**Leetcode problem attend meeting**

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

bool canAttendMeetings(vector<vector<int>> &time) {

 sort(time.begin(),time.end(),[](vector<int> &a,vector<int> &b) {

    return a[0]<b[0];

   });

 for (int i=1;i<time.size();i++) {

  if (time[i][0]<time[i-1][1]){

   return false;

  }

 }

 return true;

}

int main(){

 vector<vector<int>>time={{7,10},{2,4}};

 string a=(canAttendMeetings(time)?"true":"false");

 cout<<"Output: "<<a;

 return 0;

}

----------

#include <iostream>

#include <vector>

#include <algorithm>

#include<bits/stdc++.h>

using namespace std;

struct Interval {

 int start;

 int end;

};

bool compareIntervals(const Interval& a, const Interval& b) {

 return a.start < b.start;

}

bool canAttendMeetings(vector<Interval>& intervals) {

 sort(intervals.begin(), intervals.end(), compareIntervals);

 for(auto i: intervals){

  printf("%d",i);

 }

 cout<<endl;

 for (int i = 1; i < intervals.size(); ++i) {

  if (intervals[i].start < intervals[i - 1].end) {

   return false; // Overlapping meetings

  }

 }

 return true; // No overlaps

}

int main() {

 vector<Interval> meetings = {{0, 30}, {5, 10}, {15, 20}};

 bool result = canAttendMeetings(meetings);

 cout << (result ? "Can attend all meetings" : "Cannot attend all meetings") << endl;

 return 0;

}

----------

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

bool canAttendMeetings(vector<vector<int>>& intervals) {

 sort(intervals.begin(),intervals.end(),[](const vector<int>& a,const vector<int>& b){

  return a[0]<b[0];

 });

 for (int i=1;i<intervals.size();i++){

  if (intervals[i - 1][1]>intervals[i][0]){

   return false;

  }

 }

 return true;

}

int main(){

 vector<vector<int>> time={{0,30},{5,10},{15,20}};

 if(canAttendMeetings(time))

  cout<<"true"<<endl;

 else

  cout << "false" << endl;

   vector<vector<int>> time1={{7,10},{2,4}};

   if(canAttendMeetings(time1))

  cout<<"true"<<endl;

 else

  cout << "false" << endl;

 return 0;

}

------------

#include<iostream>

#include<vector>

#include<algorithm>

using namespace std;

vector<vector<int>>nums={{0,30},{5,10},{15,20}};

int main(){

 int flag;

 for(int i=0;i<nums.size();i++){

 if(nums[i].end()>nums[i+1].begin()){

  flag=1;

  break;

 }

}

if(flag==1){

 cout<<"False"<<endl;

}

else{

 cout<<"True";

}

return 0;

}

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

bool canAttendMeetings(vector<vector<int>> &time) {

 sort(time.begin(),time.end(),[](vector<int> &a,vector<int> &b) {

  return a[0]<b[0];

  });

 for (int i=1;i<time.size();i++) {

 if (time[i][0]<time[i-1][1]){

  return false;

 }

 }

 return true;

}

int main(){

 vector<vector<int>>time={{7,10},{2,4}};

 string a=(canAttendMeetings(time)?"true":"false");

 cout<<"Output: "<<a;

 return 0;

}

----------

#include <iostream>

#include <vector>

#include <algorithm>

#include<bits/stdc++.h>

using namespace std;

struct Interval {

 int start;

 int end;

};

bool compareIntervals(const Interval& a, const Interval& b) {

 return a.start < b.start;

}

bool canAttendMeetings(vector<Interval>& intervals) {

 sort(intervals.begin(), intervals.end(), compareIntervals);

 for(auto i: intervals){

 printf("%d",i);

 }

 cout<<endl;

 for (int i = 1; i < intervals.size(); ++i) {

 if (intervals[i].start < intervals[i - 1].end) {

  return false; // Overlapping meetings

 }

 }

 return true; // No overlaps

}

int main() {

 vector<Interval> meetings = {{0, 30}, {5, 10}, {15, 20}};

 bool result = canAttendMeetings(meetings);

 cout << (result ? "Can attend all meetings" : "Cannot attend all meetings") << endl;

 return 0;

}

----------

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

bool canAttendMeetings(vector<vector<int>>& intervals) {

 sort(intervals.begin(),intervals.end(),[](const vector<int>& a,const vector<int>& b){

 return a[0]<b[0];

 });

 for (int i=1;i<intervals.size();i++){

 if (intervals[i - 1][1]>intervals[i][0]){

  return false;

 }

 }

 return true;

}

int main(){

 vector<vector<int>> time={{0,30},{5,10},{15,20}};

 if(canAttendMeetings(time))

 cout<<"true"<<endl;

 else

 cout << "false" << endl;

  vector<vector<int>> time1={{7,10},{2,4}};

  if(canAttendMeetings(time1))

 cout<<"true"<<endl;

 else

 cout << "false" << endl;

 return 0;

}

------------

#include<iostream>

#include<vector>

#include<algorithm>

using namespace std;

vector<vector<int>>nums={{0,30},{5,10},{15,20}};

int main(){

 int flag;

 for(int i=0;i<nums.size();i++){

 if(nums[i].end()>nums[i+1].begin()){

 flag=1;

 break;

 }

}

if(flag==1){

 cout<<"False"<<endl;

}

else{

 cout<<"True";

}

return 0;

}