Yuan Gao z5239220 Q3

You are given a time schedule of arrivals ai and departures di of n trains, so $1 \leq i \leq n$, during each 24 hour period (note: a train can arrive before the midnight and leave after midnight; each train arrives and departs at the same time every day). You need to find the minimum number of platforms so that each train can stay at a platform without interfering with other arrivals and departures.

Assume time schedule accurate to minutes

Therefore, the whole day can be seen as discrete points $[0, 1440](24hour \times 60 = 1440\text{min})$

Create a Two-dimensional array $time\_table$, every element contain two numbers which are departures and arrivals$[arrivals, departures]$.($arrivals, departures \in [0, 1440]$)

Step 1:

Separate the schedule which across midnight to the two schedules.

$[arrivals, 1440]$ and $[0, departures]$

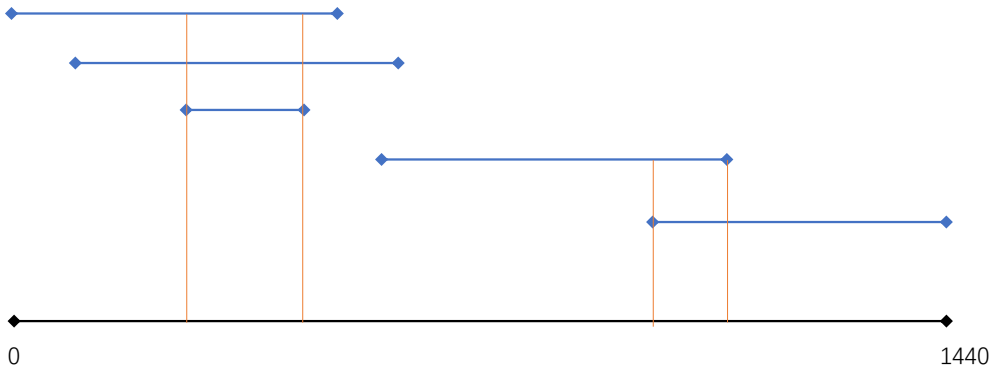
Convert a cycle to a line

(if $arrivals > departures$ means across midnight)

Step2:

Sort $time\_table$ array based on $arrivals$ time

Step3:



We need to calculate the maximum number of trains stay in platform at same time. Only in this way can there be no conflicts.

It makes sure every train has platform to stop

Code:

```
1.  int findMinplatform(vector<vector<int>>& timeTable){
2.      if(timeTable.size() < 1) return timeTable.size();
3.
4.      //Separate the schedule which across midnight to the two schedules.
5.      auto i = timeTable.begin();
6.      while(i != timeTable.end()){
7.          if((*i)[0] > (*i)[1]){
8.              vector<int> b = {0, (*i)[1]};
```

```
            (*i)[1] = 1044;
            i = timeTable.insert(i + 1, b);
        }else{
            ++i;
        }
    }

    //Sort timeTable array based on departures time
    sort(timeTable.begin(), timeTable.end(), [](vector<int> &a, vector<int> &b){
                                            return a[0] < b[0];
    });

    //calculate the maximum number of trains stay in platform at same time
    //initialize the first interval
    int left = timeTable[0][0];
    int right = timeTable[0][1];
    //initialize the result and tmp
    //res is the minimum number of platforms
    int res = 1;
    //tmp is count the maximum number of trains stay in platform at same time
    int tmp = 1;

    for(int i = 1; i < timeTable.size(); ++i){
        //if current departures time not in the interval
        //picked max(res, tmp)
        if(right < timeTable[i][0]){
            left = timeTable[i][0];
            right = timeTable[i][1];
            res = max(res, tmp);
            tmp = 1;
        }else{
            //if current departures time in the interval
            //shrink the interval
            if(left < timeTable[i][0]){
                left = timeTable[i][0];
            }
            if(right > timeTable[i][1]){
                right = timeTable[i][1];
            }
            //the number of trains stay in platform at same time +1
            ++tmp;
        }
    }
    return res;
```

```
53. }
```