K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Computer Engineering

Batch: E2	**Roll No.: 16010123325**
Experiment / assignment / tutorial No.	
Grade: AA / AB / BB / BC / CC / CD /DD	
Signature of the Staff In-charge with date	

TITLE : Multi-dimensional Arrays (Jagged Array)

AIM: Write a program which stores information about n players in a two dimensional array. The array should contain the number of rows equal to the number of players. Each row will have a number of columns equal to the number of matches played by that player which may vary from player to player. The program should display player number (index +1), runs scored in all matches and its batting average as output. (It is expected to assign columns to each row dynamically after getting value from the user.

Expected OUTCOME of Experiment:

CO1:Apply the features of object oriented programming languages. (C++ and Java)

CO2:Explore arrays, vectors, classes and objects in C++ and Java

---

Books/ Journals/ Websites referred:

1. E. Balagurusamy, “Programming with Java”, McGraw-Hill.
2. E. Balagurusamy, “Object Oriented Programming with C++”, McGraw-Hill.

Pre Lab/ Prior Concepts:

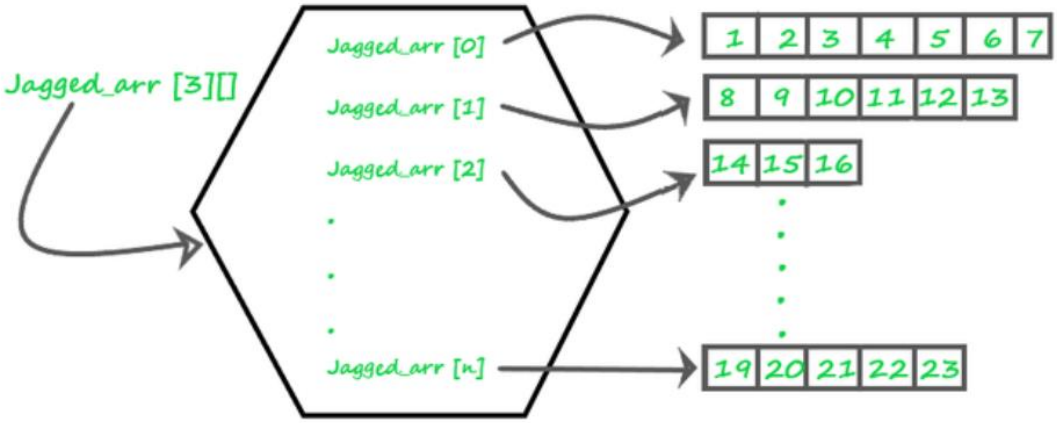
Arrays

Multi-Dimensional Array:

10 12 43 11 22
20 45 56 1 33
30 67 32 14 44
40 12 87 14 55
50 86 66 13 66

60 53 44 12 11
A multi-dimensional array is one that can hold all the values above. You set them up like this:
int[ ][ ] numbers = new int[6**][**5**];**
The first set of square brackets is for the rows and the second set of square brackets is for the columns. In the above line of code, we're telling Java to set up an array with 6 rows and 5 columns.
aryNumbers[0][0] = 10;
aryNumbers[0][1] = 12;
aryNumbers[0][2] = 43;
aryNumbers[0][3] = 11;
aryNumbers[0][4] = 22;
So the first row is row 0. The columns then go from 0 to 4, which is 5 items.
Jagged Array:

A jagged array, also known as a "ragged array," is an array of arrays where each "inner" array can have different lengths. This contrasts with a rectangular array (or a multi-dimensional array), where every inner array must have the same length. Jagged arrays are useful when dealing with data structures that naturally vary in size, such as lists of lists or matrices with different numbers of columns.



Theory

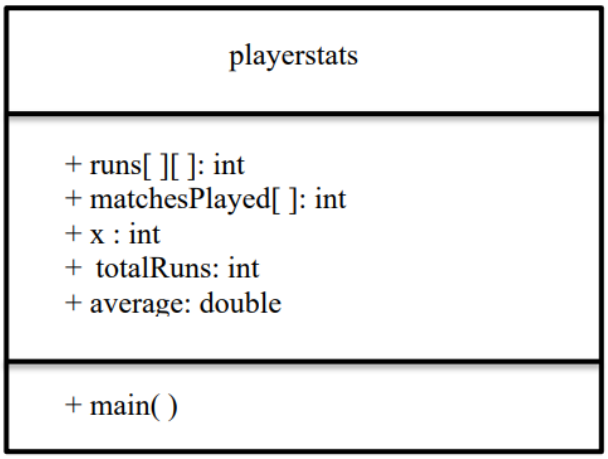
1. **Definition**: A jagged array is an array whose elements are arrays, possibly of different lengths. This means that the length of each inner array can vary.
2. **Memory Layout**: Unlike a rectangular array where memory allocation is continuous, each inner array in a jagged array is a separate array stored at different locations in memory.

