# Minimum Time Difference

Given a list of 24-hour clock time points in "Hour:Minutes" format, find the minimum **minutes** difference between any two time points in the list.

### Example 1:

```
Input: ["23:59","00:00"]
Output: 1
```

### Note:

1. The number of time points in the given list is at least 2 and won't exceed 20000.
2. The input time is legal and ranges from 00:00 to 23:59.

## Solution 1

There is only 24 * 60 = 1440 possible time points. Just create a boolean array, each element stands for if we see that time point or not. Then things become simple...

```
public class Solution {
    public int findMinDifference(List<String> timePoints) {
        boolean[] mark = new boolean[24 * 60];
        for (String time : timePoints) {
            String[] t = time.split(":");
            int h = Integer.parseInt(t[0]);
            int m = Integer.parseInt(t[1]);
            if (mark[h * 60 + m]) return 0;
            mark[h * 60 + m] = true;
        }

        int prev = 0, min = Integer.MAX_VALUE;
        int first = Integer.MAX_VALUE, last = Integer.MIN_VALUE;
        for (int i = 0; i < 24 * 60; i++) {
            if (mark[i]) {
                if (first != Integer.MAX_VALUE) {
                    min = Math.min(min, i - prev);
                }
                first = Math.min(first, i);
                last = Math.max(last, i);
                prev = i;
            }
        }

        min = Math.min(min, (24 * 60 - last + first));

        return min;
    }
}
```

written by shawngao original link here

## Solution 2

```
public class Solution {
    public int findMinDifference(List<String> timePoints) {
        int n = timePoints.size();
        List<Time> times = new ArrayList<>();
        for (String tp : timePoints) {
            String[] strs = tp.split(":");
            times.add(new Time(Integer.parseInt(strs[0]), Integer.parseInt(strs[1]))
);
        }
        Collections.sort(times);
        Time earlist = times.get(0);
        times.add(new Time(earlist.h + 24, earlist.m));
        int minDiff = Integer.MAX_VALUE;
        for (int i = 0; i < n; i++) {
            int diff = (int) Math.abs(times.get(i).getDiff(times.get(i + 1)));
            minDiff = Math.min(minDiff, diff);
        }
        return minDiff;
    }

}

class Time implements Comparable<Time> {
    int h;
    int m;
    public Time(int h, int m) {
        this.h = h;
        this.m = m;
    }

    public int compareTo(Time other) {
        if (this.h == other.h) {
            return this.m - other.m;
        }
        return this.h - other.h;
    }

    public int getDiff(Time other) {
        return (this.h - other.h) * 60 + (this.m - other.m);
    }
}
```

written by zzwcsong original link here

## Solution 3

Convert each timestamp to it's integer number of minutes past midnight, and sort the array of minutes.
The required minimum difference must be a difference between two adjacent elements in the circular array (so the last element is "adjacent" to the first.) We take the minimum value of all of them.

```
def findMinDifference(self, A):
    def convert(time):
        return int(time[:2]) * 60 + int(time[3:])
    minutes = map(convert, A)
    minutes.sort()

    return min( (y - x) % (24 * 60)
                for x, y in zip(minutes, minutes[1:] + minutes[:1]) )
```

written by awice original link here