Name: Mansi Jainendra Tandel

SCU id: W1606463

Course: 279 Design and analysis : Algorithms(28009)

Assignment: 4

1. Given a time table of railway trains, design an algorithm to find the minimum number of platforms so that all the trains can be accommodated. Example of time table:

| Trains | Arrival | Departure |
|---|---|---|
| A | 9am | 9:30am |
| B | 9:15am | 1:00pm |
| C | 10:30am | 11:00pm |
| D | 10:45am | 11:45am |

```
int findplatform(int arr[], int dep[], int n)
{
        sort(arr, arr+n)     //sort arr array which is for arrival array
        sort(dep, dep+n)     //sort dep array which is for departure array
        int needed_platform = 1, result_platform = 1;
        int i=1, j=0;
        while( i < n &&  j<n)
        {
        if(arr[i] <= dep[j])
        {
                needed_platform= needed_platform + 1;
                i++;
                //update result_platform if needed_platform  is greater than
//result_platform
                if(needed_platform > result_platform)
                        result_platform= needed_platform;
        }
```

```
        else
                // Decrement needed_platform if needed_platform  is less than or equal
                //to the result_platform
        {
                needed_platform = needed_ platform - 1;
                j++;
        }
}
return result_platform;
}
```

Solved example:

Sorted order of arr[] and dep[]

| | | | | |
|---|---|---|---|---|
| arr[] | 9:00 | 9:15 | 10:30 | 10:45 |
| dep[] | 9:30 | 11:00 | 11:45 | 13:00 |

Initially , needed_platform = 1, result_platform = 1;

1.For i at arr[1]=9:15 and j at dep[0]=9:30; j>i; increment I to 10:30 and increment result_platform which gives result_platform= 1+1=2

needed_platform < result_platform

so increment needed_platform by 1 which gives needed_platform= 1+1=2.

2. For i at arr[2]=10:30 and j at dep[0]=9:30; dep[0]<arr[2] So 1 train will depart. Increment dep[0] by 1 and decrement result_platform by 1 which gives result_platform=2-1=1.

needed_platform > result_platform

So, needed_platform remains unchanged which is needed_platform = 2

3. For i at arr[2]=10:30 and j at dep[1]=11:00; arr[2]<dep[1]. Increment arr[2] by 1.

Increment result_platform by 1 which gives result_platform=1+1=2.

Here, needed_platform = result_platform

So, needed_platform remains unchanged which is needed_platform = 2

4. For i at arr[3]=10:45 and j at dep[1]=11:00; arr[3]<dep[1].

Increment result_platform by 1 which gives result_platform=2+1=3.