3. **Usage**: Jagged arrays are often used in scenarios where the data is inherently irregular. For example, they can be useful in representing data structures like adjacency lists in graphs, where different nodes have different numbers of neighbors.
4. **Advantages**:
   - **Space Efficiency**: Only the required space is allocated for each sub-array, saving memory when dealing with irregular data.
   - **Flexibility**: Allows more flexibility in managing arrays of varying lengths.
5. **Disadvantages**:
   - **Complexity**: Increased complexity in managing and accessing elements.
   - **Performance**: Potentially lower performance due to non-contiguous memory allocation

Syntax :

// Declare a jagged array with 3 elements
int[][] jaggedArray = new int[3][];

Class Diagram:



Algorithm:

Start Program

Input Number of Players (n)

Initialize Arrays:

- runs as a 2-D array of size n
- matchesPlayed as a 1-D array of size n

For Each Player (Index i):

- Input the number of matches played and store in matchesPlayed[i]
- Initialize runs[i] with the number of matches
- For each match (Index j), input runs scored and store in runs[i][j]

Display Stats for Each Player:

- Calculate and display total runs and batting average

End Program

Implementation details:

```
import java.util.Scanner;

public class PlayerStatistics {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of players: ");
        int n = sc.nextInt();
        int runs[][] = new int[n][];
        int matchesPlayed[] = new int[n];

        for (int i = 0; i < n; i++) {
            System.out.print("Enter the number of matches played by
player " + (i + 1) + ": ");
            int x = sc.nextInt();
            matchesPlayed[i] = x;
            runs[i] = new int[x];

            for (int j = 0; j < x; j++) {
                System.out.print("Enter runs scored in match " + (j + 1)
+ " by player " + (i + 1) + ": ");
                runs[i][j] = sc.nextInt();
            }
        }

        System.out.println("\nPlayer Stats:");

        for (int i = 0; i < n; i++) {
            System.out.print("Player " + (i + 1) + " scores: ");
            int totalRuns = 0;

            for (int j = 0; j < matchesPlayed[i]; j++) {
                System.out.print(runs[i][j] + " ");
                totalRuns += runs[i][j];
            }

            double average = (double) totalRuns / matchesPlayed[i];
            System.out.printf("\nBatting Average: %.2f\n", average);
        }
    }
}
```

Output:

```
java -cp /tmp/5FbUNPhkmi/PlayerStatistics
Enter the number of players: 2
Enter the number of matches played by player 1: 2
Enter runs scored in match 1 by player 1: 809
Enter runs scored in match 2 by player 1: 722
Enter the number of matches played by player 2: 2
Enter runs scored in match 1 by player 2: 342
Enter runs scored in match 2 by player 2: 12

Player Stats:
Player 1 scores: 809 722
Batting Average: 765.50
Player 2 scores: 342 12
Batting Average: 177.00

=== Code Execution Successful ===
```

Conclusion:

The above program highlights the dynamic implementation of arrays and the application and use of jaggered arrays.

Date: _____ **Signature of faculty in-charge**

Post Lab Descriptive Questions:

Q.1 Write a program for Given an array arr[] of size N. The task is to find the sum of the contiguous subarray within a arr[] with the largest sum.

Ans:

```
import java.util.Scanner;

public class MaxSubarraySum {
    public static long maxSubarraySum(int[] arr, int n) {
        long maxi = Long.MIN_VALUE;
        long sum = 0;

        for (int i = 0; i < n; i++) {
            sum += arr[i];
            if (sum > maxi) {
                maxi = sum;
            }
            if (sum < 0) {
                sum = 0;
            }
        }
        return maxi;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the size of the array: ");
        int n = sc.nextInt();
        int[] arr = new int[n];

        System.out.println("Enter the elements of the array:");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }

        long maxSum = maxSubarraySum(arr, n);
        System.out.println("Maximum contiguous subarray sum is: " +
maxSum);
        sc.close(); // Close the scanner to prevent resource leak
    }
}
```

```
java -cp /tmp/We6jXsL3T4/MaxSubarraySum
Enter the size of the array: 5
Enter the elements of the array:
7382 382 2732 378 099
Maximum contiguous subarray sum is: 10973

=== Code Execution Successful ===
```

```
java -cp /tmp/yGkFCi6KFV/MaxSubarraySum
Enter the size of the array: 5
Enter the elements of the array:
1 2 3 4 5
Maximum contiguous subarray sum is: 15

=== Code Execution Successful ===
```

Q.2.Create a jagged array of integers. This array should consist of two 2-D arrays. First 2-D array should contain 3 rows having length of 4,3,and 2 respectively. Second 2-D array should contain 2 rows with length 3 and 4 respectively.

Ans:

```java
public class JaggedArray {
    public static void main(String[] args) {
        int[][] first = new int[3][];
        first[0] = new int[4];
        first[1] = new int[3];
        first[2] = new int[2];

        int[][] second = new int[2][];
        second[0] = new int[3];
        second[1] = new int[4];

        // Initialize the first jagged array
        for (int i = 0; i < first.length; i++) {
            for (int j = 0; j < first[i].length; j++) {
                first[i][j] = i + j;
            }
```

```
        }

        // Initialize the second jagged array
        for (int i = 0; i < second.length; i++) {
            for (int j = 0; j < second[i].length; j++) {
                second[i][j] = i + j + 1;
            }
        }

        System.out.println("First 2-D Array:");
        printJaggedArray(first);

        System.out.println("Second 2-D Array:");
        printJaggedArray(second);
    }

    public static void printJaggedArray(int[][] array) {
        for (int i = 0; i < array.length; i++) {
            for (int j = 0; j < array[i].length; j++) {
                System.out.print(array[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

```
java -cp /tmp/6NhfoBgd7x/JaggedArray
First 2-D Array:
0 1 2 3
1 2 3
2 3
Second 2-D Array:
1 2 3
2 3 4 5

=== Code Execution Successful ===
```

Q.3. Consider the following code

int number[] = new int[5];

After execution of this statement, which of the following are true?

(A) number[0] is undefined

(B) number[5] is undefined

(C) number[4] is null

(D) number[2] is 0

(E) number.length() is 5

(i) (C) & (E)

(ii) (A) & (E)

(iii) (E)

(iv) (B), (D) & (E)

Ans:

(iv) (B), (D) & (E)

Output:

The array number would look like: number=[0,0,0,0,0]