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

Batch: B2 **Roll No.:16010124107**

Experiment / assignment / tutorial No. 4

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];

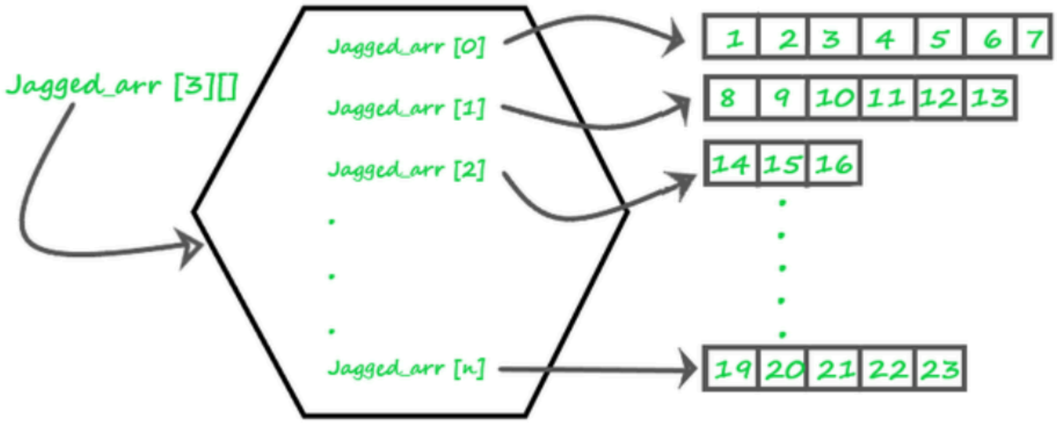
The first set of square brackets is for the rows and the second set of square brackets is for the columns. In the above line of code, we're telling Java to set up an array with 6 rows and 5 columns.

```
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

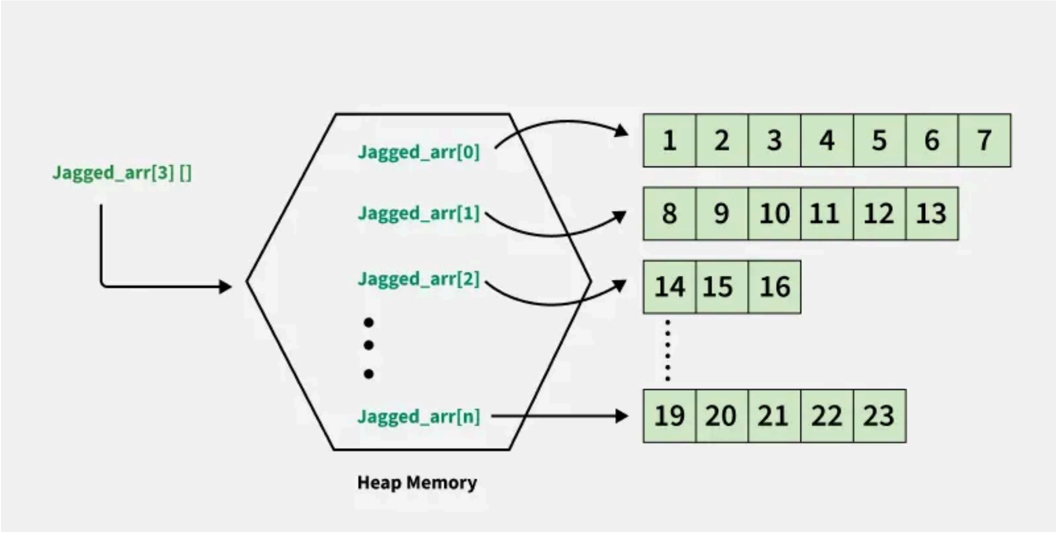
