DAA Assignment- I

1 .Given a row wise sorted matrix of size R\*C where R and C are always odd, find the median of the matrix. 5Marks

Test Case 1:

from bisect import bisect\_right as upper\_bound

MAX = 100;

def binaryMedian(m, r, d):

pi = m[0][0]

p2 = 0

for i in range(r):

if p[i][0] < pi:

pi= p[i][0]

if p[i][d-1] > p2 :

p2 = p[i][d-1]

desired = (r \* d + 1) // 2

while (pi < p2):

mid = pi + (p2 - pi) // 2

place = [0];

for i in range(r):

j = upper\_bound(p[i], mid)

place[0] = place[0] + j

if place[0] < desired:

pi = mid + 1

else:

p2 = mid

print ("Median is", mi)

return

r, d = 3, 3

m = [ [1, 3, 5], [2, 6, 9], [3, 6, 9]]

binaryMedian(m, r, d)

OUTPUT:

Median is 5

Test Case 2:

from bisect import bisect\_right as upper\_bound

MAX = 100;

def binaryMedian(m, r, d):

pi = m[0][0]

p2 = 0

for i in range(r):

if p[i][0] < pi:

pi = p[i][0]

if p[i][d-1] > p2:

p2= p[i][d-1]

desired = (r \* d + 1) // 2

while (pi < p2):

mid = mi + (p2 - pi) // 2

place = [0];

than or equal to mid

for i in range(r):

j = upper\_bound(m[i], mid)

place[0] = place[0] + j

if place[0] < desired:

pi = mid + 1

else:

p2 = mid

print ("Median is", mi)

return

r, d = 3,1

m= [ [1], [2], [3]]

binaryMedian(m, r, d)

OUTPUT:

Median 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. 5Marks

Test Case 1:

import heapq

def findPlatform(a, dep, n):

a2 = []

for i in range(n):

a2.append([a[i], dep[i]])

a2.sort() #

p = []

count = 1

heapq.heappush(p, a2[0][1])

for i in range(1, n):

if p[0] >= a2[i][0]:

count += 1

else:

heapq.heappop(p)

heapq.heappush(p, a2[i][1])

return count

if \_\_name\_\_ == "\_\_main\_\_":

a= [900, 940, 950, 1100, 1500, 1800]

dep = [910, 1200, 1120, 1130, 1900, 2000]

n = len(a)

print(findPlatform(a, dep, n))

OUTPUT:

3

Test Case 2:

import heapq

def findPlatform(a, dep, n):

a2 = []

for i in range(n):

a2.append([a[i], dep[i]])

a2.sort()

p = []

count = 1

heapq.heappush(p, a2[0][1])

for i in range(1, n):

if p[0] >= a2[i][0]:

count += 1

else:

heapq.heappop(p)

heapq.heappush(p, a2[i][1])

return count

if \_\_name\_\_ == "\_\_main\_\_":

a = [900, 940]

dep = [910, 1200]

n = len(a)

print(findPlatform(a, dep, n))

OUTPUT:

1

Roll number: 21071A6791

GitHub link:https://github.com/hasini