Experiment No. 6
Implement a program on 2D array & strings functions.
Date of Performance:05/09/24
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:

2D array:

```
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:

```
The 2D array is:
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
```

String:

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

        String s1 = "Anish";
        char[] ch = {'s', 't', 'r', 'i', 'n', 'g', 's'};
        String s2 = new String(ch);
        String s3 = new String("Naik");



        System.out.println(s1);
        System.out.println(s2);
        System.out.println(s3);



    }
}
```

Output:

Anish

strings

Naik

Conclusion:

In conclusion, the provided Java programs effectively illustrate fundamental concepts in working with 2D arrays and strings. The `TwoDarray` class demonstrates how to initialize and populate a 2D integer array, showcasing basic nested loop structures for data manipulation. The output clearly represents the array's content, reinforcing understanding of multidimensional arrays.

Similarly, the `MyString` class highlights various ways to create and display strings in Java, including the use of character arrays and string literals. Together, these examples serve as a solid foundation for understanding array manipulation and string handling, essential skills in Java programming.