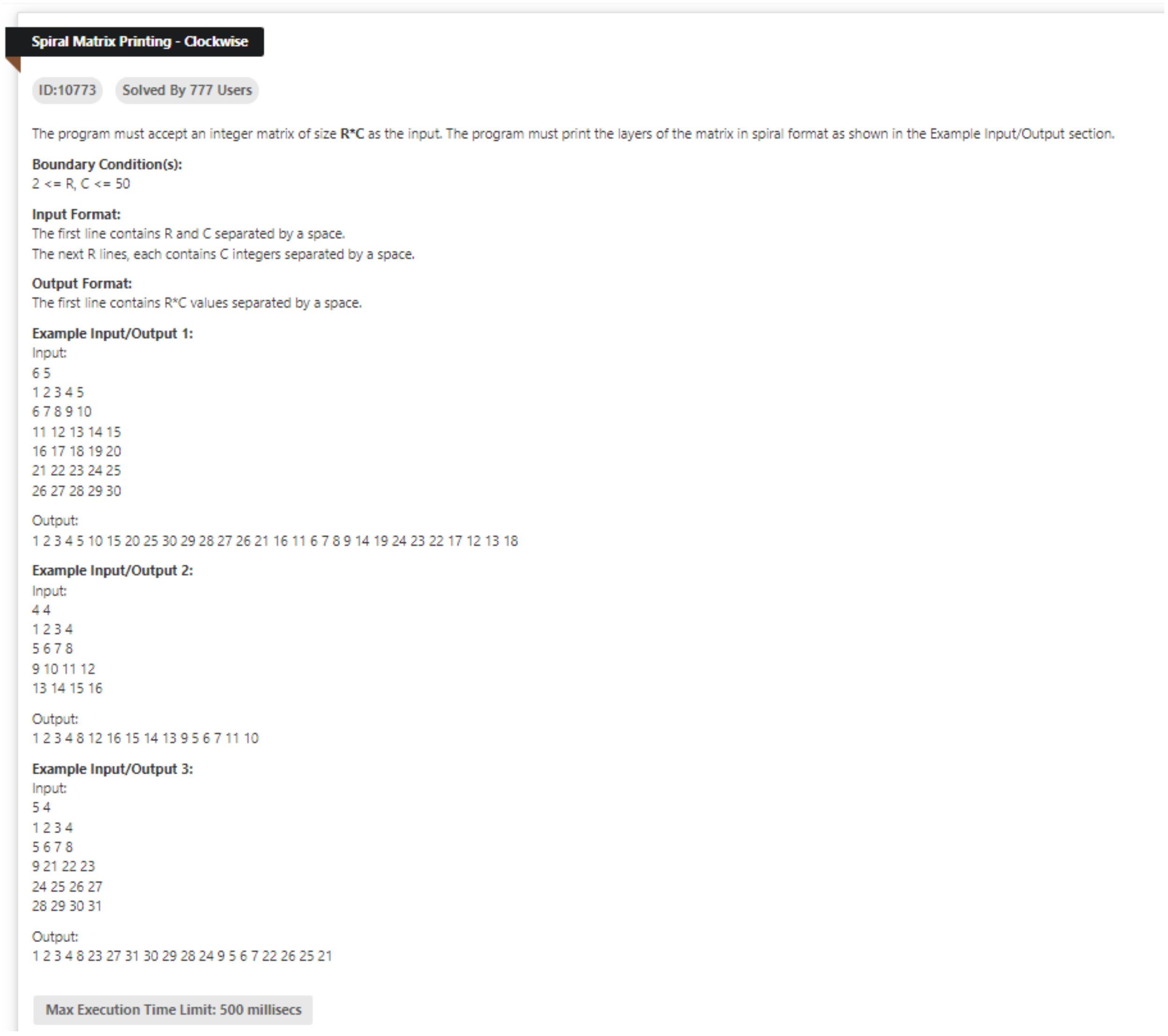
DAY 3:

SESSION 1:



Spiral Matrix Printing - Clockwise

ID:10773 Solved By 777 Users

The program must accept an integer matrix of size **R*C** as the input. The program must print the layers of the matrix in spiral format as shown in the Example Input/Output section.

Boundary Condition(s):
2 <= R, C <= 50

Input Format:
The first line contains R and C separated by a space.
The next R lines, each contains C integers separated by a space.

Output Format:
The first line contains R*C values separated by a space.

Example Input/Output 1:
Input:
```
6 5
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25
26 27 28 29 30
```
Output:
```
1 2 3 4 5 10 15 20 25 30 29 28 27 26 21 16 11 6 7 8 9 14 19 24 23 22 17 12 13 18
```

Example Input/Output 2:
Input:
```
4 4
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
```
Output:
```
1 2 3 4 8 12 16 15 14 13 9 5 6 7 11 10
```

Example Input/Output 3:
Input:
```
5 4
1 2 3 4
5 6 7 8
9 21 22 23
24 25 26 27
28 29 30 31
```
Output:
```
1 2 3 4 8 23 27 31 30 29 28 24 9 5 6 7 22 26 25 21
```

Max Execution Time Limit: 500 millisecs

Code:

```
import java.util.*;
public class matrixtraversal {

    public static void printLeftToRight(int matrix[][],int row,int startCol,int
endCol){
        for(int col=startCol;col<=endCol;col++){
            System.out.print(matrix[row][col]+" ");
        }
    }
    public static void printTopToBottom(int matrix[][],int col,int startRow ,int
endRow){
        for(int row=startRow;row<=endRow;row++){
            System.out.print(matrix[row][col]+" ");
        }
    }
    public static void printRightToLeft(int matrix[][],int row,int startCol,int
endCol){
        for(int col=endCol;col>=startCol;col--){
            System.out.print(matrix[row][col]+" ");
        }
    }
    public static void printBottomToTop(int matrix[][],int col,int startRow,int
endRow){
        for(int row=endRow;row>=startRow;row--){
            System.out.print(matrix[row][col]+" ");
        }
    }

    public static void main(String[] args) {
        //Your Code Here
        Scanner in=new Scanner(System.in);
        int R=in.nextInt();
        int C=in.nextInt();
        int matrix[][] = new int[R][C];
        for(int row=0;row<R;row++){
            for(int col=0;col<C;col++){
                matrix[row][col]=in.nextInt();
            }
        }

        int topRow=0,bottomRow=R-1,leftCol=0,rightCol=C-1;
        while(topRow<=bottomRow && leftCol<=rightCol){
            printLeftToRight(matrix,topRow,leftCol,rightCol);
            printTopToBottom(matrix,rightCol,topRow+1,bottomRow);
            if(topRow!=bottomRow){
                printRightToLeft(matrix,bottomRow,leftCol,rightCol-1);
            }
            if(leftCol!=rightCol){
                printBottomToTop(matrix,leftCol,topRow+1,bottomRow-1);
            }
            topRow++;
            bottomRow--;
            leftCol++;
```

