

## DAA ASSINGMENT-21071A6779

### QUESTION

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. **5Marks**

#### 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.

### CODE:

```
R=int(input("Enter the number of rows:"))
C=int(input("Enter the number of columns:"))
matrix=[]
for i in range(R):
    a=[]
    for j in range(C):
        a.append(int(input()))
    matrix.append(a)
matrix.sort()
import numpy as nk
h=nk.median(matrix)
print(h)
```

main.py	<div><div>Run</div></div>	Shell
<pre>1 R=int(input("Enter the number of rows:")) 2 C=int(input("Enter the number of columns:")) 3 matrix=[] 4 for i in range(R): 5     a=[] 6     for j in range(C): 7         a.append(int(input())) 8     matrix.append(a) 9 matrix.sort() 10 import numpy as nk 11 h=nk.median(matrix) 12 print('Sorting matrix elements gives us') 13 print(matrix) 14 print(f'Hence {h} is median')</pre>		<pre>Enter the number of rows:3 Enter the number of columns:3 1 3 5 2 6 9 3 6 9 Sorting matrix elements gives us [[1, 3, 5], [2, 6, 9], [3, 6, 9]] Hence 5.0 is median &gt;</pre>

## 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.

main.py	<div><div>Run</div></div>	Shell
<pre>1 R=int(input("Enter the number of rows:")) 2 C=int(input("Enter the number of columns:")) 3 matrix=[] 4 for i in range(R): 5     a=[] 6     for j in range(C): 7         a.append(int(input())) 8     matrix.append(a) 9 matrix.sort() 10 import numpy as nk 11 h=nk.median(matrix) 12 print('Sorting matrix elements gives us') 13 print(matrix) 14 print(f'Hence {h} is median')</pre>		<pre>Enter the number of rows:3 Enter the number of columns:1 1 2 3 Sorting matrix elements gives us [[1], [2], [3]] Hence 2.0 is median &gt;  </pre>

## QUESTION 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.

**5Marks**

### 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)

### Test case 2

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

**Output:** 1

**Explanation:** Only one platform is needed.

## CODE:

```
arrival=list(map(int,input().split()))
departure=list(map(int,input().split()))
n=len(arrival)
arrival.sort()
departure.sort()
platforms=1
required_platforms=1
i=1
j=0
while i<n and j<n:
    if arrival[i]<=departure[j]:
        platforms=platforms+1
        i=i+1
    else:
        platforms=platforms-1
        j=j+1
    required_platforms=max(required_platforms,platforms)
print(required_platforms)
```

## TEST CASE 1:

main.py	Shell
<pre>1 arrival=list(map(int,input().split())) 2 departure=list(map(int,input().split())) 3 n=len(arrival) 4 arrival.sort() 5 departure.sort() 6 platforms=1 7 required_platforms=1 8 i=1 9 j=0 10 while i&lt;n and j&lt;n: 11     if arrival[i]&lt;=departure[j]: 12         platforms=platforms+1 13         i=i+1 14     else: 15         platforms=platforms-1 16         j=j+1 17     required_platforms=max(required_platforms,platforms) 18 print(required_platforms)</pre>	<pre>900 940 950 1100 1500 1800 910 1200 1120 1130 1900 2000 3 &gt;  </pre>

## TEST CASE 2:

main.py	Shell
<pre>1 arrival=list(map(int,input().split())) 2 departure=list(map(int,input().split())) 3 n=len(arrival) 4 arrival.sort() 5 departure.sort() 6 platforms=1 7 required_platforms=1 8 i=1 9 j=0 10 while i&lt;n and j&lt;n: 11     if arrival[i]&lt;=departure[j]: 12         platforms=platforms+1 13         i=i+1 14     else: 15         platforms=platforms-1 16         j=j+1 17     required_platforms=max(required_platforms,platforms) 18 print(required_platforms)</pre>	<pre>900 940 910 1200 1 &gt;  </pre>