Maximum Activities

Approach :

Same as N meetings in a room.

```
#include <bits/stdc++.h>
#define PII pair<pair<int,int>,int>
class Compare
{
public:
    bool operator()(PII a, PII b)
    {
        if(a.first.first>b.first.first)
        {
            return true;
        }
        else if(a.first.first==b.first.first)
        {
            return a.second>b.second;
        }
        return false;

    }
};

int maximumActivities(vector<int> &start, vector<int> &end) {
    // Write your code here.
    priority_queue<PII,vector<PII>,Compare> meet;
    for(int i=1;i<=start.size();i++)
    {
      meet.push({{end[i - 1], start[i - 1]}, i});
    }
    auto top=meet.top();
    int ans=1;
    int endlimit=top.first.first;
    meet.pop();

    while(!meet.empty())
    {
        auto t=meet.top();
        meet.pop();
        // cout<<t.first.second<<" "<<t.first.first<<endl;
        if(t.first.second>=endlimit)
        {
            ans++;
            endlimit=t.first.first;
        }
    }
```

```
    return ans;
}
```