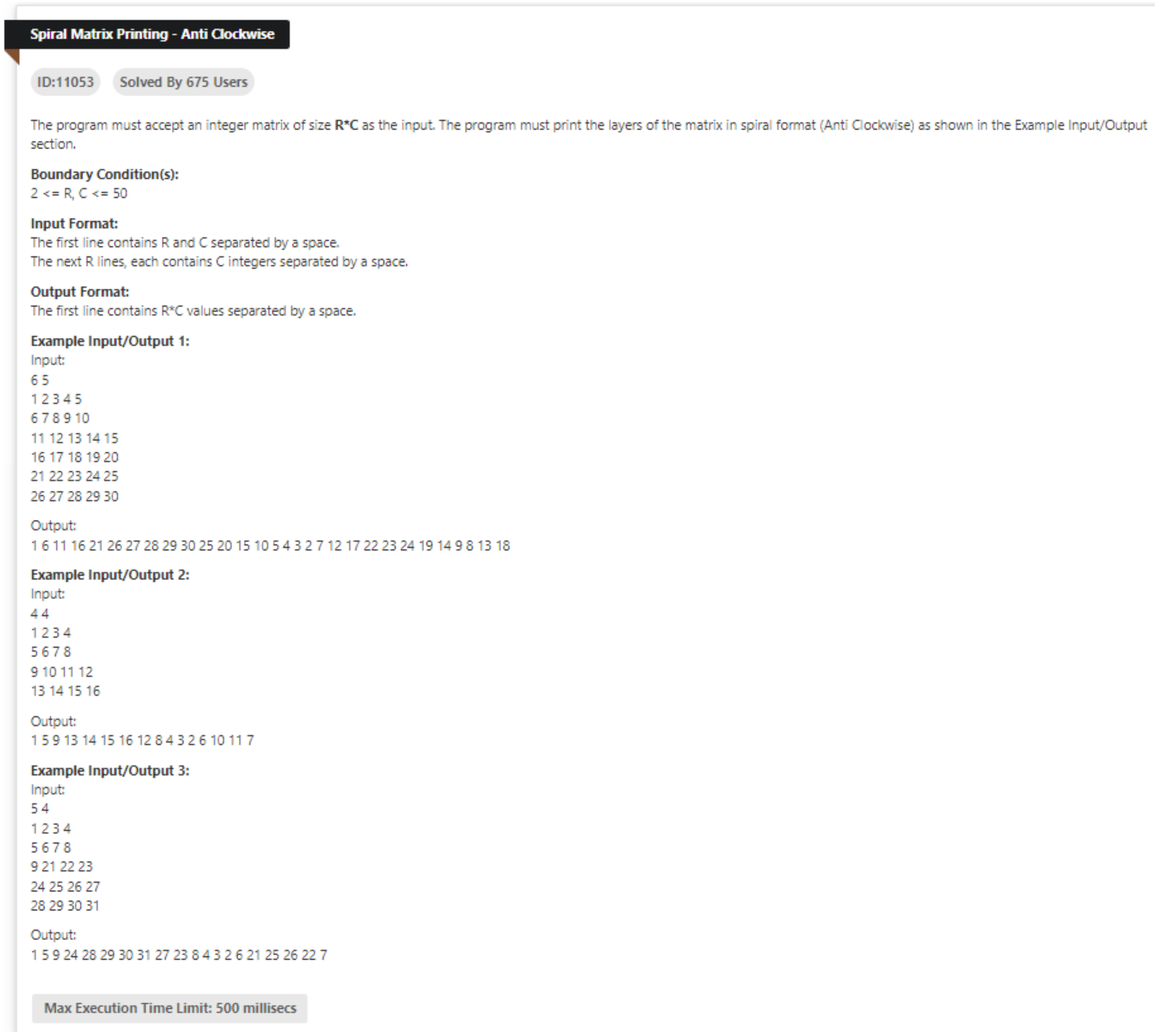
```
                rightCol--;
            }
        }
}
```

Stimulation:

https://pythontutor.com/visualize.html#code=%0Apublic%20class%20matrixTraversalClockwise%20%7B%0A%0A%20%20%20%20public%20static%20void%20printLeftToRight%28int%20matrix%5B%5D%5B%5D,int%20row,int%20startCol,int%20endCol%29%7B%0A%20%20%20%20%20%20%20%20for%28int%20col%3DstartCol%3Bcol%3C%3DendCol%3Bcol%2B%2B%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20System.out.print%28matrix%5Brow%5D%5Bcol%5D%2B%22%20%22%29%3B%0A%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%7D%0A%20%20%20%20%0A%20%20%20%20public%20static%20void%20printTopToBottom%28int%20matrix%5B%5D%5B%5D,int%20col,int%20startRow%20,int%20endRow%29%7B%0A%20%20%20%20%20%20%20%20for%28int%20row%3DstartRow%3Brow%3C%3DendRow%3Brow%2B%2B%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20System.out.print%28matrix%5Brow%5D%5Bcol%5D%2B%22%20%22%29%3B%0A%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%7D%0A%20%20%20%20public%20static%20void%20printRightToLeft%28int%20matrix%5B%5D%5B%5D,int%20row,int%20startCol,int%20endCol%29%7B%0A%20%20%20%20%20%20%20%20for%28int%20col%3DendCol%3Bcol%3E%3DstartCol%3Bcol--%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20System.out.print%28matrix%5Brow%5D%5Bcol%5D%2B%22%20%22%29%3B%0A%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%7D%0A%20%20%20%20public%20static%20void%20printBottomToTop%28int%20matrix%5B%5D%5B%5D,int%20col,int%20startRow,int%20endRow%29%7B%0A%20%20%20%20%20%20%20%20for%28int%20row%3DendRow%3Brow%3E%3DstartRow%3Brow--%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20System.out.print%28matrix%5Brow%5D%5Bcol%5D%2B%22%20%22%29%3B%0A%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%7D%0A%20%20%20%20%0A%20%20%20%20public%20static%20void%20main%28String%5B%5D%20args%29%20%7B%0A%0A%20%20%20%20%20%20%20%20int%20R%3D4%3B%0A%20%20%20%20%20%20%20%20int%20C%3D4%3B%0A%20%20%20%20%20%20%20%20int%20matrix%5B%5D%5B%5D%20%3D%20%7B%7B1,2,3,4%7D,%7B5,6,7,8%7D,%7B9,10,11,12%7D,%7B13,14,15,16%7D%7D%3B%0A%0A%20%20%20%20%20%20%20%20int%20topRow%3D0,bottomRow%3DR-1,leftCol%3D0,rightCol%3DC-1%3B%0A%20%20%20%20%20%20%20%20while%28topRow%3C%3DbottomRow%20%26%26%20leftCol%3C%3DrightCol%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20printLeftToRight%28matrix,topRow,leftCol,rightCol%29%3B%0A%20%20%20%20%20%20%20%20%20%20%20%20printTopToBottom%28matrix,rightCol,topRow%2B1,bottomRow%29%3B%0A%20%20%20%20%20%20%20%20%20%20%20%20if%28topRow!%3DbottomRow%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20printRightToLeft%28matrix,bottomRow,leftCol,rightCol-1%29%3B%0A%20%20%20%20%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20%20%20%20%20%20if%28leftCol!%3DrightCol%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20printBottomToTop%28matrix,leftCol,topRow%2B1,bottomRow-1%29%3B%0A%20%20%20%20%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20%20%20%20%20%20topRow%2B%2B%3B%0A%20%20%20%20%20%20%20%20%20%20%20%20bottomRow--