1. **Definition**: A jagged array is an array whose elements are arrays, possibly of different lengths. This means that the length of each inner array can vary.
2. **Memory Layout**: Unlike a rectangular array where memory allocation is continuous, each inner array in a jagged array is a separate array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

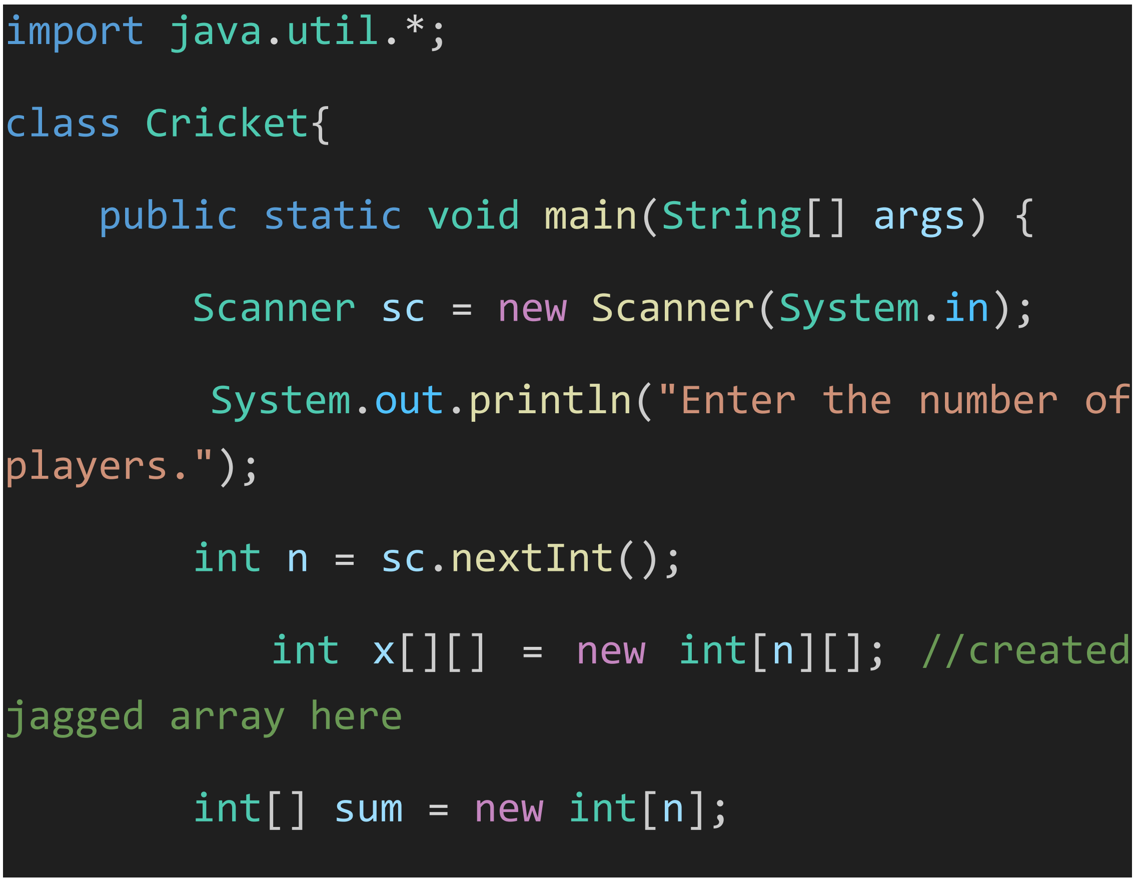
Class Diagram:



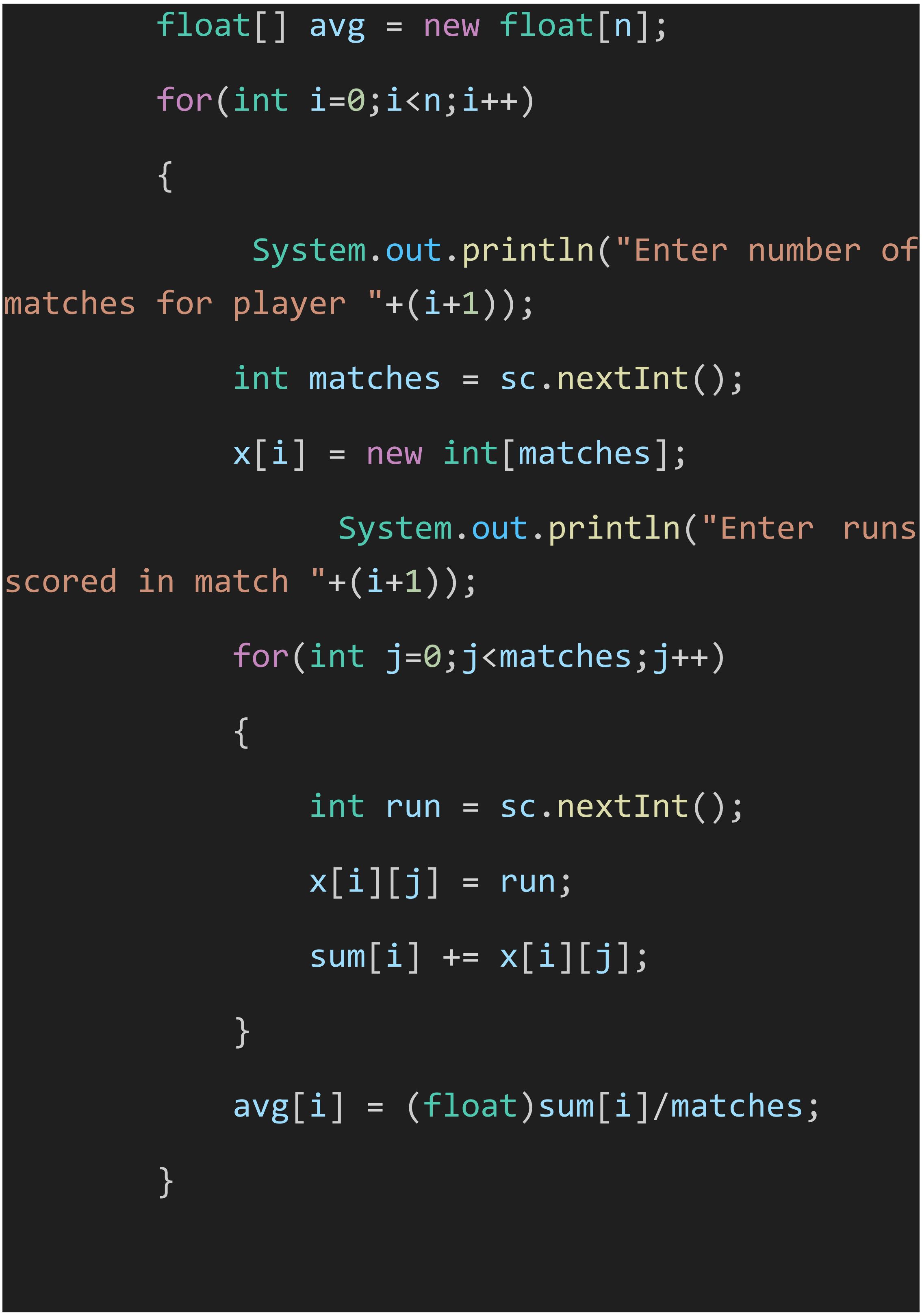
Algorithm:

1. start
2. import util package
