o/p =

After appending: Hello World

54.	Create a method that inserts the string "Beautiful " at index 6 in the StringBuffer containing "Hello World". Input: "Hello World" Output: "Hello Beautiful World"
55.	Write a Java program that reverses the contents of a StringBuffer initialized with "Java Programming". Input: "Java Programming" Output: "gnimmargorPavaJ"
56.	Create a method that deletes a substring from a StringBuffer. For example, remove "World" from "Hello World". Input: "Hello World" Output: "Hello "
57.	Write a program that initializes a StringBuffer with "Java Programming" and reverses its content. Input: "Java Programming" Output: "gnimmargorP avaJ"
58.	Create a method that deletes the substring "World" from a StringBuffer initialized with "Hello World". Print the modified StringBuffer. Input: "Hello World" Output: "Hello "

54 =

```
public class InsertString {
    public static void main(String[] args) {
        StringBuffer stringBuffer = new StringBuffer("Hello World");
        stringBuffer.insert(6, "Beautiful ");
        System.out.println("After insertion: " + stringBuffer);
    }
}
```

o/p =

After insertion: Hello Beautiful World

55 =

```
public class ReverseStringBuffer {
```

```
    public static void main(String[] args) {
        StringBuffer stringBuffer = new StringBuffer("Java Programming");
        stringBuffer.reverse();
        System.out.println("Reversed StringBuffer: " + stringBuffer);
    }
}
```

o/p=

Reversed StringBuffer: gnimmargorPavaJ

56 =

```
public class DeleteSubstring {
    public static void main(String[] args) {
        StringBuffer stringBuffer = new StringBuffer("Hello World");
        stringBuffer.delete(6, 11); // "World" starts at index 6 and ends at index 11
        System.out.println("After deletion: " + stringBuffer);
    }
}
```

o/p=

After deletion: Hello

57 =

```
public class ReverseString {
    public static void main(String[] args) {
        StringBuffer stringBuffer = new StringBuffer("Java Programming");
        stringBuffer.reverse();
        System.out.println("Reversed StringBuffer: " + stringBuffer);
    }
}
```

o/p=

Reversed StringBuffer: gnimmargorP avaJ

58 =

```
public class DeleteWorld {
   public static void main(String[] args) {
      StringBuffer stringBuffer = new StringBuffer("Hello World");
      stringBuffer.delete(6, 11); // "World" starts at index 6 and ends at index 11
      System.out.println("Modified StringBuffer: " + stringBuffer);
   }
}
```

o/p =

Modified StringBuffer: Hello

59.	Write a Java program that replaces "Java" with "Python" in a StringBuffer initialized with "I love Java programming". Input: "I love Java programming" Output: "I love Python programming"
60.	Write a program that creates a StringBuffer, checks its initial capacity, and then appends enough characters to exceed that capacity. Print the new capacity. Input: Initial capacity of StringBuffer Output: New capacity after appending characters
61.	Write a method that converts a StringBuffer to a String and returns it. Initialize a StringBuffer with "Hello World" and use your method to print the resulting string. Input: StringBuffer initialized with "Hello World" Output: "Hello World"
62.	Create a method that counts the number of vowels in a StringBuffer. Initialize it with any string and print the number of vowels. Input: "Hello World" Output: 3

59 =

```
public class ReplaceString {
   public static void main(String[] args) {
      StringBuffer stringBuffer = new StringBuffer("I love Java programming");
      int start = stringBuffer.indexOf("Java");
      int end = start + "Java".length();
      if (start != -1) {
         stringBuffer.replace(start, end, "Python");
```

```
        }
        System.out.println("After replacement: " + stringBuffer);
    }
}
```

o/p = After replacement: I love Python programming

60 =

```
public class StringBufferCapacity {
    public static void main(String[] args) {
        StringBuffer stringBuffer = new StringBuffer();
        System.out.println("Initial capacity: " + stringBuffer.capacity());

        // Append characters to exceed initial capacity
        stringBuffer.append("This is a long string that exceeds the initial capacity of the StringBuffer.");
        System.out.println("New capacity after appending: " + stringBuffer.capacity());
    }
}
```

o/p= Initial capacity: 16
New capacity after appending: 70

61 =

```
public class ConvertToString {
    public static String convertToString(StringBuffer stringBuffer) {
        return stringBuffer.toString();
    }

    public static void main(String[] args) {
        StringBuffer stringBuffer = new StringBuffer("Hello World");
        String result = convertToString(stringBuffer);
```

```
        System.out.println("Converted String: " + result);
    }
}
```

o/p= Converted String: Hello World

62 =

```
public class CountVowels {
    public static int countVowels(StringBuffer stringBuffer) {
        String vowels = "aeiouAEIOU";
        int count = 0;
        for (int i = 0; i < stringBuffer.length(); i++) {
            if (vowels.indexOf(stringBuffer.charAt(i)) != -1) {
                count++;
            }
        }
        return count;
    }

    public static void main(String[] args) {
        StringBuffer stringBuffer = new StringBuffer("Hello World");
        int vowelCount = countVowels(stringBuffer);
        System.out.println("Number of vowels: " + vowelCount);
    }
}
```

o/p =

Number of vowels: 3

63.	Write a Java program that initializes a StringBuffer with extra spaces (e.g., " Hello World ") and trims the whitespace from both ends. Input: " Hello World " Output: "Hello World"
64.	Create a method that takes two StringBuffer objects and merges them into one, separating them with a space. Print the resulting StringBuffer. Input: StringBuffer1: "Hello", StringBuffer2: "World" Output: "Hello World"

63=

```
public class TrimStringBuffer {
    public static String trimWhitespace(StringBuffer stringBuffer) {
        String trimmedString = stringBuffer.toString().trim();
        return trimmedString;
    }

    public static void main(String[] args) {
        StringBuffer stringBuffer = new StringBuffer("   Hello World   ");
        String result = trimWhitespace(stringBuffer);
        System.out.println("After trimming: \"" + result + "\"");
    }
}
```

o/p =

```
After trimming: "Hello World"
```

64 =

```
public class MergeStringBuffers {
    public static StringBuffer mergeStringBuffers(StringBuffer sb1, StringBuffer sb2) {
        StringBuffer merged = new StringBuffer();
        merged.append(sb1).append(" ").append(sb2);
        return merged;
    }
```

```
    public static void main(String[] args) {
        StringBuffer sb1 = new StringBuffer("Hello");
        StringBuffer sb2 = new StringBuffer("World");
        StringBuffer result = mergeStringBuffers(sb1, sb2);
        System.out.println("Merged StringBuffer: " + result);
    }
}
```

o/p =

Merged StringBuffer: Hello World