%3B%0A%20%20%20%20%20%20%20%20%20%20%20%20leftCol%2B%2B%3B%0A%20%20%20%20%20%20%20%20%20%20%20%20rightCol--%3B%0A%20%20%20%20%20%20%20%20%7D%0A%0A%20%20%20%20%7D%0A%7D&cumulative=false&curInstr=90&heapPrimitives=nevernest&mode=display&origin=opt-frontend.js&py=java&rawInputLstJSON=%5B%5D&textReferences=false

Problem 2:



Spiral Matrix Printing - Anti Clockwise

ID:11053 Solved By 675 Users

The program must accept an integer matrix of size **R*C** as the input. The program must print the layers of the matrix in spiral format (Anti Clockwise) as shown in the Example Input/Output section.

Boundary Condition(s):
2 <= R, C <= 50

Input Format:
The first line contains R and C separated by a space.
The next R lines, each contains C integers separated by a space.

Output Format:
The first line contains R*C values separated by a space.

Example Input/Output 1:
Input:
```
6 5
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25
26 27 28 29 30
```

Output:
```
1 6 11 16 21 26 27 28 29 30 25 20 15 10 5 4 3 2 7 12 17 22 23 24 19 14 9 8 13 18
```

Example Input/Output 2:
Input:
```
4 4
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
```

Output:
```
1 5 9 13 14 15 16 12 8 4 3 2 6 10 11 7
```

Example Input/Output 3:
Input:
```
5 4
1 2 3 4
5 6 7 8
9 21 22 23
24 25 26 27
28 29 30 31
```

Output:
```
1 5 9 24 28 29 30 31 27 23 8 4 3 2 6 21 25 26 22 7
```

Max Execution Time Limit: 500 millisecs

Code:

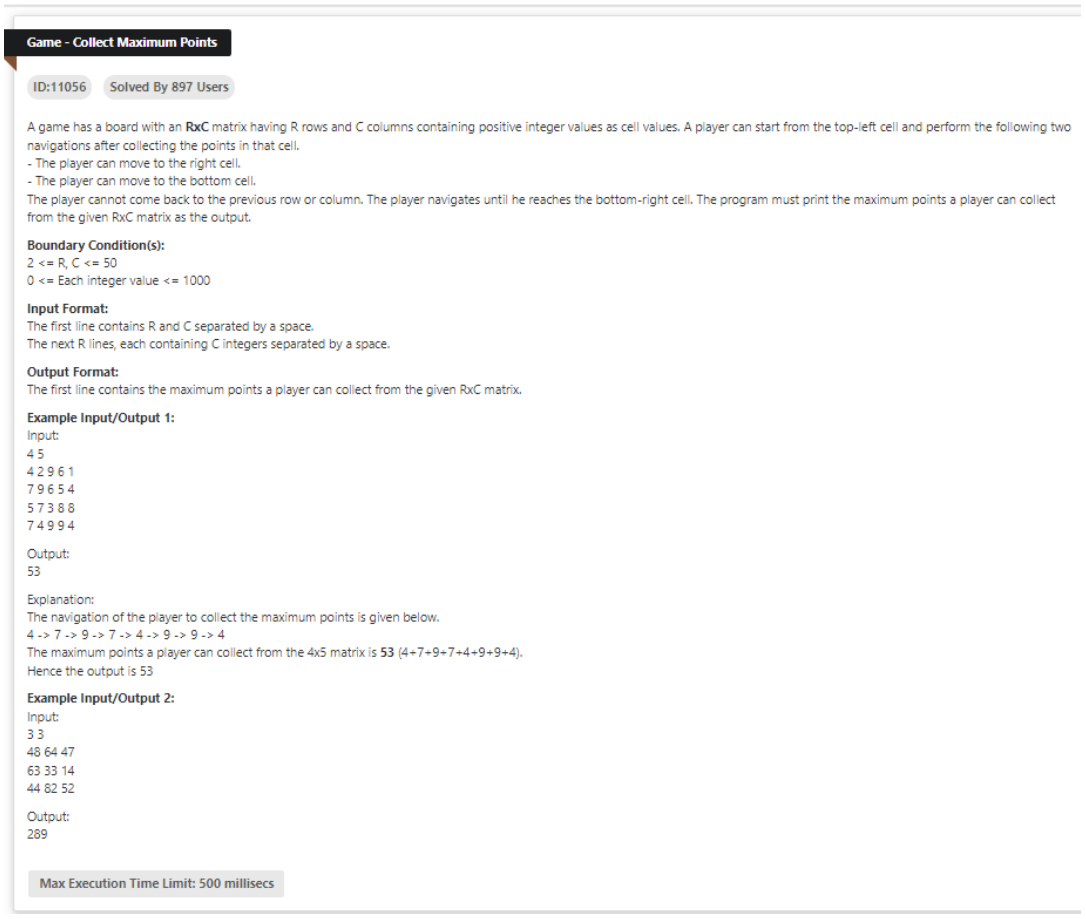
```
import java.util.*;
public class matrixTraversalAntiClockwise {

        //Your Code Here
        public static void printLeftToRight(int matrix[][],int row,int 
startCol,int endCol){
            for(int col=startCol;col<=endCol;col++){
                System.out.print(matrix[row][col]+" ");
            }
        }
        public static void printTopToBottom(int matrix[][],int col,int 
startRow,int endRow){
            for(int row=startRow;row<=endRow;row++ ){
                System.out.print(matrix[row][col]+" ");
            }
        }
        public static void printRightToLeft(int matrix[][],int row ,int 
startCol,int endCol){
            for(int col=endCol;col>=startCol;col--){
                System.out.print(matrix[row][col]+" ");
            }
        }
        public static void printBottomToTop(int matrix[][],int col ,int startRow 
,int endRow){
            for(int row=endRow;row>=startRow;row--){
                System.out.print(matrix[row][col]+" ");
            }
        }

        public static void main(String args[]){
        Scanner in = new Scanner(System.in);
        int R=in.nextInt();
        int C=in.nextInt();
        int matrix[][] = new int[R][C];
        for(int row=0;row<R;row++){
            for(int col=0;col<C;col++){
                matrix[row][col]=in.nextInt();
            }
        }
        int topRow=0,bottomRow=R-1,leftCol=0,rightCol=C-1;
        while(topRow<=bottomRow && leftCol <=rightCol){
            printTopToBottom(matrix,leftCol,topRow,bottomRow);
            printLeftToRight(matrix,bottomRow,leftCol+1,rightCol);
            if(leftCol!=rightCol){
                printBottomToTop(matrix,rightCol,topRow,bottomRow-1);
            }
            if(topRow!=bottomRow){
                printRightToLeft(matrix,topRow,leftCol+1,rightCol-1);
            }

        topRow++;
        bottomRow--;
        leftCol++;
        rightCol--;
        }

        }
    }
```