3. create class and main method
4. initialise scanner
5. Prompt the user to enter the number of players
6. Initialise a jagged array int x[][] = new int[n][]
7. Initialise two helper arrays sum[] and avg[]
8. Run a for loop from 0 to n-1 to populate the jagged array. At each computation, keep updating the sum array and at the end, populate the avg array.
9. After taking input for all players, initialize another for loop that displays details as required.
10. Stop

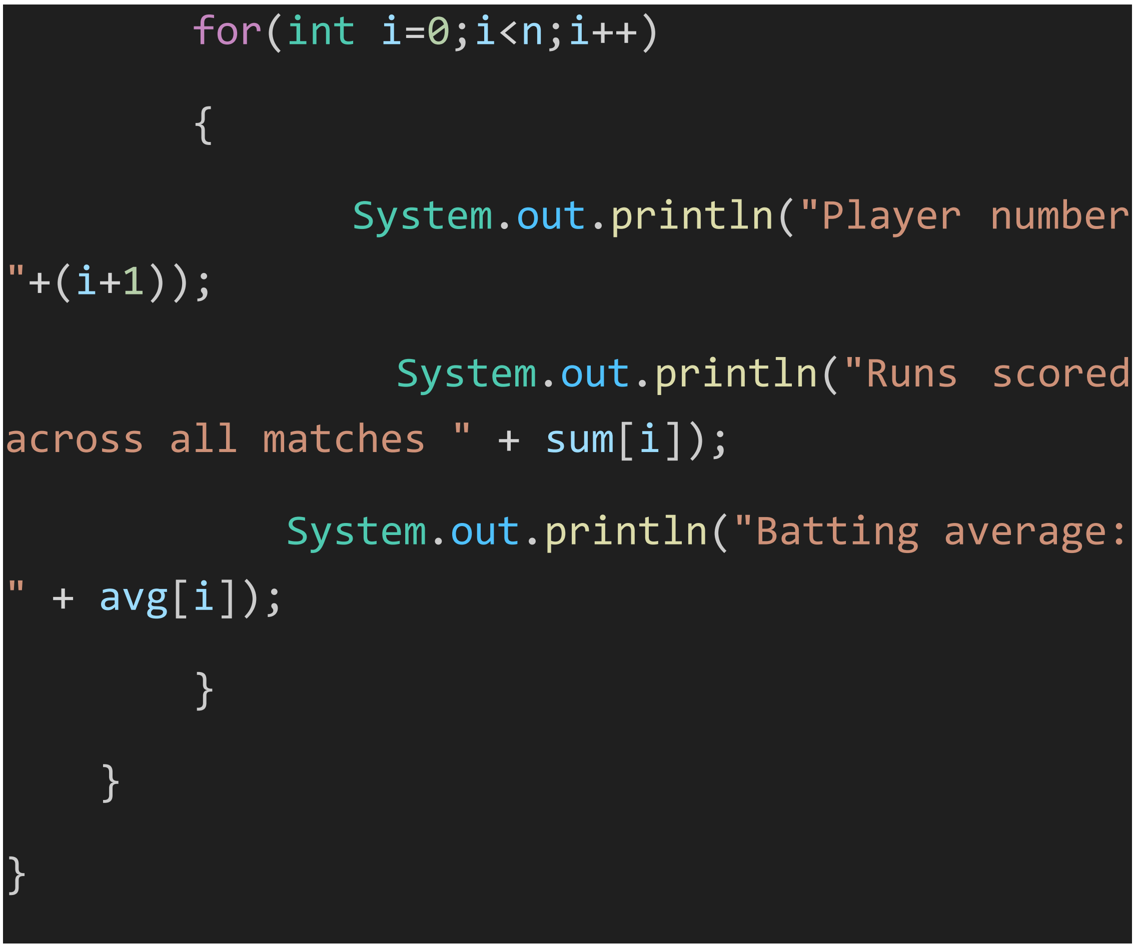
Implementation details:



