Competitive Programming Lab - 6

Academic year: 2020-2021 **Semester:** Long Sem

Faculty Name: Dr. Ajith Jublison sir **Date:** 4/ 7/ 2022

Student name: Taran Mamidala **Reg. no.:** 19BCE7346

Number of Grounds required

Given an 2D integer array A of size N x 2denoting time intervals of different matches.

Where:

- A[i][0] = start time of the i'th match.
- A[i][1] = end time of the i'th match.

Find the minimum number of grounds required so that all matches can be done.
Problem Constraints

```
1 <= N <= 10
0 <= A[i][0] < A[i][1] <= 2 * 109 Input Format
```

The only argument given is the matrix A. Output Format

Return the minimum number of grounds required so that all matches can be done.

Example Input

Input 1:

```
A = [ [0, 30]
[5, 10]
[15, 20]]
```

Input 2:

```
A = [ [1, 18]
[18, 23]
[15, 29]
[4, 15]
[2, 11]
```

```
[5, 13]
]
```

Example Output

Output 1:

2

Output 2:

4

Example Explanation

Explanation 1:

Match one can be done in ground 1 form 0 - 30.
Match two can be done in ground 2 form 5 - 10.
Match three can be done in ground 2 form 15 - 20 as it is free in this interval.

Explanation 2:

Match one can be done in ground 1 from 1 - 18.
Match five can be done in ground 2 from 2 - 11.
Match four can be done in ground 3 from 4 - 15.
Match six can be done in ground 4 from 5 - 13.
Match three can be done in ground 2 from 15 - 29 as it is free in this interval.
Match two can be done in ground 4 from 18 - 23 as it is free in this interval.

Input 3:

```
A = [ [ 0 , 7]
[ 9 , 15]
[ 19 , 25]
[ 18 , 22]
[ 14 , 17]
[ 8 , 13]
[12 , 23]
[ 16 , 20]
]
```

Explanation:

Match one can be done in ground 1 from 0 - 7.
Match two can be done in ground 2 from 9 - 15.
Match three can be done in ground 3 from 19 - 25.
Match seven can be done in ground 4 from 12 - 22.

Match four can be done in ground 2 from 18 - 23 as it is free in this interval
Match five can be done in ground 1 from 14 - 17 as it is free in this interval.
Match six can be done in ground 1 from 8 - 13 as it is free in this interval
Match eight can be done in ground 2 from 16 - 20 as it is free in this interval.

Output 3:

4

Input 4:

```
A= [ [ 2 , 13]
[ 7 ,16]
[ 17 , 24]
[20 , 26]
[19 , 22] [25 , 30]
]
```

Match one can be done in ground 1 from 2 - 13.
Match two can be done in ground 2 from 7 - 16.
Match three can be done in ground 3 from 17 - 24.
Match four can be done in ground 2 from 20 - 26 as it is free in this interval
Match five can be done in ground 1 from 19 - 22 as it is free in this interval.
Match six can be done in ground 3 from 25 - 30 as it is free in this interval

Output 3:

3

CODE:

```
import java.util.*;

public class NoOfGrounds {
	public static int getGround(int[][] A) {
	int n = A.length;
	int[] begin = new int[n];
	int[] end = new int[n];
	for (int i = 0; i < n; i++) {
		begin[i] = A[i][0];
		end[i] = A[i][1];
	}
	Arrays.sort(begin);
```

```
        Arrays.sort(end);
        int i = 1, j = 0, current = 1;
        int result = 1;
        while (i < begin.length && j < end.length) {
            if (begin[i] < end[j]) {
                current++;
                i++;
            } else {
                current--;
                j++;
            }
            result = Integer.max(result, current);
        }
        return result;
        }
        public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter No.of rows in an array(m) : ");
        int m=sc.nextInt();
        System.out.print("\nEnter No.of columns in an array(n) : ");
        int n=sc.nextInt();

        int[][] A = new int[m][n];
        System.out.print("\n\nA = [");

        for(int i=0; i<m;i++){
            System.out.print("\n\t\t\t[");
            A[i][0] = sc.nextInt();
            for(int j=1; j<n;j++){
            System.out.print(",");
            A[i][j] = sc.nextInt();
            }
            System.out.print("]");
        }
        System.out.print("\n\t]");
        sc.close();


        System.out.println("\n\nOutput : "+getGround(A));
        }
}
```

Output:

Result

compiled and executed in 42.421 sec(s)

```
Enter No.of rows in an array(m) : 3
Enter No.of columns in an array(n) : 2

A = [
            [0 ,30 ]
            [5 ,10 ]
            [15 ,20 ]
    ]

Output : 2
```

Result

compiled and executed in 69.445 sec(s)

```
Enter No.of rows in an array(m) : 6
Enter No.of columns in an array(n) : 2

A = [
            [1 ,18 ]
            [18 ,23 ]
            [15 ,29 ]
            [4 ,15 ]
            [2 ,11 ]
            [5 ,13 ]
    ]

Output : 4
```

compiled and executed in 62.377 sec(s)

```
Enter No.of rows in an array(m) : 8
Enter No.of columns in an array(n) : 2

A = [
            [0 ,7 ]
            [9 ,15 ]
            [19 ,25 ]
            [18 ,22 ]
            [14 ,17 ]
            [8 ,13 ]
            [12 ,23 ]
            [16 ,20 ]
    ]

Output : 4
```

compiled and executed in 53.43 sec(s)

```
Enter No.of rows in an array(m) : 6
Enter No.of columns in an array(n) : 2

A = [
            [2 ,13 ]
            [7 ,16 ]
            [17 ,24 ]
            [20 ,26 ]
            [19 ,22 ]
            [25 ,30 ]
    ]

Output : 3
```

Result

compiled and executed in 48.357 sec(s)

```
Enter No.of rows in an array(m) : 4
Enter No.of columns in an array(n) : 3

A = [
            [3 ,5 ,12 ]
            [7 ,15 ,8 ]
            [9 ,5 ,4 ]
            [13 ,1 ,7 ]
    ]

Output : 1
```

compiled and executed in 54.912 sec(s)

```
Enter No.of rows in an array(m) : 5
Enter No.of columns in an array(n) : 3

A = [
            [4 ,5 ,6 ]
            [2 ,7 ,1 ]
            [9 ,2 ,8 ]
            [2 ,9 ,3 ]
            [5 ,8 ,7 ]
    ]

Output : 2
```

Result

compiled and executed in 13.895 sec(s)

```
Enter No.of rows in an array(m) : 3
Enter No.of columns in an array(n) : 2

A = [
            [4 ,12 ]
            [34 ,56 ]
            [23 ,76 ]
    ]

Output : 2
```