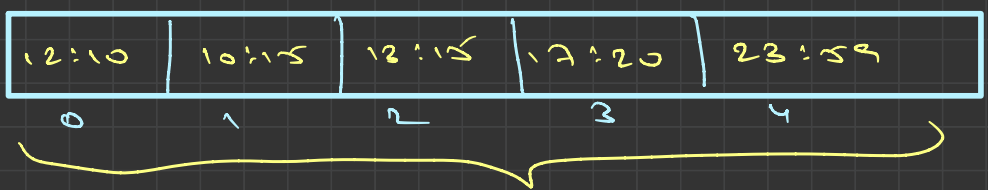
Minimum Time Difference — 539

Given a list of 24 hr clock time points in 'HH:mm' format, return the minimum minutes diff b/w any two time points in the list.



for Ex

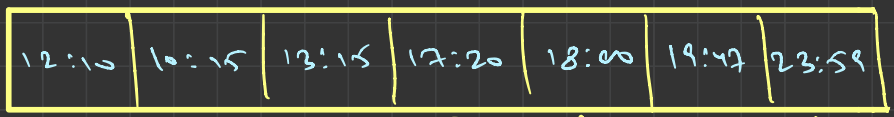
12:10 → 13:10

Diff → 1:00 → 60 min

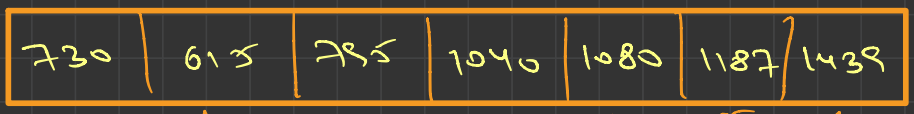
→ Convert whole string into → minutes

Stoi → Convert string into integer

* make sure we calculated time from 12:00

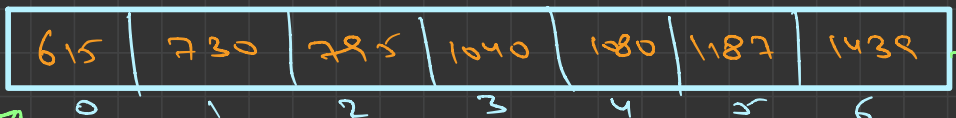


⇓ minutes



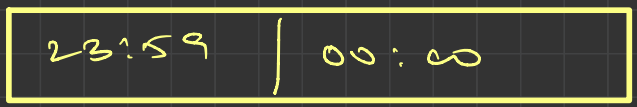
⇒ find min diff

to min T.C.
$O(n^2) \rightarrow O(n \log n)$ // Sort ? ⇒ now we dont check each ele with other ele



get difference → min

Compare this also
↓
Edge Case



↓ min

↓ sort

↓

Diff => 1439

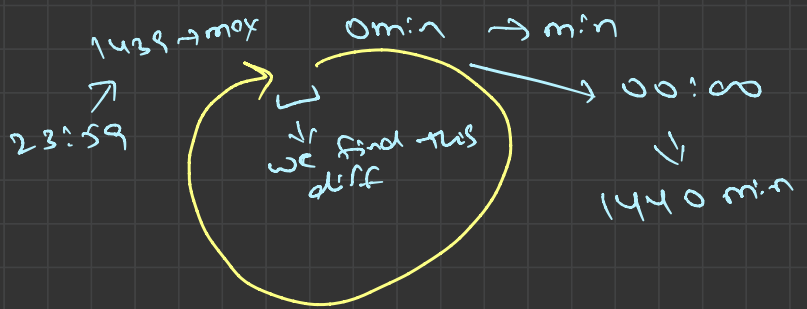
Exact ans → 1 min

↓

How? cyclic clock

↓

Time is circular



Can we replace 0 → 1440 ?

1 circle of clock => 24 hr → 1440 min

↓

after 24hr the cycle will restart

```
class Solution {
public:
    int findMinDifference(vector<string>& timePoints) {
        vector<int> minutes;

        // convert string to min and push in vector
        for(int i = 0; i < timePoints.size(); i++){
            string curr = timePoints[i];

            // fetch hour
            // stoi -> convert string into number
            int hr = stoi(curr.substr(0,2));
            int min = stoi(curr.substr(3,2));

            int totalMin = hr * 60 + min;
            minutes.push_back(totalMin);
        }

        // to reduce T.C. n^2 -> nlogn
        sort(minutes.begin(),minutes.end());

        // find difference and cal min diff
        int mini = INT_MAX;

        int n = minutes.size();
        for(int i = 0; i < n-1; i++){
            int diff = minutes[i+1] - minutes[i];
            mini = min(mini, diff);
        }

        // edge case
        int lastDiff = (minutes[0] + 1440) - minutes[n-1];
        mini = min(mini, lastDiff);

        return mini;
    }
```