```
import java.util.*;

class Cricket{

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

         System.out.println("Enter the number of
players.");

        int n = sc.nextInt();

           int x[][] = new int[n][]; //created
jagged array here

        int[] sum = new int[n];
```

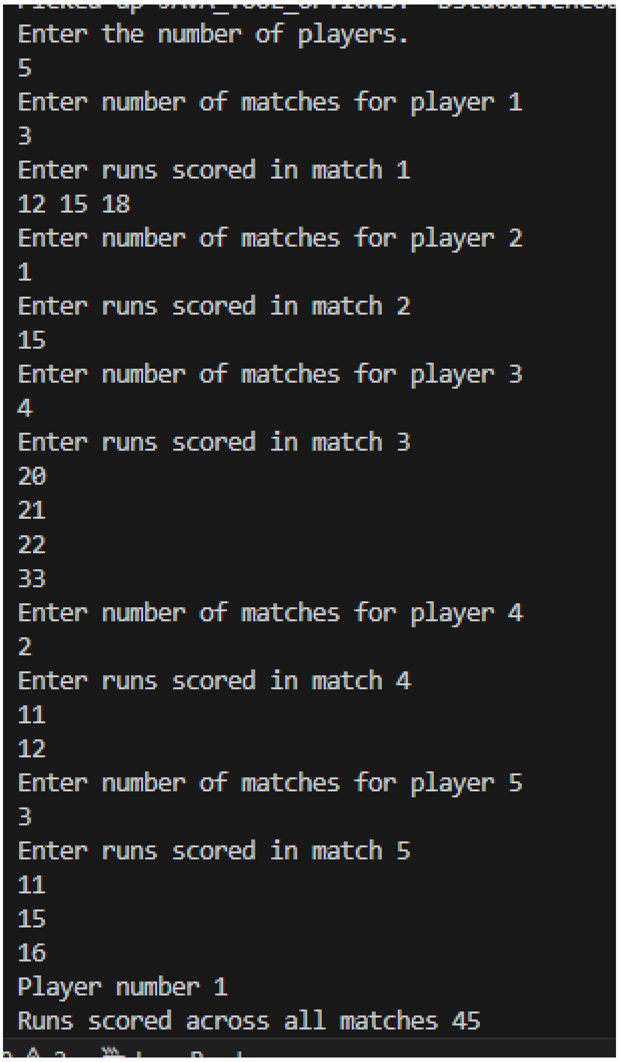


```
        float[] avg = new float[n];

        for(int i=0;i<n;i++)

        {

             System.out.println("Enter number of
matches for player "+(i+1));

            int matches = sc.nextInt();

            x[i] = new int[matches];

                 System.out.println("Enter  runs
scored in match "+(i+1));

            for(int j=0;j<matches;j++)

            {

                int run = sc.nextInt();

                x[i][j] = run;

                sum[i] += x[i][j];

            }

            avg[i] = (float)sum[i]/matches;

        }
```

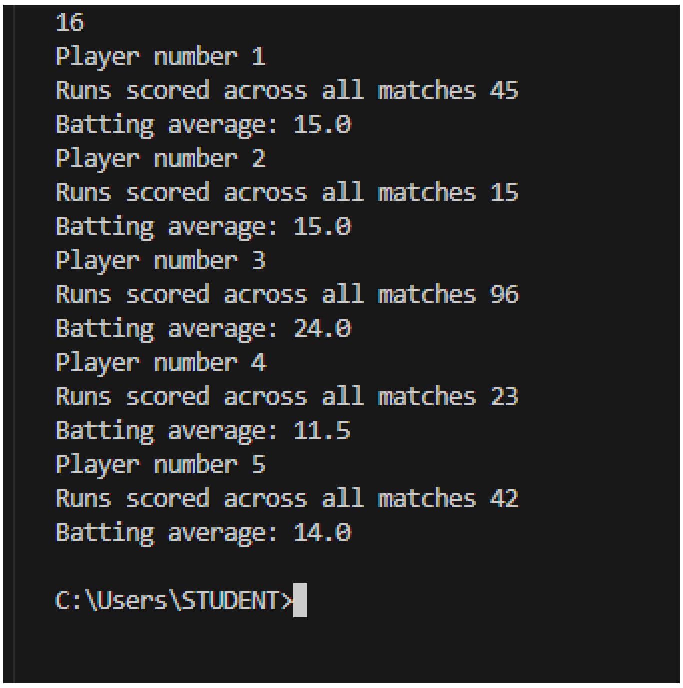


```
        for(int i=0;i<n;i++)

        {

               System.out.println("Player number
"+(i+1));

                 System.out.println("Runs scored
across all matches " + sum[i]);

            System.out.println("Batting average:
" + avg[i]);

        }

    }

}
```

Output:



```
Enter the number of players.
5
Enter number of matches for player 1
3
Enter runs scored in match 1
12 15 18
Enter number of matches for player 2
1
Enter runs scored in match 2
15
Enter number of matches for player 3
4
Enter runs scored in match 3
20
21
22
33
Enter number of matches for player 4
2
Enter runs scored in match 4
11
12
Enter number of matches for player 5
3
Enter runs scored in match 5
11
15
16
Player number 1
Runs scored across all matches 45
```



```
16
Player number 1
Runs scored across all matches 45
Batting average: 15.0
Player number 2
Runs scored across all matches 15
Batting average: 15.0
Player number 3
Runs scored across all matches 96
Batting average: 24.0
Player number 4
Runs scored across all matches 23
Batting average: 11.5
Player number 5
Runs scored across all matches 42
Batting average: 14.0