Stimulation:

https://pythontutor.com/visualize.html#code=%0Apublic%20class%20matrixTraversalClockwise%20%7B
%0A%0A%20%20%20%20public%20static%20void%20printLeftToRight%28int%20matrix%5B%5D%5B%5
D,int%20row,int%20startCol,int%20endCol%29%7B%0A%20%20%20%20%20%20%20%20for%28int%20
col%3DstartCol%3Bcol%3C%3DendCol%3Bcol%2B%2B%29%7B%0A%20%20%20%20%20%20%20%20%2
0%20%20%20System.out.print%28matrix%5Brow%5D%5Bcol%5D%2B%22%20%22%29%3B%0A%20%20
%20%20%20%20%20%20%7D%0A%20%20%20%20%7D%0A%20%20%20%20%0A%20%20%20%20publi
c%20static%20void%20printTopToBottom%28int%20matrix%5B%5D%5B%5D,int%20col,int%20startRow
%20,int%20endRow%29%7B%0A%20%20%20%20%20%20%20%20for%28int%20row%3DstartRow%3Br
ow%3C%3DendRow%3Brow%2B%2B%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20Sys
tem.out.print%28matrix%5Brow%5D%5Bcol%5D%2B%22%20%22%29%3B%0A%20%20%20%20%20%20
%20%20%7D%0A%20%20%20%20%7D%0A%20%20%20%20public%20static%20void%20printRightToLef
t%28int%20matrix%5B%5D%5B%5D,int%20row,int%20startCol,int%20endCol%29%7B%0A%20%20%20
%20%20%20%20%20for%28int%20col%3DendCol%3Bcol%3E%3DstartCol%3Bcol--
%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20System.out.print%28matrix%5Brow%5D
%5Bcol%5D%2B%22%20%22%29%3B%0A%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%
7D%0A%20%20%20%20public%20static%20void%20printBottomToTop%28int%20matrix%5B%5D%5B%
5D,int%20col,int%20startRow,int%20endRow%29%7B%0A%20%20%20%20%20%20%20%20for%28int%
20row%3DendRow%3Brow%3E%3DstartRow%3Brow--
%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20System.out.print%28matrix%5Brow%5D
%5Bcol%5D%2B%22%20%22%29%3B%0A%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%
7D%0A%20%20%20%20%0A%20%20%20%20public%20static%20void%20main%28String%5B%5D%20ar
gs%29%20%7B%0A%0A%20%20%20%20%20%20%20%20int%20R%3D4%3B%0A%20%20%20%20%20%
20%20%20int%20C%3D4%3B%0A%20%20%20%20%20%20%20%20int%20matrix%5B%5D%5B%5D%20
%3D%20%7B%7B1,2,3,4%7D,%7B5,6,7,8%7D,%7B9,10,11,12%7D,%7B13,14,15,16%7D%7D%3B%0A%0A
%20%20%20%20%20%20%20%20int%20topRow%3D0,bottomRow%3DR-1,leftCol%3D0,rightCol%3DC-

1%3B%0A%20%20%20%20%20%20%20%20while%28topRow%3C%3DbottomRow%20%26%26%20leftC
ol%20%3C%3DrightCol%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20printTopToBotto
m%28matrix,leftCol,topRow,bottomRow%29%3B%0A%20%20%20%20%20%20%20%20%20%20%20%2
0printLeftToRight%28matrix,bottomRow,leftCol%2B1,rightCol%29%3B%0A%20%20%20%20%20%2
0%20%20%20%20%20%20%0A%20%20%20%20%20%20%20%20%20%20%20%20if%28leftCol!%3DrightCol
%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20printBottomToTop%28
matrix,rightCol,topRow,bottomRow-
1%29%3B%0A%20%20%20%20%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20
%20%20%20%20%20if%28topRow!%3DbottomRow%29%7B%0A%20%20%20%20%20%20%20%20%20
%20%20%20%20%20%20%20printRightToLeft%28matrix,topRow,leftCol%2B1,rightCol-
1%29%3B%0A%20%20%20%20%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20
%20%0A%20%20%20%20%20%20%20%20topRow%2B%2B%3B%0A%20%20%20%20%20%20%20%20bo
ttomRow--
%3B%0A%20%20%20%20%20%20%20%20leftCol%2B%2B%3B%0A%20%20%20%20%20%20%20%20rig
htCol--
%3B%0A%20%20%20%20%20%20%20%20%7D%0A%0A%20%20%20%20%7D%0A%7D&cumulative=fals
e&curInstr=44&heapPrimitives=nevernest&mode=display&origin=opt-
frontend.js&py=java&rawInputLstJSON=%5B%5D&textReferences=false

Session 2:



Game - Collect Maximum Points

ID:11056 Solved By 897 Users

A game has a board with an **RxC** matrix having R rows and C columns containing positive integer values as cell values. A player can start from the top-left cell and perform the following two navigations after collecting the points in that cell.
- The player can move to the right cell.
- The player can move to the bottom cell.
The player cannot come back to the previous row or column. The player navigates until he reaches the bottom-right cell. The program must print the maximum points a player can collect from the given RxC matrix as the output.

Boundary Condition(s):
2 <= R, C <= 50
0 <= Each integer value <= 1000

Input Format:
The first line contains R and C separated by a space.
The next R lines, each containing C integers separated by a space.

Output Format:
The first line contains the maximum points a player can collect from the given RxC matrix.

