

K.SATYA CHARAN(21071A67A3)

DAA ASSIGNMENT-1

1 .Given a row wise sorted matrix of size $R \times C$ where R and C are always **odd**, find the median of the matrix.

Test Case 1:

Input:

Input:

R = 3, C = 3

M = [[1, 3, 5],

 [2, 6, 9],

 [3, 6, 9]]

Output: 5

Explanation: Sorting matrix elements gives us {1,2,3,3,5,6,6,9,9}. Hence, 5 is median.

PROGRAM:

```
#include <iostream>
```

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
int median(int a[3][3],int r,int c)
```

```
{
```

```
    int m[r*c];
```

```
    int q=0;
```

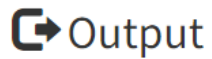
```

for(int i=0;i<r;i++){
    for(int j=0;j<c;j++){
        m[q]=a[i][j];
        q++;
    }
}
for(int a=0;a<q;a++){
    for(int b=0;b<q-a;b++){
        if(m[b]>m[b+1]){
            int temp=m[b];
            m[b]=m[b+1];
            m[b+1]=temp;
        }
    }
}
return m[((r*c)/2)+1];
}

int main(){
    int r=3,c=3;
    int a[3][3] = {{1, 3, 5},{2, 6, 9},{3, 6, 9}};
    cout <<"median of the rowwise sorted matrix is:"<<median(a,r,c) <<"\n";
    return 0;
}

```

output 1:



Output

Accepted 0.004s, 15360KB

median of the rowwise sorted matrix is:5

Test Case 2:

Input:

R = 3, C = 1

M = [[1], [2], [3]]

Output: 2

Explanation: Sorting matrix elements gives us {1,2,3}. Hence, 2 is median.

PROGRAM:

```
#include <iostream>

#include <bits/stdc++.h>

using namespace std;

int median(int a[3][3],int r,int c)
{
    int m[r*c];

    int q=0;

    for(int i=0;i<r;i++){
        for(int j=0;j<c;j++){
            m[q]=a[i][j];

            q++;
        }
    }
```


```

    }
    for(int a=0;a<q;a++){
        for(int b=0;b<q-a;b++){
            if(m[b]>m[b+1]){
                int temp=m[b];
                m[b]=m[b+1];
                m[b+1]=temp;
            }
        }
    }
    return m[((r*c)/2)+1];
}

int main(){
    int r=3,c=1;
    int a[3][3]={{1},{2},{3}};
    cout <<"median of the rowwise sorted matrix is:"<<median(a,r,c) <<"\n";
    return 0;
}

```

output 2:

 Output

Accepted 0.006s, 14264KB

```
median of the rowwise sorted matrix is:2
```

2. Given the arrival and departure times of all trains that reach a railway station, the task is to find the minimum number of platforms required for the railway station so that no train waits. We are given two arrays that represent the arrival and departure times of trains that stop.

Test case 1

Input: `arr[] = {9:00, 9:40, 9:50, 11:00, 15:00, 18:00}, dep[] = {9:10, 12:00, 11:20, 11:30, 19:00, 20:00}`

Output: 3

Explanation: *There are at-most three trains at a time (time between 9:40 to 12:00)*

PROGRAM:

```
#include <iostream>

#include <bits/stdc++.h>

using namespace std;

int findPlatform(int arr[],int dep[],int n){

    int plat_needed=1,result=1;

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

        plat_needed = 1;

        for (int j=0;j<n;j++) {

            if (i!=j)

                if (arr[i]>=arr[j] && dep[j]>=arr[i])

                    plat_needed++;

        }

        result = max(plat_needed, result);

    }

    return result;

}

int main(){
```

```

int arr[]={ 900 , 940 , 950 , 1100 , 1500 , 1800 };

int dep[]={ 910 , 1200 , 1120 , 1130 , 1900 , 2000 };

int n = sizeof(arr)/sizeof(arr[0]);


cout <<"minimum no.of platforms required:\t"<<findPlatform(arr, dep, n);

return 0;

}

```

output 1:

 Output

Accepted 0.004s, 15524KB

```

minimum no.of platforms required:    3

```

Test case 2

Input: `arr[] = {9:00, 9:40}, dep[] = {9:10, 12:00}`

Output: 1

Explanation: Only one platform is needed.

PROGRAM:

```

#include <iostream>

#include <bits/stdc++.h>

using namespace std;

int findPlatform(int arr[],int dep[],int n){

    int plat_needed=1,result=1;

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

        plat_needed = 1;

```


```

        for (int j=0;j<n;j++) {
            if (i!=j)
                if (arr[i]>=arr[j] && dep[j]>=arr[i])
                    plat_needed++;
        }
        result = max(plat_needed, result);
    }
    return result;
}

int main(){
    int arr[]={ 900 , 940 };
    int dep[]={ 910 , 1200 };
    int n = sizeof(arr)/sizeof(arr[0]);
    cout <<"minimum no.of platforms required:\t"<<findPlatform(arr, dep, n);
    return 0;
}

```

output 2:

 Output

Accepted 0.005s, 8980KB

```

minimum no.of platforms required:    1

```