C:\Users\STUDENT>
```

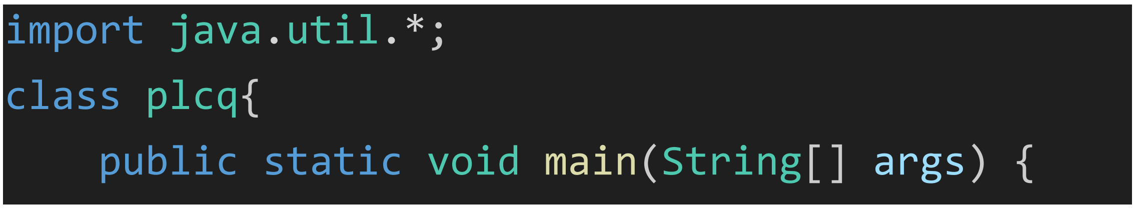
Conclusion:

Jagged arrays are two-dimensional arrays that allow for dynamic sizing of each row(or column). This is particularly useful in cases where a collection of certain objects is considered but every object may not necessarily have the same number of objects within. Using for loops smartly can help us minimise time and complexity, like we did not run additional loops for sum and avg but did it during input itself.

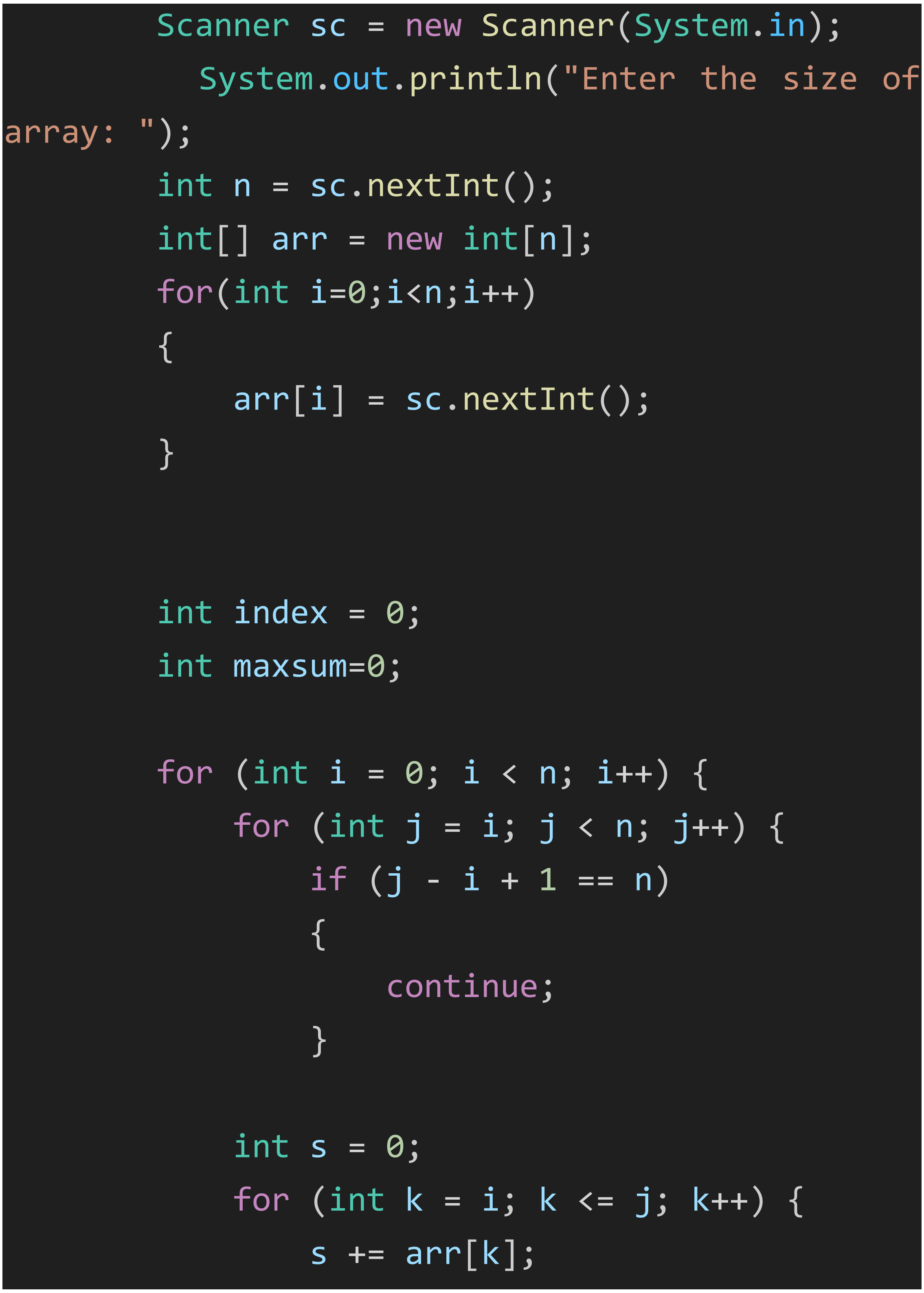
Date:25/08/25 **Signature of faculty in-charge**

