DAA Assignment

A.Meghana:21071A6771

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

#include<iostream>

#include<bits/stdc++.h>

using namespace std;

int main() {

int R,C;

int a[100][100];

int i,j,l,k;

cout<<"enter number of rows and colums";

cin>>R>>C;

int t[R\*C],m;

if( (R+1)%2 ==0 && (C+1)%2==0)

{

if(1<=R && C<=400)

{

cout<<"enter array elements";

for(i=0;i< R;i++)

{

for(j=0;j<C;j++)

{

cin>>a[i][j];

}

}

int t[R\*C],k=0,i,j;

//copying all elements of matrix into an array

for(i=0;i<R;i++)

{

for(j=0;j<C;j++,k++)

{

t[k]=a[i][j];

}

}

sort(t,t+k);

for(k=0;k<R\*C;k++)

cout<<t[k]<<" ";

int n=(R\*C)/2;

cout<<"Median:"<<t[n];

}

}

else

{

cout<<"Both rows and columns should be odd";

}

}

**Test Case 1:**

**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

R = 3, C = 3

M = [[1, 3, 5],

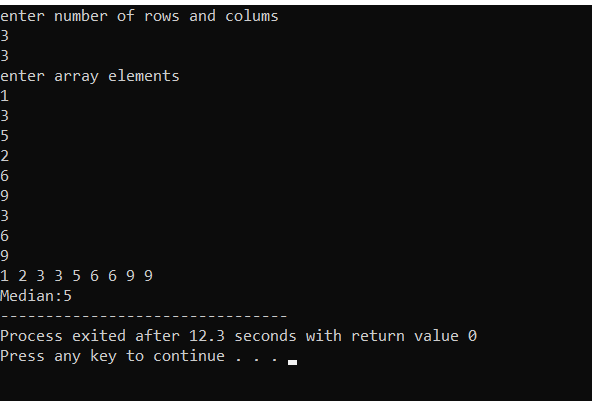
  [2, 6, 9],

  [3, 6, 9]]

**Output:** 5

**Explanation**: Sorting matrix elements gives

us {1,2,3,3,5,6,6,9,9}. Hence, 5 is median.



**Test Case 2:**

**Input:**

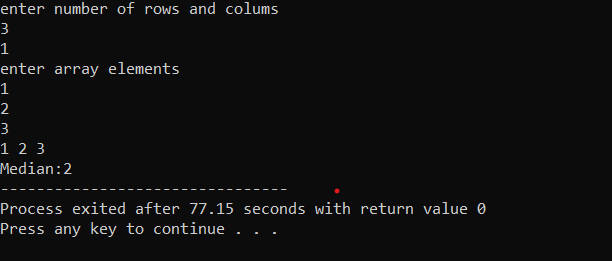
R = 3, C = 1

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

**Output:** 2

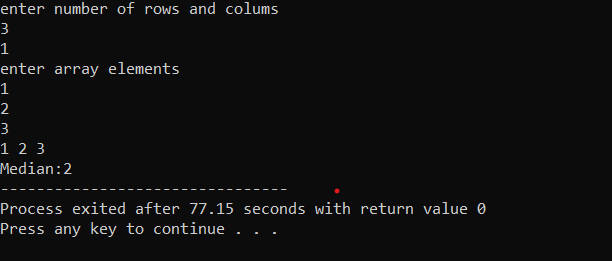
**Explanation**: Sorting matrix elements gives

us {1,2,3}. Hence, 2 is median.



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.

//us {1,2,3}. Hence, 2 is median.



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.

//program

#include<iostream>

#include<bits/stdc++.h>

using namespace std;

int main()

{

int t;

cin>>t;

int arr[t],dep[t];

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

cin>>arr[i];

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

cin>>dep[i];

sort(arr,arr+t);

sort(dep,dep+t);

//p means number of platforms required

int p=1,result=1,i=1,j=0;

while(i<t && j<t){

if(arr[i]<=dep[j]){

p++;

i++;

}

else if(arr[i]>dep[j]){

p--;

j++;

}

if(p>result)

result=p;

}

int p=1,result=1,i=1,j=0;

while(i<t && j<t){

if(arr[i]<=dep[j]){

p++;

i++;

}

else if(arr[i]>dep[j]){

p--;

j++;

}

if(p>result)

result=p;

}

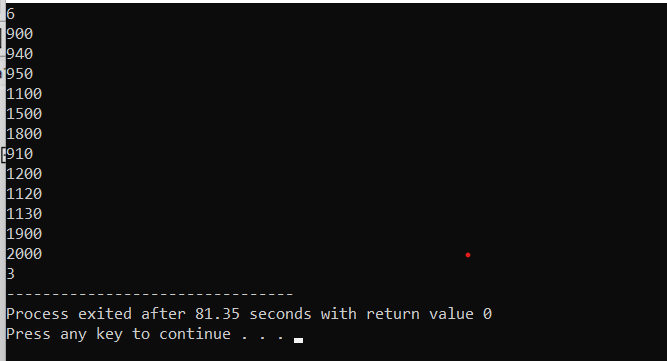
cout<<result;

return 0;

}

**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*

**

**Test case 2**

***Input:****arr[] = {9:00, 9:40}, dep[] = {9:10, 12:00}****Output:****1****Explanation:****Only one platform is needed.*