Example Input/Output 1:
Input:
```
4 5
4 2 9 6 1
7 9 6 5 4
5 7 3 8 8
7 4 9 9 4
```

Output:
```
53
```

Explanation:
The navigation of the player to collect the maximum points is given below.
4 -> 7 -> 9 -> 7 -> 4 -> 9 -> 9 -> 4
The maximum points a player can collect from the 4x5 matrix is **53** (4+7+9+7+4+9+9+4).
Hence the output is 53

Example Input/Output 2:
Input:
```
3 3
48 64 47
63 33 14
44 82 52
```

Output:
```
289
```

Max Execution Time Limit: 500 millisecs

Code:

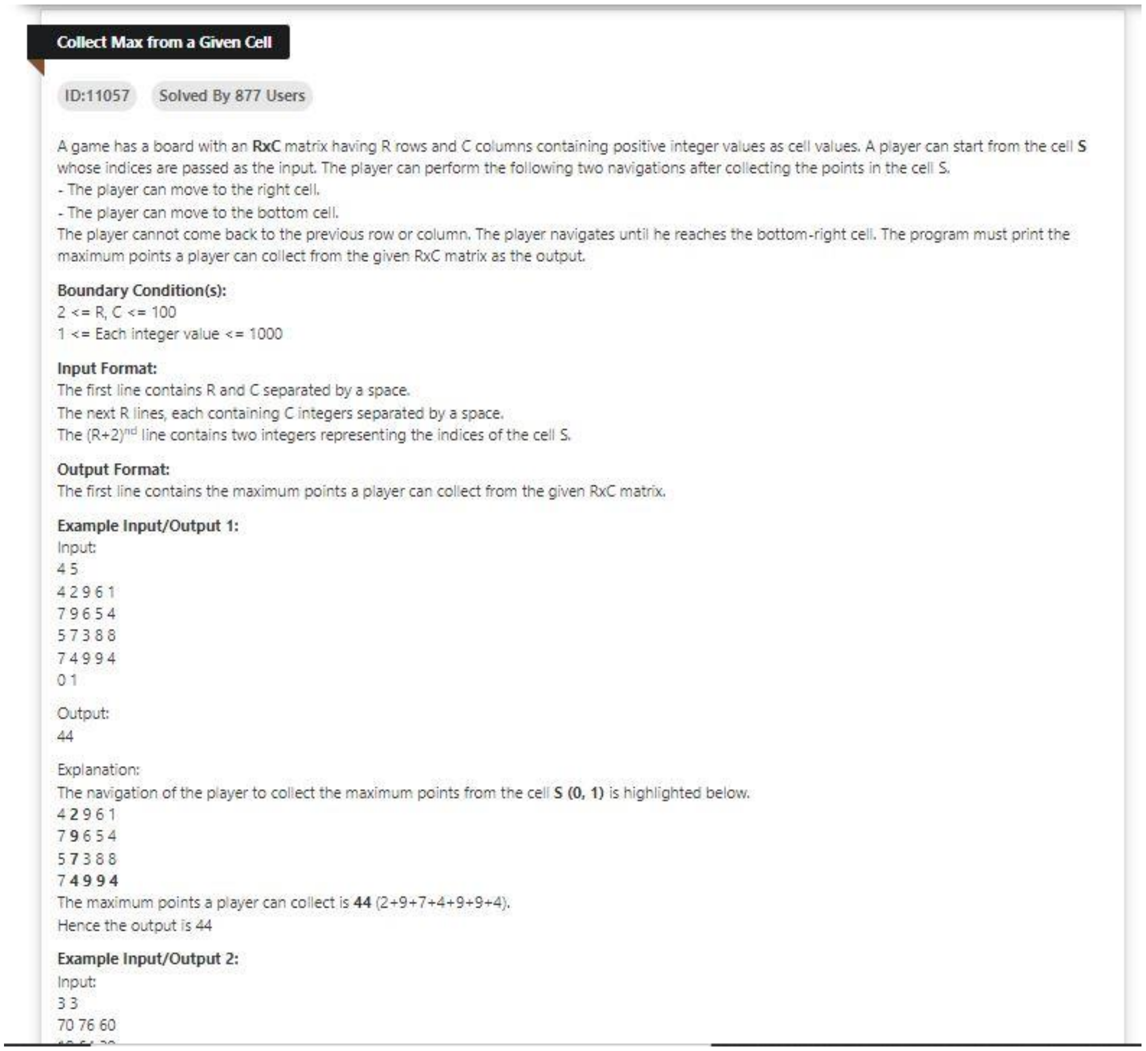
```
import java.util.*;
public class collectMaxPoints {

    public static void main(String[] args) {
        //Your Code Here
        Scanner in = new Scanner(System.in);
        int R=in.nextInt();
        int C=in.nextInt();
        int matrix[][] = new int[R][C];

        for(int row=0;row<R;row++){
            for(int col=0;col<C;col++){
                matrix[row][col]=in.nextInt();
            }
        }
        int max[][] = new int[R][C];
        max[0][0]=matrix[0][0];
        for(int col=1;col<C;col++){
            max[0][col]=matrix[0][col]+max[0][col-1];
        }
        for(int row=1;row<R;row++){
            max[row][0]=matrix[row][0]+max[row-1][0];
        }
        for(int row=1;row<R;row++){
            for(int col=1;col<C;col++){
                max[row][col]=Math.max(max[row][col-1],max[row-1][col])+
matrix[row][col];
            }
        }
        System.out.println(max[R-1][C-1]);
    }
}
```

Stimulation:

https://pythontutor.com/visualize.html#code=%0Apublic%20class%20collectMaxPoints%20%7B%0A%0A%20%20%20%20public%20static%20void%20main%28String%5B%5D%20args%29%20%7B%0A%20%20%20%20%20%20%20%20//Your%20Code%20Here%0A%0A%20%20%20%20%20%20%20%20int%20R%3D3%3B%0A%20%20%20%20%20%20%20%20int%20C%3D3%3B%0A%20%20%20%20%20%20%20%20int%20matrix%5B%5D%5B%5D%20%3D%20%7B%7B48,64,47%7D,%7B63,33,14%7D,%7B44,82,52%7D%7D%3B%0A%0A%20%20%20%20%20%20%20%20int%20max%5B%5D%5B%5D%20%3D%20new%20int%5BR%5D%5BC%5D%3B%0A%20%20%20%20%20%20%20%20max%5B0%5D%5B0%5D%3Dmatrix%5B0%5D%5B0%5D%3B%0A%20%20%20%20%20%20%20%20for%28int%20col%3D1%3Bcol%3CC%3Bcol%2

