A black background with a black square

Description automatically generated with medium confidence

**Compiler Design Lab (CSE4001L)**

**School of Engineering and Sciences**

# Department of Computer Science and Engineering

|  |  |
| --- | --- |
|  | **Submitted By** |
| **Student Name** | **Raghav Saini** |
| **Roll No** | **200020203027** |
| **Program** | **B-Tech (CSE)** |
| **Section/Group** | **T1** |
| **Department** | **Computer Science and Engineering** |
| **Session/ Semester** | **2020-24/7thSemester** |
|  | **Submitted To S** |
| **Faculty Name** | **Dr. Alina Banerjee** |

**G.D Goenka University,Gurgaon, Haryana**

**Table of Contents**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr.no** | **Contents** | **Date** | **Signature** |
| **1.** | **Write a Java program to implement a Bubble Sort Algorithm.** |  |  |
| **2.** | **Write a Java program to implement Matrix Multiplication.** |  |  |
| **3.** | **Write a Java program to check if a character is white space or not.** |  |  |
| **4.** | **Write a program in Java to implement a DFA which accepts 01 at the beginning of the string.** |  |  |
| **5.** | **Write a program in Java to implement a DFA which accepts 001 at the end of the string.** |  |  |

# **EXPERIMENT- 1**

**AIM : Write a Java program to implement a Bubble Sort Algorithm.**

**Theory:**

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in the wrong order. This algorithm is not suitable for large data sets as its average and worst-case time complexity is quite high.

**SourceCode:**

package src;

public class BubbleSort {

static void bubbleSort(int[] arr) {

int n = arr.length;

for (int i = 0; i < n-1; i++) {

for (int j = 0; j < n-i-1; j++) {

if (arr[j] > arr[j+1]) {

int temp = arr[j];

arr[j] = arr[j+1];

arr[j+1] = temp;

}

}

}

}

public static void main(String[] args) {

int[] arr = {64, 34, 25, 12, 22, 11, 90};

System.out.println("Original array:");

for (int value : arr) {

System.out.print(value + " ");

}

System.out.println();

bubbleSort(arr);

System.out.println("\nSorted array:");

for (int value : arr) {

System.out.print(value + " ");

}

}

}

**Output:**

A black screen with white text

Description automatically generated

# **EXPERIMENT- 2**

**AIM: Write a Java program to implement Matrix Multiplication.**

**Theory:**

Matrix multiplication in C: We can add, subtract, multiply and divide 2 matrices. To do so, we are taking input from the user for row number, column number, first matrix elements and second matrix elements. Then we are performing multiplication on the matrices entered by the user.

**Source Code:**

package src;

public class MatrixMultiplication {

static int[][] multiplyMatrices(int[][] matrix1, int[][] matrix2) {

int rows1 = matrix1.length;

int cols1 = matrix1[0].length;

int cols2 = matrix2[0].length;

int[][] result = new int[rows1][cols2];

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols2; j++) {

for (int k = 0; k < cols1; k++) {

result[i][j] += matrix1[i][k] \* matrix2[k][j];

}

}

}

return result;

}

public static void main(String[] args) {

int[][] matrix1 = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

int[][] matrix2 = {{9, 8, 7}, {6, 5, 4}, {3, 2, 1}};

int[][] result = multiplyMatrices(matrix1, matrix2);

System.out.println("Matrix 1:");

printMatrix(matrix1);

System.out.println("\nMatrix 2:");

printMatrix(matrix2);

System.out.println("\nResultant Matrix:");

printMatrix(result);

}

static void printMatrix(int[][] matrix) {

for (int[] row : matrix) {

for (int value : row) {

System.out.print(value + " ");

}

System.out.println();

}

}

}

**Output:**

A black screen with white text

Description automatically generated

## **EXPERIMENT- 3**

**AIM: Write a Java program to check if a character is white space or not.**

**Source Code:**

public class CheckWhiteSpace {

public static void main(String[] args) {

char character = ' ';

if (isWhiteSpace(character)) {

System.out.println("'" + character + "' is a white space character.");

} else {

System.out.println("'" + character + "' is not a white space character.");

}

}

static boolean isWhiteSpace(char ch) {

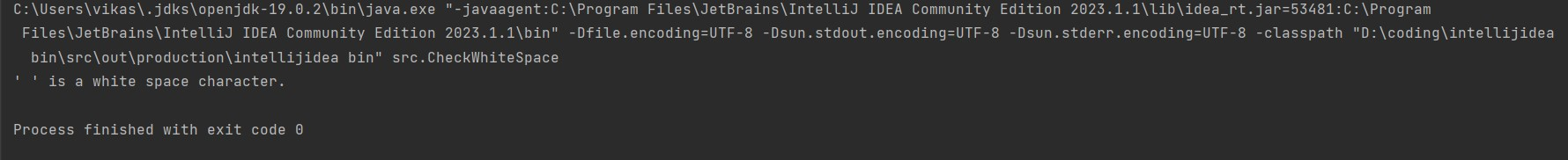
// Using Character.isWhitespace method to check if the character is a white space character

return Character.isWhitespace(ch);

}

}

**Output:**



# **EXPERIMENT- 4**

**AIM: Write a program in Java to implement a DFA which accepts 01 at the beginning of the string.**

**Source Code:**

package src;

public class StartWith01DFA {

public static void main(String[] args) {

String[] testStrings = {"01abc", "0101xyz", "1001", "012345"};

for (String str : testStrings) {

if (str.charAt(0) == '0' && str.charAt(1) == '1') {

System.out.println("Accepted: " + str);

} else {

System.out.println("Rejected: " + str);

}

}

}

}

**Output:**

A black screen with white text

Description automatically generated

# **EXPERIMENT- 5**

**AIM: Write a program in Java to implement a DFA which accepts 001 at the end of the string.**

**Source Code:**

package src;

public class EndWith001DFA {

public static void main(String[] args) {

String[] testStrings = {"abc001", "xyz1001", "001", "012345"};

for (String str : testStrings) {

if (str.charAt(str.length()-1) == '1' &&

str.charAt(str.length()-2) == '0' &&

str.charAt(str.length()-3) == '0') {

System.out.println("Accepted: " + str);

} else {

System.out.println("Rejected: " + str);

}

}

}

}

**Output:**

A black screen with white lines

Description automatically generated