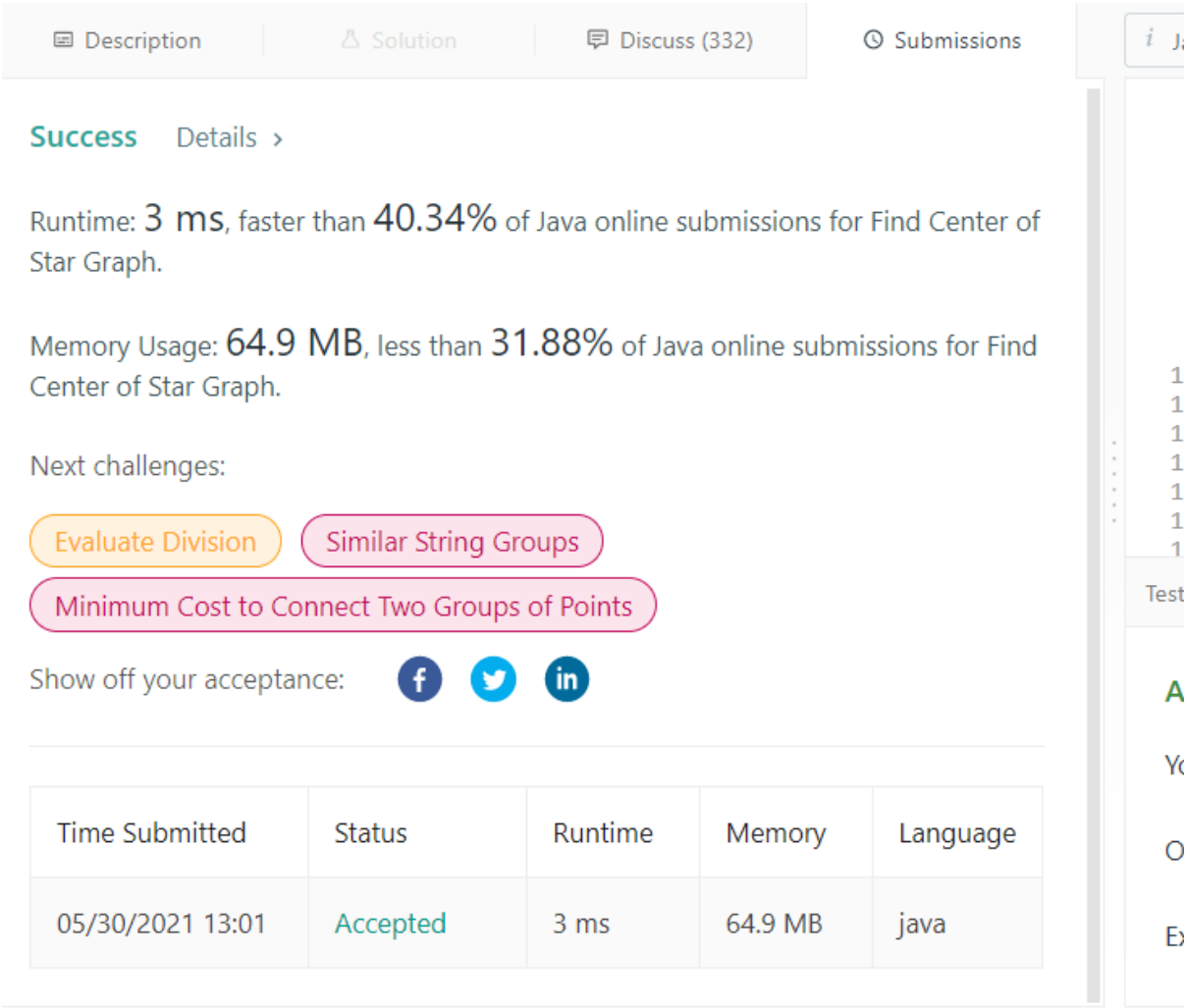
Here, needed_platform < result_platform

So, increment needed_platform by 1 which gives needed_platform=2+1=3.

All the arr[] have been traversed.

The algorithm returns result_ platform which is 3 for this example as shown above.

2. Solve leetcode problem "1791. Find Center of Star Graph" and make a successful submission.




3. Given an integer representing money amount, one problem is to use minimum number of coins (with given values) to make up this value (assuming there is unlimited number of coins).

For example, given $14, and coin system {1, 5, 10, 15, 20}, it can be changed into {10, 1,1,1,1} (one 10 and four 1s)

Do you think if this problem can be solved by Greedy Algorithm? If not, give a counter example.

Yes, this problem of coins can be solved by using greedy algorithm.

The greedy algorithm for this problem will be as following:

1. Sort the array of coins in decreasing order.
2. Initialize the result as 0.
3. Find the largest coin that is smaller than the current sum.
4. Add that coin to the result. Also, subtract the value of coin found from the amount.
5. If amount=0 then print the result.
6. Else repeat steps 3 and 4 for the new value of the total sum of the amount.

In the above example,

1. Sorted array = {20,15,10,5,1}
2. Initialize result=0
3. Largest coin is 10 which is smaller than 14. So add the 10 to the result. Subtract 10 from 14. And amount becomes amount=4.
4. Largest coin is 1 which is smaller than 4. So add the 1 to the result. Subtract 1 from 4. And amount becomes amount=3.
5. Largest coin is 1 which is smaller than 3. So add the 1 to the result. Subtract 1 from 3. And amount becomes amount=2.
6. Largest coin is 1 which is smaller than 2. So add the 1 to the result. Subtract 1 from 2. And amount becomes amount=1.
7. Largest coin is 1 which is smaller than 1. So add the 1 to the result. Subtract 1 from 1. And amount becomes amount=0.
8. Here, amount becomes 0. Then print the result.

So, the output will be {10,1,1,1,1}