Post Lab Descriptive Questions:

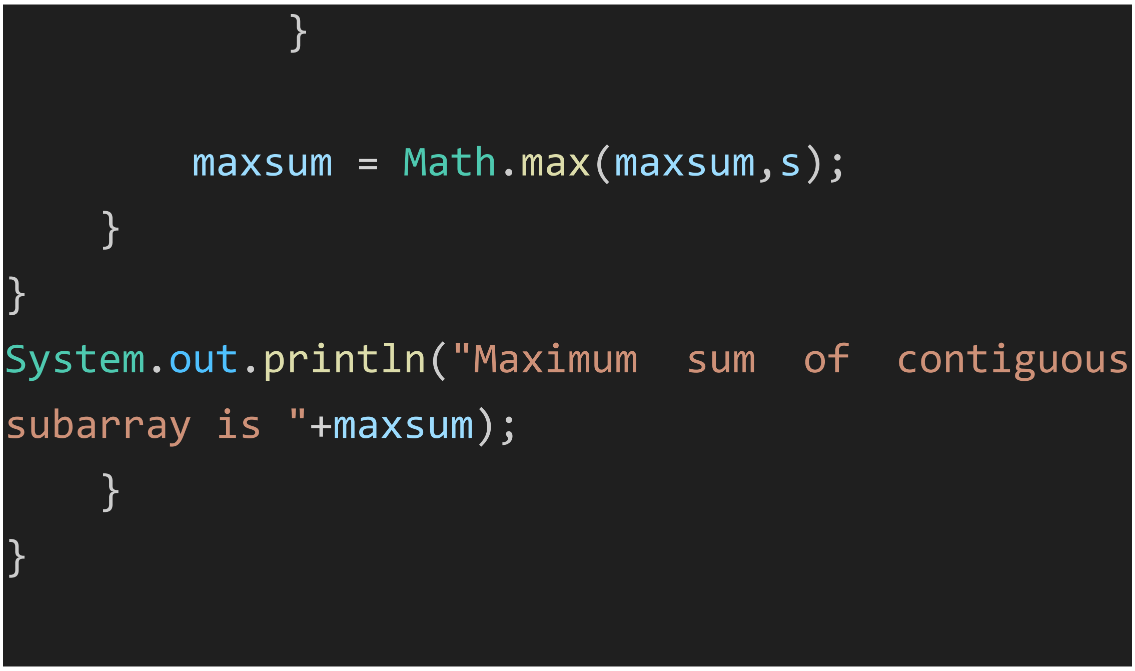
Q.1 Write a program for Given an array arr[] of size N. The task is to find the sum of the contiguous subarray within a arr[] with the largest sum.



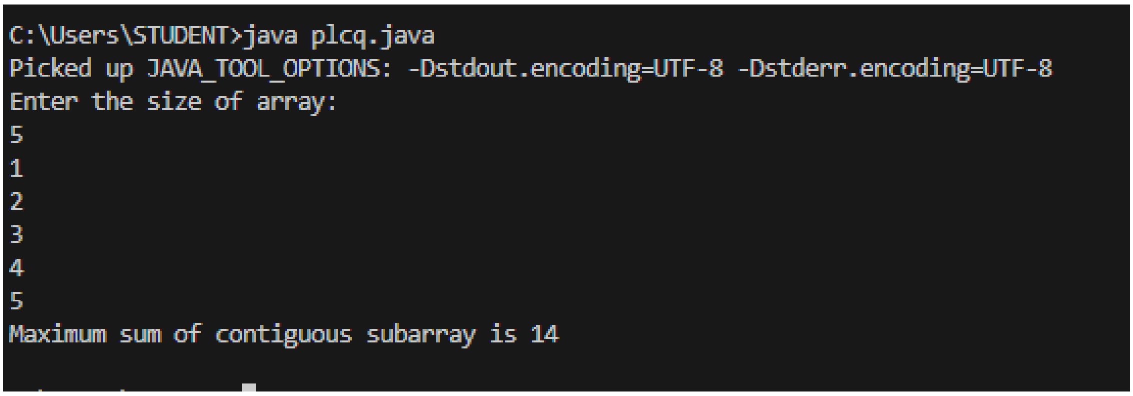
```
import java.util.*;
class plcq{
    public static void main(String[] args) {
```



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



        int index = 0;
        int maxsum=0;

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

            int s = 0;
            for (int k = i; k <= j; k++) {
                s += arr[k];
```

