| Experiment No. 6 |
| --- |
| Implement a program on 2D array & strings functions. |
| Date of Performance: |
| Date of Submission: |

## Aim: To use 2D arrays and Strings for solving given problem.

**Objective:** To use 2D array concept and strings in java to solve real world problem

**Theory:**

- An array is used to store a fixed-size sequential collection of data of the same type.
- An array can be init in two ways:
  1. Initializing at the time of declaration:
     dataType[] myArray = {value0, value1, ..., valuek};
  2. Dynamic declaration:
     dataType[] myArray = new dataType[arraySize];
     myArray[index] = value;
- Two – dimensional array is the simplest form of a multidimensional array. Data of only same data type can be stored in a 2D array.Data in a 2D Array is stored in a tabular manner which can be represented as a matrix.
- A 2D Array can be declared in 2 ways:
  1. Intializing at the time of declaration:
     dataType[][] myArray = { {valueR1C1, valueR1C2...}, {valueR2C1, valueR2C2...},..}
  2. Dynamic declaration:
     **dataType[][] myArray = new dataType[x][y];**
     **myArray[row_index][column_index] = value;**

In Java, string is basically an object that represents sequence of char values. An array of characters works same as Java string. **Java String** class provides a lot of methods to perform operations on strings such as compare(), concat(), equals(), split(), length(), replace(), compareTo(), intern(), substring() etc.

1.String literal

To make Java more memory efficient (because no new objects are created if it exists already in the string constant pool).

**Example:**

String demoString = "GeeksforGeeks";

2. Using new keyword

- String s = new String("Welcome");
- In such a case, JVM will create a new string object in normal (non-pool) heap memory and the literal "Welcome" will be placed in the string constant pool. The variable s will refer to the object in the heap (non-pool)

**Example:**

String demoString = new String ("GeeksforGeeks");

**Code:**

1. **2D Array Example:**

```
public class TwoDArray {
    public static void main(String[] args) {
        int rows = 4;
        int columns = 4;

        int[][] array = new int[rows][columns];

        int value = 1;
```

```
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < columns; j++) {
                array[i][j] = value;
                value++;
            }
        }

        System.out.println("The 2D array is: ");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < columns; j++) {
                System.out.print(array[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

**Output:-**

```
C:\Users\admin\Desktop>java TwoDArray.java"
The 2D array is:
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
```

**2. String Example:**

```
public class StringExample {
    public static void main(String[] args) {
        String s1 = "java";
        char[] ch = {'s', 't', 'r', 'i', 'n', 'g', 's'};
        String s2 = new String(ch);
        String s3 = new String("example");

        System.out.println("String s1: " + s1);
        System.out.println("String s2: " + s2);
        System.out.println("String s3: " + s3);

        int lengthS1 = strlen(s1);
        System.out.println("Length of s1: " + lengthS1);

        int comparisonResult = strcmp(s1, s2);
                System.out.println("Comparison of s1 and s2: " +
comparisonResult);

        String concatenatedString = strcat(s1, s2);
                    System.out.println("Concatenated String: " +
concatenatedString);

        String reversedS1 = strrev(s1);
        System.out.println("Reversed s1: " + reversedS1);
    }

    public static int strlen(String s) {
        return s.length();
    }
```

```
    public static int strcmp(String s1, String s2) {
        return s1.compareTo(s2);
    }

    public static String strcat(String s1, String s2) {
        return s1 + s2;
    }

    public static String strrev(String s) {
        return new StringBuilder(s).reverse().toString();
    }
}
```

```
C:\Users\admin\Desktop>java StringExample.java"
String s1: java
String s2: strings
String s3: example
Length of s1: 4
Comparison of s1 and s2: -9
Concatenated String: javastrings
Reversed s1: avaj
```

**Conclusion:**

Comment on how you have used the concept of string and 2D array.

→

2D Arrays:

- Represent and manipulate data in a matrix form.
- Use nested loops for initialization and iteration.
- Useful for applications requiring a grid or table structure.

Strings:

- Create strings from literals, character arrays, and the new keyword.
- Perform operations like length calculation, comparison, concatenation, and reversal.
- Java provides powerful methods and classes (String, StringBuilder) for handling and manipulating strings efficiently.