**Powerfull Integer :-**

Easy Accuracy: 51.91% Submissions: 12K+ Points: 2

You are given a **2D integer** array of intervals whose length is **n** where **intervals[i]=[start, end]** I.e. all integers from start to end**inclusive** of start and endare also present and also we are given an integer **k**. We have to return the **Powerfull Integer**. A powerful Integer is an integer that occurs at **least k** times. If multiple integers have at least **k occurrences,** we have to return the **maximum integer** out of all those elements.

**Note**: If no integer occurs at least k times return **-1**.

**Example 1:**

**Input :**

n=3

intervals={{1,3},{4,6},{3,4}}

k=2

**Output:** 4

**Explanation:**

As we can see that 3 and 4 are the 2 integers

that have 2 occurences(2>=k) so we have 4

in this case as the Powerfull integer as it

is the maximum element which satisfies the condition.

**Example 2:**

**Input :**

n=4

intervals={{1,4},{12,45},{3,8},{10,12}}

k=3

**Output:** -1

**Explanation:**

As we can see that no integer occurs

3 times so in that case we have to

return -1 (see Note).

**Example 3:**

**Input :**

n=5

intervals={{16,21},{5,8},{12,17}

  {17,29},{9,24}}

k=3

**Output:**

**21**

**Explanation:**

As we can see in the given **range** 17

occured 4 times while 16,18,19,20 & 21 occured

3 times. So here we need 3 I.e K occurences

so we can choose any but we will pick **21**,

because it is maximum.

**Your Task:**  
You don't need to read input or print anything. Your task is to complete the function **powerfullInteger**() which takes an integer**n,**a **2d array intervals**and an integer **k** respectively and you have to return powerful Integer if it exists else return **-1**.

**Expected Time Complexity:** O(NlogN)  
**Expected Space Complexity:** O(N)

**Constraints:**  
1<=**n**<=105  
1<=**intervals**[i][0]<=**intervals**[i][1]<=109  
1<=**k**<=105  
The sum of n over all test cases won't exceed **106**

**Code :-**

//{ Driver Code Starts

//Initial Template for C++

#include<bits/stdc++.h>

using namespace std;

// } Driver Code Ends

//User function Template for C++

class Solution{

public:

int powerfullInteger(int n,vector<vector<int>> &intervals,int k){

map<long long, int> mp;

for(auto v:intervals){

mp[v[0]]++;

mp[v[1]+1]--;

}

int ans=-1, psum=0;

for(auto p:mp){

if(p.second < 0){

if(psum >= k)

ans = p.first - 1;

}

psum += p.second;

if(psum >= k)

ans = p.first;

}

return ans;

}

};

//{ Driver Code Starts.

int main(){

int t;

cin>>t;

while(t--){

int n,k;

cin>>n;

vector<vector<int>> intervals(n,vector<int>(2));

for(int i=0;i<n;i++){

cin>>intervals[i][0]>>intervals[i][1];

}

cin>>k;

Solution ob;

cout<<ob.powerfullInteger(n,intervals,k)<<endl;

}

return 0;

}

// } Driver Code Ends

**Logic :-**

Used Prefix sum technology