B%2B%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20max%5B0%5D%5Bcol%5D%3Dmat
rix%5B0%5D%5Bcol%5D%2Bmax%5B0%5D%5Bcol-
1%5D%3B%0A%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20%20for%28int%2
0row%3D1%3Brow%3CR%3Brow%2B%2B%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%2
0max%5Brow%5D%5B0%5D%3Dmatrix%5Brow%5D%5B0%5D%2Bmax%5Brow-
1%5D%5B0%5D%3B%0A%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20%20for
%28int%20row%3D1%3Brow%3CR%3Brow%2B%2B%29%7B%0A%20%20%20%20%20%20%20%20%20
%20%20%20for%28int%20col%3D1%3Bcol%3CC%3Bcol%2B%2B%29%7B%0A%20%20%20%20%20%
20%20%20%20%20%20%20%20%20%20%20max%5Brow%5D%5Bcol%5D%3DMath.max%28max%5Brow%5
D%5Bcol-1%5D,max%5Brow-
1%5D%5Bcol%5D%29%2B%20matrix%5Brow%5D%5Bcol%5D%3B%0A%20%20%20%20%20%20%20%20
%20%20%20%20%7D%0A%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20%20S
ystem.out.println%28max%5BR-1%5D%5BC-
1%5D%29%3B%0A%20%20%20%20%7D%0A%7D&cumulative=false&curInstr=36&heapPrimitives=never
nest&mode=display&origin=opt-frontend.js&py=java&rawInputLstJSON=%5B%5D&textReferences=false

Problem 2:


Collect Max from a Given Cell

ID:11057 Solved By 877 Users

A game has a board with an **RxC** matrix having R rows and C columns containing positive integer values as cell values. A player can start from the cell **S** whose indices are passed as the input. The player can perform the following two navigations after collecting the points in the cell S.
- The player can move to the right cell.
- The player can move to the bottom cell.
The player cannot come back to the previous row or column. The player navigates until he reaches the bottom-right cell. The program must print the maximum points a player can collect from the given RxC matrix as the output.

Boundary Condition(s):
2 <= R, C <= 100
1 <= Each integer value <= 1000

Input Format:
The first line contains R and C separated by a space.
The next R lines, each containing C integers separated by a space.
The (R+2)nd line contains two integers representing the indices of the cell S.

Output Format:
The first line contains the maximum points a player can collect from the given RxC matrix.

Example Input/Output 1:
Input:
```
4 5
4 2 9 6 1
7 9 6 5 4
5 7 3 8 8
7 4 9 9 4
0 1
```
Output:
```
44
```
Explanation:
The navigation of the player to collect the maximum points from the cell **S (0, 1)** is highlighted below.
4 **2** 9 6 1
7 **9** 6 5 4
5 **7** 3 8 8
7 **4 9 9 4**
The maximum points a player can collect is **44** (2+9+7+4+9+9+4).
Hence the output is 44

Example Input/Output 2:
Input:
```
3 3
70 76 60
```

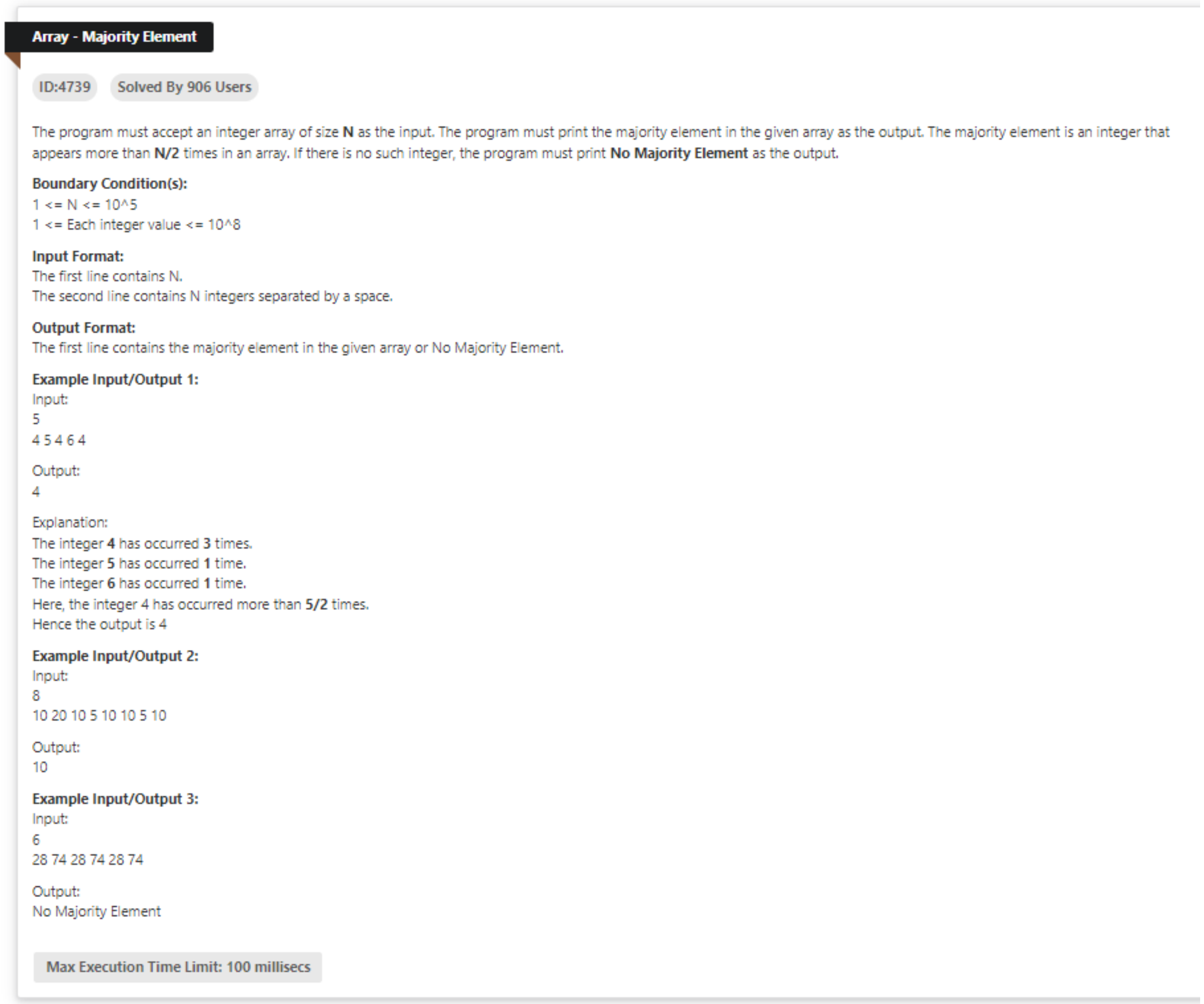
Stimulation:

https://cscircles.cemc.uwaterloo.ca/java_visualize/#code=%0Apublic+class+collectMaxGivenCell+%7B%0A%0A++++public+static+void+main(String%5B%5D+args)+%7B%0A%09%09//Your+Code+Here%0A%0A%09%09int+R%3D4%3B%0A%09%09int+C%3D5%3B%0A%09%09int%5B%5D%5B%5D+matrix%3D+%7B%7B4,2,9,6,1%7D,%7B7,9,6,5,4%7D,%7B5,7,3,8,8%7D,%7B7,4,9,9,4%7D%7D%3B%0A%09%09%0A%0A%09%09%0A%09%09int+startRow%3D1,startCol%3D1%3B%0A%09%09int%5B%5D%5B%5D+max+%3D+new+int%5BR%5D%5BC%5D%3B%0A%09%09max%5BstartRow%5D%5BstartCol%5D%3Dmatrix%5BstartRow%5D%5BstartCol%5D%3B%0A%09%09for(int+col%3DstartCol%2B1%3Bcol%3CC%3Bcol%2B%2B)%7B%0A%09%09++++max%5BstartRow%5D%5Bcol%5D%3D+matrix%5BstartRow%5D%5Bcol%5D%2Bmax%5BstartRow%5D%5Bcol-1%5D%3B%0A%09%09%7D%0A%09%09%0A%09%09for(int+row%3DstartRow%2B1%3Brow%3CR%3Brow%2B%2B)%7B%0A%09%09++++max%5Brow%5D%5BstartCol%5D%3Dmatrix%5Brow%5D%5BstartCol%5D%2Bmax%5Brow-1%5D%5BstartCol%5D%3B%0A%09%09%7D%0A%09%09%0A%09%09for(int+row%3DstartRow%2B1%3Brow%3CR%3Brow%2B%2B)%7B%0A%09%09++++for(int+col%3DstartCol%2B1%3Bcol%3CC%3Bcol%2B%2B)%7B%0A%09%09++++++++max%5Brow%5D%5Bcol%5D%3DMath.max(max%5Brow%5D%5Bcol-1%5D,max%5Brow-1%5D%5Bcol%5D)%2Bmatrix%5Brow%5D%5Bcol%5D%3B%0A%09%09++++%7D%0A%09%09%7D%0A++++System.out.println(max%5BR-1%5D%5BC-1%5D)%3B%0A%09%7D%0A%7D&mode=display&curInstr=57

Code:

```
import java.util.*;
public class collectMaxGivenCell {

    public static void main(String[] args) {
        //Your Code Here
        Scanner in =new Scanner(System.in);
        int R=in.nextInt();
        int C=in.nextInt();
        int[][] matrix= new int[R][C];

        for(int row=0;row<R;row++){
            for(int col=0;col<C;col++){
                matrix[row][col]=in.nextInt();
            }
        }

        int startRow=in.nextInt(),startCol=in.nextInt();
        int[][] max = new int[R][C];
        max[startRow][startCol]=matrix[startRow][startCol];
        for(int col=startCol+1;col<C;col++){
            max[startRow][col]= matrix[startRow][col]+max[startRow][col-1];
        }
        for(int row=startRow+1;row<R;row++){
            max[row][startCol]=matrix[row][startCol]+max[row-1][startCol];
        }
        for(int row=startRow+1;row<R;row++){
            for(int col=startCol+1;col<C;col++){
                max[row][col]=Math.max(max[row][col-1],max[row-
1][col])+matrix[row][col];
            }
        }
    System.out.println(max[R-1][C-1]);
    }
}
```

SESSION 3:



Array - Majority Element

ID:4739 Solved By 906 Users

The program must accept an integer array of size **N** as the input. The program must print the majority element in the given array as the output. The majority element is an integer that appears more than **N/2** times in an array. If there is no such integer, the program must print **No Majority Element** as the output.

Boundary Condition(s):
1 <= N <= 10^5
1 <= Each integer value <= 10^8

Input Format:
The first line contains N.
The second line contains N integers separated by a space.

Output Format:
The first line contains the majority element in the given array or No Majority Element.

Example Input/Output 1:
Input:
5
4 5 4 6 4

Output:
4

Explanation:
The integer **4** has occurred **3** times.
The integer **5** has occurred **1** time.
The integer **6** has occurred **1** time.
Here, the integer 4 has occurred more than **5/2** times.
Hence the output is 4

