<https://leetcode.com/problems/determine-if-two-events-have-conflict/description/>

You are given two arrays of strings that represent two inclusive events that happened **on the same day**, event1 and event2, where:

* event1 = [startTime1, endTime1] and
* event2 = [startTime2, endTime2].

Event times are valid 24 hours format in the form of HH:MM.

A **conflict** happens when two events have some non-empty intersection (i.e., some moment is common to both events).

Return true *if there is a conflict between two events. Otherwise, return* false.

**Example 1:**

Input: event1 = ["01:15","02:00"], event2 = ["02:00","03:00"]

Output: true

Explanation: The two events intersect at time 2:00.

**Example 2:**

Input: event1 = ["01:00","02:00"], event2 = ["01:20","03:00"]

Output: true

Explanation: The two events intersect starting from 01:20 to 02:00.

**Example 3:**

Input: event1 = ["10:00","11:00"], event2 = ["14:00","15:00"]

Output: false

Explanation: The two events do not intersect.

**Constraints:**

* evnet1.length == event2.length == 2.
* event1[i].length == event2[i].length == 5
* startTime1 <= endTime1
* startTime2 <= endTime2
* All the event times follow the HH:MM format.

**Attempt 1: 2023-12-02**

**Solution 1: Sweep Line (10 min)**

class Solution {

public boolean haveConflict(String[] event1, String[] event2) {

int[] timeline = new int[60 \* 24 + 1];

String[] e11 = event1[0].split(":");

String[] e12 = event1[1].split(":");

String[] e21 = event2[0].split(":");

String[] e22 = event2[1].split(":");

int a = Integer.parseInt(e11[0]) \* 60 + Integer.parseInt(e11[1]);

int b = Integer.parseInt(e12[0]) \* 60 + Integer.parseInt(e12[1]);

int c = Integer.parseInt(e21[0]) \* 60 + Integer.parseInt(e21[1]);

int d = Integer.parseInt(e22[0]) \* 60 + Integer.parseInt(e22[1]);

timeline[a]++;

timeline[c]++;

timeline[b + 1]--;

timeline[d + 1]--;

int count = 0;

for(int t : timeline) {

count += t;

if(count > 1) {

return true;

}

}

return false;

}

}

Time Complexity: O(N)

Space Complexity: O(N)

**Solution 2: Intersection detect (10 min)**

class Solution {

public boolean haveConflict(String[] event1, String[] event2) {

String[] e11 = event1[0].split(":");

String[] e12 = event1[1].split(":");

String[] e21 = event2[0].split(":");

String[] e22 = event2[1].split(":");

int a = Integer.parseInt(e11[0]) \* 60 + Integer.parseInt(e11[1]);

int b = Integer.parseInt(e12[0]) \* 60 + Integer.parseInt(e12[1]);

int c = Integer.parseInt(e21[0]) \* 60 + Integer.parseInt(e21[1]);

int d = Integer.parseInt(e22[0]) \* 60 + Integer.parseInt(e22[1]);

if(b < c || a > d) {

return false;

}

return true;

}

}

Time Complexity: O(1)

Space Complexity: O(1)

**Refer to**

<https://leetcode.com/problems/determine-if-two-events-have-conflict/solutions/2734145/line-sweep-learn-this-for-all-time-day-problems/>

This technique wasn't necessary at all, but it works with many time/day problems. It's easy to implement too.

Explanation:

* The HH:MM is converted in to a line sweep of only minutes. (24\*\60 mins array)
* We add the event [start\_time] and [end\_time +1] for both events.
* If two events were to conflict, the value in line sweep will be > 1.

**Method 1 - Line Sweep**

bool haveConflict(vector<string>& e1, vector<string>& e2) {

vector<int> times(16000);

int t1start = stoi(e1[0].substr(0,2)) \* 60 + stoi(e1[0].substr(3));

int t1end = stoi(e1[1].substr(0,2)) \* 60 + stoi(e1[1].substr(3));

times[t1start] += 1;

times[t1end+1] -= 1;

int t2start = stoi(e2[0].substr(0,2)) \* 60 + stoi(e2[0].substr(3));

int t2end = stoi(e2[1].substr(0,2)) \* 60 + stoi(e2[1].substr(3));

times[t2start] += 1;

times[t2end+1] -= 1;

int cnt = 0;

for(int t: times){

cnt += t;

if(cnt > 1)

return true;

}

return false;

}

**Method 2 - Naive**

* Yes, of course line sweep isn't needed.

bool haveConflict(vector<string>& e1, vector<string>& e2) {

int t1start = stoi(e1[0].substr(0,2)) \* 60 + stoi(e1[0].substr(3));

int t1end = stoi(e1[1].substr(0,2)) \* 60 + stoi(e1[1].substr(3));

int t2start = stoi(e2[0].substr(0,2)) \* 60 + stoi(e2[0].substr(3));

int t2end = stoi(e2[1].substr(0,2)) \* 60 + stoi(e2[1].substr(3));

if(t2start <= t1end && t2end >= t1start)

return true;

return false;

}