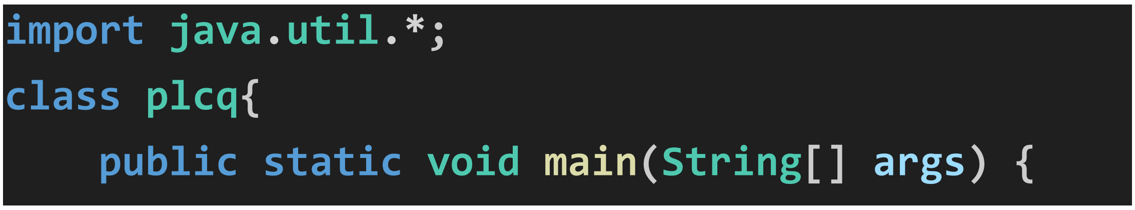


```
            }

        maxsum = Math.max(maxsum,s);
    }
}
System.out.println("Maximum sum of contiguous subarray is "+maxsum);
    }
}
```

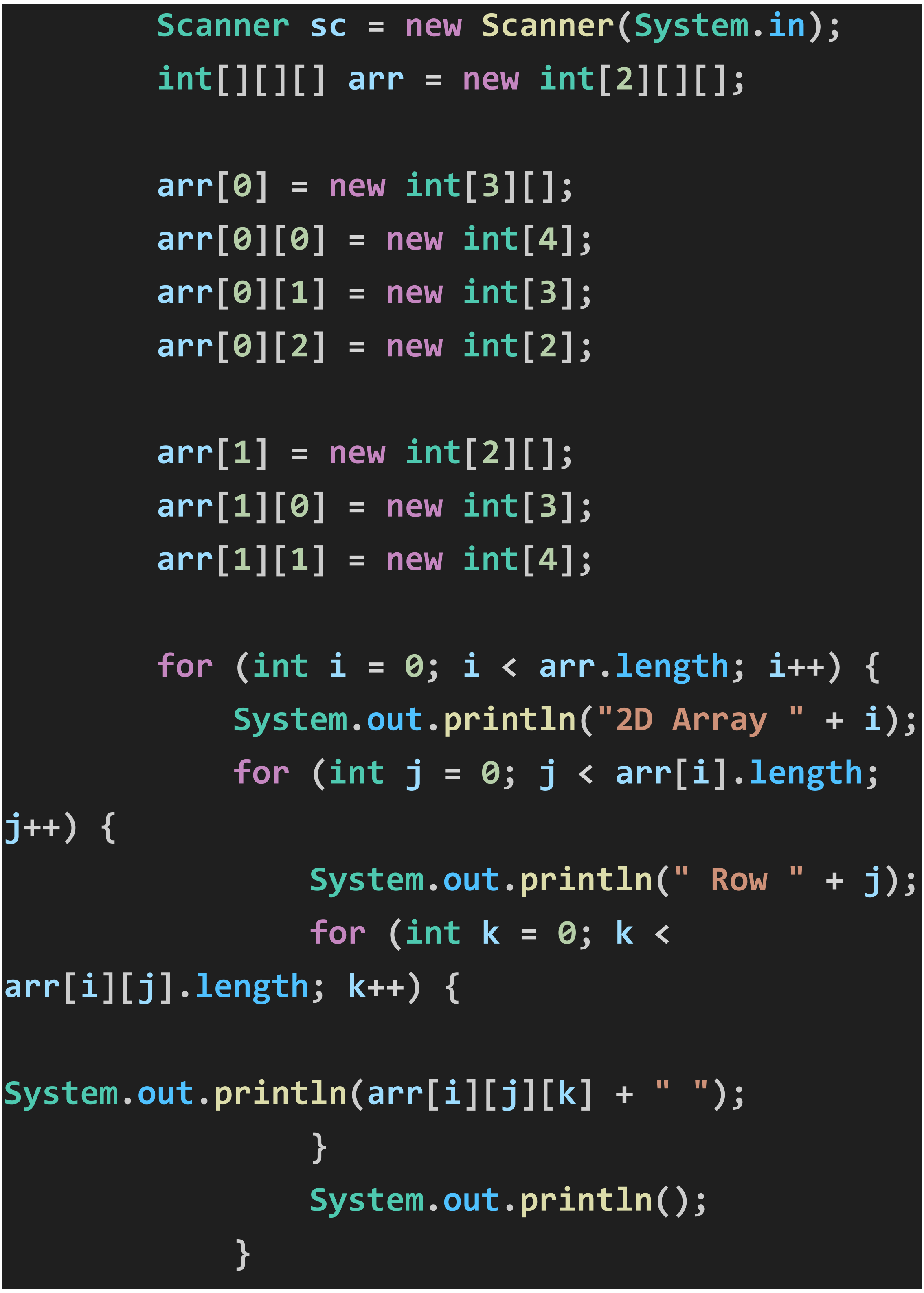


```
C:\Users\STUDENT>java plcq.java
Picked up JAVA_TOOL_OPTIONS: -Dstdout.encoding=UTF-8 -Dstderr.encoding=UTF-8
Enter the size of array:
5
1
2
3
4
5
Maximum sum of contiguous subarray is 14
```

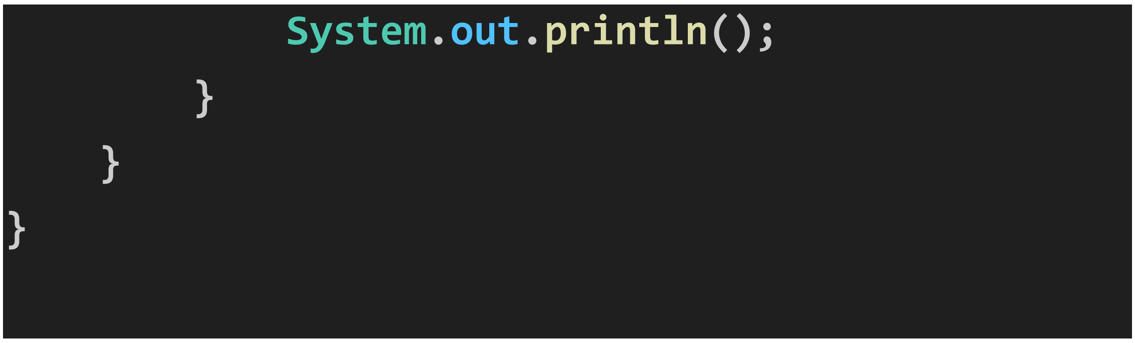
Q.2.Create a jagged array of integers. This array should consist of two 2-D arrays. First 2-D array should contain 3 rows having length of 4,3,and 2 respectively. Second 2-D array should contain 2 rows with length 3 and 4 respectively.