Example Input/Output 2:
Input:
8
10 20 10 5 10 10 5 10

Output:
10

Example Input/Output 3:
Input:
6
28 74 28 74 28 74

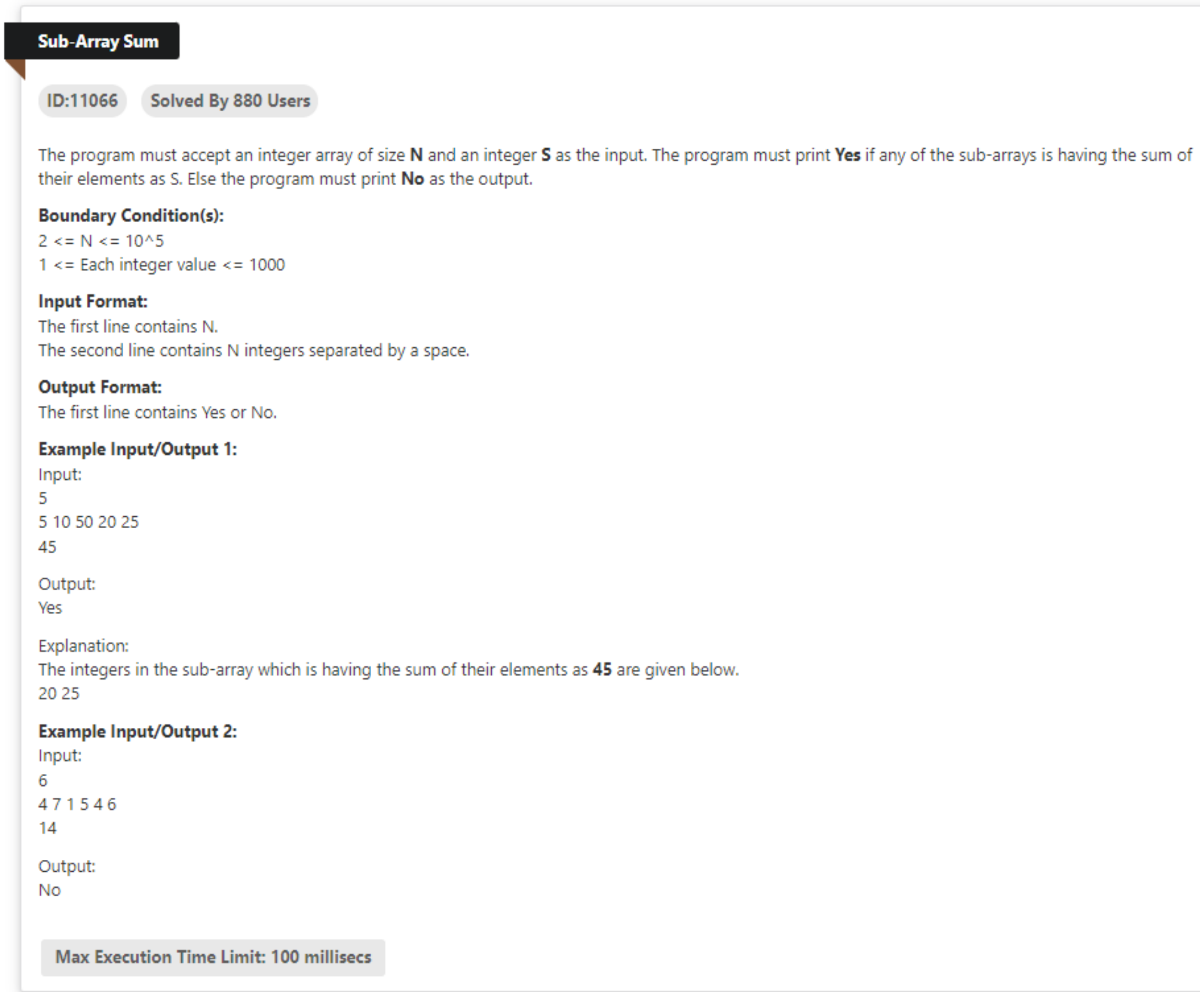
Output:
No Majority Element

Max Execution Time Limit: 100 millisecs

Code:

```
import java.util.*;
public class majorityElement {

    public static void main(String[] args) {
        //Your Code Here
        Scanner in=new Scanner(System.in);
        int N=in.nextInt();
        int[] arr=new int[N];
        for(int i=0;i<N;i++){
            arr[i]=in.nextInt();
        }
        int counter=1,majorityElement=arr[0];
       // int flag=0;
        for(int i=0;i<N;i++){
            if(majorityElement==arr[i]){
                counter++;
            }
            else{
                counter--;
                if(counter ==0){
                    majorityElement=arr[i];
                    counter=1;
                }
            } }
        if(counter>0){
            int actualCount=0;
            for(int i=0;i<N;i++){
                if(arr[i]==majorityElement){
                    actualCount++;
                }
            }
            if(actualCount>N/2){
                System.out.println(majorityElement);
                return;
                //flag=1;
            }
        }
        // if(flag==0){
        //     System.out.println("No Majority Element"); // both commented and
return works // }
        System.out.println("No Majority Element");
    }
}
```

Problem 2:



Sub-Array Sum

ID:11066 Solved By 880 Users

The program must accept an integer array of size **N** and an integer **S** as the input. The program must print **Yes** if any of the sub-arrays is having the sum of their elements as S. Else the program must print **No** as the output.

Boundary Condition(s):
2 <= N <= 10^5
1 <= Each integer value <= 1000

Input Format:
The first line contains N.
The second line contains N integers separated by a space.

Output Format:
The first line contains Yes or No.

Example Input/Output 1:
Input:
5
5 10 50 20 25
45

Output:
Yes

Explanation:
The integers in the sub-array which is having the sum of their elements as **45** are given below.
20 25

Example Input/Output 2:
Input:
6
4 7 1 5 4 6
14

Output:
No

Max Execution Time Limit: 100 millisecs

Code:

```java
import java.util.*;
public class subArraySum {

    public static void main(String[] args) {
        //Your Code Here
        Scanner in=new Scanner(System.in);
        int N=in.nextInt();
        int[] arr=new int[N];
        for(int i=0;i<N;i++){
            arr[i]=in.nextInt();
        }
        int currSum=arr[0];
        int sum=in.nextInt();
        for(int li=0,ri=0;li<N &&ri<N;){
            if(sum==currSum){
                System.out.println("Yes");
                return;
            }
            else if(currSum<sum){
                ri++;
                if(ri<N){
                    currSum+=arr[ri];
                }
            }
            else{
                 currSum-=arr[li];
                 li++;
            }
        }
        System.out.println("No");
    }
}
```

Stimulation:

https://pythontutor.com/visualize.html#code=%0Apublic%20class%20subArraySum%20%7B%0A%0A%2
0%20%20%20public%20static%20void%20main%28String%5B%5D%20args%29%20%7B%0A%0A%20%2
0%20%20%20%20%20%20int%20N%3D5%3B%0A%20%20%20%20%20%20%20%20int%5B%5D%20arr%
3D%7B5,10,50,20,25%7D%3B%0A%0A%20%20%20%20%20%20%20%20int%20currSum%3Darr%5B0%5
D%3B%0A%20%20%20%20%20%20%20%20int%20sum%3D45%3B%0A%20%20%20%20%20%20%20%2
0for%28int%20li%3D0,ri%3D0%3Bli%3CN%20%26%26ri%3CN%3B%29%7B%0A%20%20%20%20%20%20
%20%20%20%20%20%20if%28sum%3D%3DcurrSum%29%7B%0A%20%20%20%20%20%20%20%20%20
%20%20%20%20%20%20%20System.out.println%28%22Yes%22%29%3B%0A%20%20%20%20%20%20
%20%20%20%20%20%20%20%20%20%20return%3B%0A%20%20%20%20%20%20%20%20%20%20%20
%20%7D%0A%20%20%20%20%20%20%20%20%20%20%20%20else%20if%28currSum%3Csum%29%7B
%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20ri%2B%2B%3B%0A%20%20%20
%20%20%20%20%20%20%20%20%20%20%20%20%20if%28ri%3CN%29%7B%0A%20%20%20%20%20%
20%20%20%20%20%20%20%20%20%20%20%20%20%20%20currSum%2B%3Darr%5Bri%5D%3B%0A%2
0%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20
%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20%20%20%20%20%20else%7B%0A%20%20%
20%20%20%20%20%20%20%20%20%20%20%20%20%20%20currSum-
%3Darr%5Bli%5D%3B%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20li%2B
%2B%3B%0A%20%20%20%20%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20%
20%7D%0A%20%20%20%20%20%20%20%20System.out.println%28%22No%22%29%3B%0A%20%20%2
0%20%7D%0A%7D&cumulative=false&curInstr=36&heapPrimitives=nevernest&mode=display&origin=o
pt-frontend.js&py=java&rawInputLstJSON=%5B%5D&textReferences=false