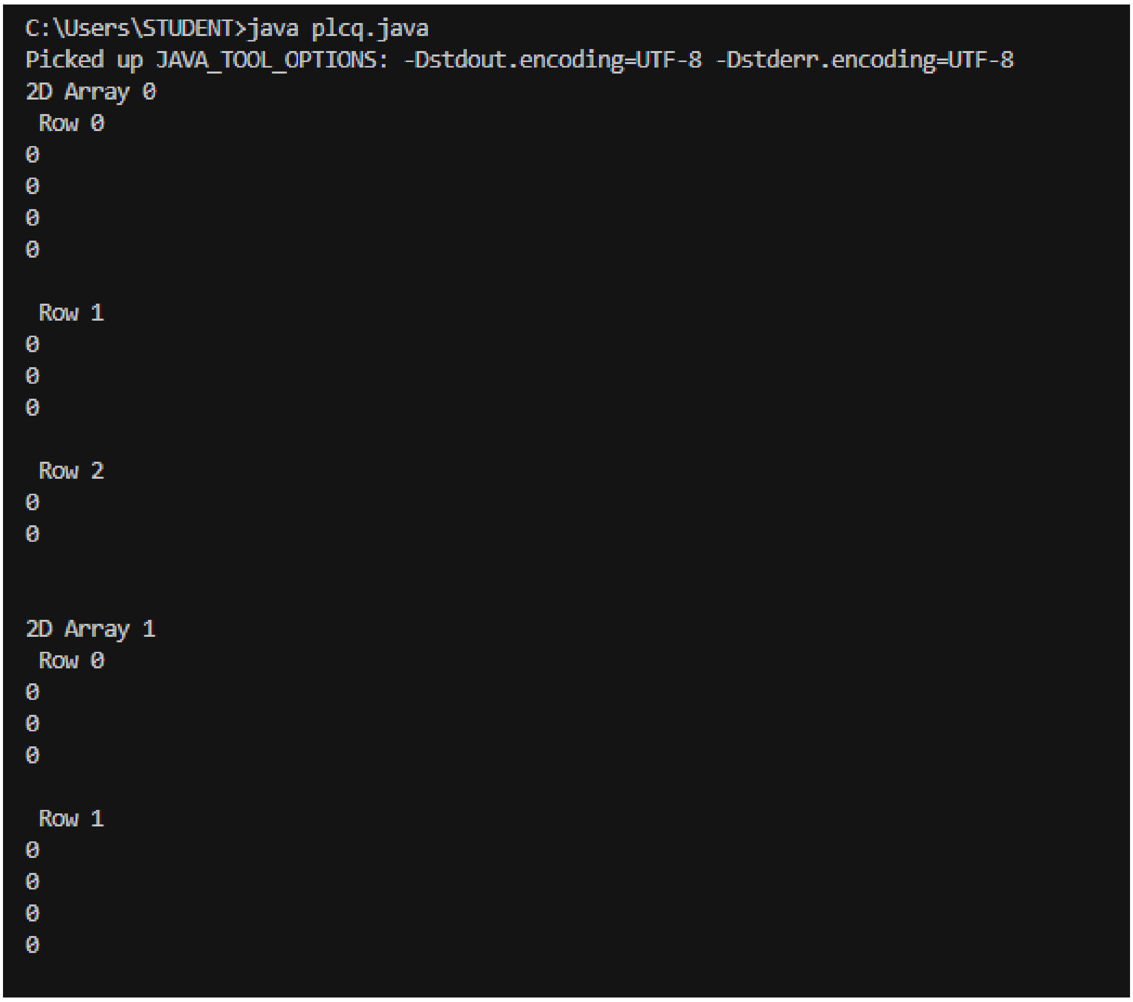
```
import java.util.*;
class plcq{
    public static void main(String[] args) {
```



```
        Scanner sc = new Scanner(System.in);
        int[][][] arr = new int[2][][];

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

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

        for (int i = 0; i < arr.length; i++) {
            System.out.println("2D Array " + i);
            for (int j = 0; j < arr[i].length;
j++) {
                System.out.println(" Row " + j);
                for (int k = 0; k <
arr[i][j].length; k++) {

System.out.println(arr[i][j][k] + " ");
                }
                System.out.println();
            }
```



```
            System.out.println();
        }
    }
}
```



```
C:\Users\STUDENT>java plcq.java
Picked up JAVA_TOOL_OPTIONS: -Dstdout.encoding=UTF-8 -Dstderr.encoding=UTF-8
2D Array 0
 Row 0
0
0
0
0

 Row 1
0
0
0

 Row 2
0
0


2D Array 1
 Row 0
0
0
0

 Row 1
0
0
0
0
```

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, and E
number[5] is undefined, number[2] is zero, and number.length() = 5
```