LAB REPORT

Submitted by

Stuti Jain [RA2011031010031]

Under the Guidance of

Dr. V.R.Balasaraswathi

Assistant Professor
Department of NWC

In partial satisfaction of the requirements for the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE ENGINEERING

with specialization in Information Technology

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

JUNE 2022

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR-603203

BONAFIDE CERTIFICATE

Certified that this lab report titled **“DAA mini project”** is the bonafide work done by Stuti Jain(RA2011031010031) who carried out the lab exercises under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other work.

SIGNATURE
Dr. V.R.Balasaraswathi
DAA – Course Faculty
Assistant Professor
Department of NWC

A MINI PROJECT REPORT

Submitted in fulfilment of the

requirement of DAA

By

- RA20110031010065 VANI KUMAR
- RA2011031010031 STUTI JAIN

Under the Guidance of **Dr.V.R.Balasaraswathi**

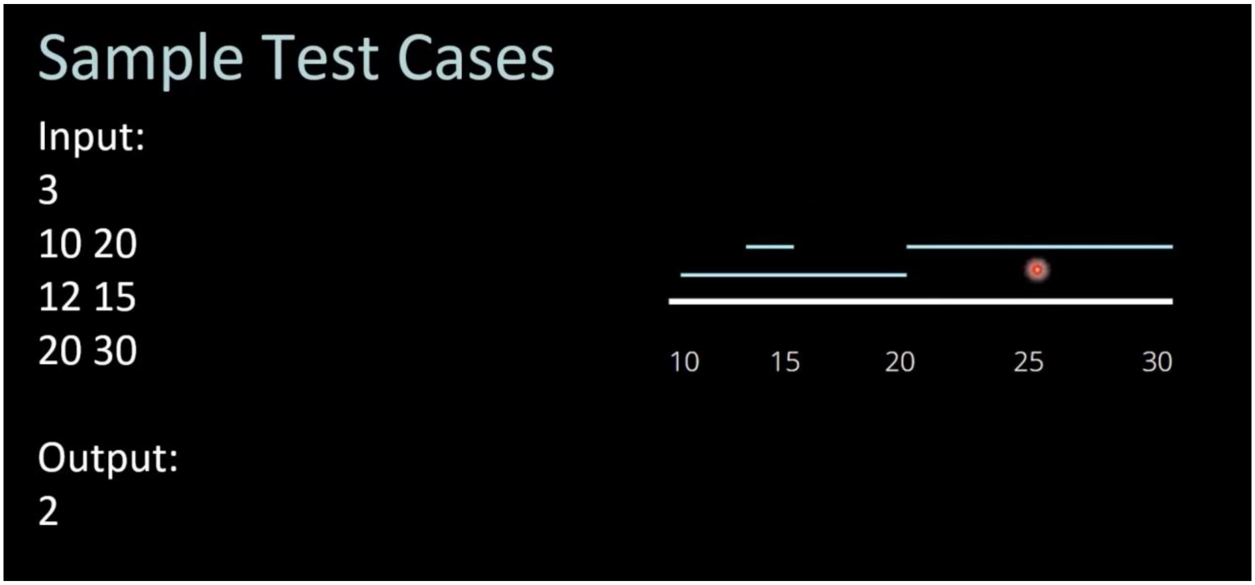
Assistant Professor

Department of NWC

SRM Institute of Science and Technology, Kattankulathur

Problem

You are given n activities with their start and finish times. Select the maximum number of activities that can be performed by a single person, assuming that a person can only work on a single activity at a time.



Sample Test Cases

Input:

```
3
10 20
12 15
20 30
```

Output:

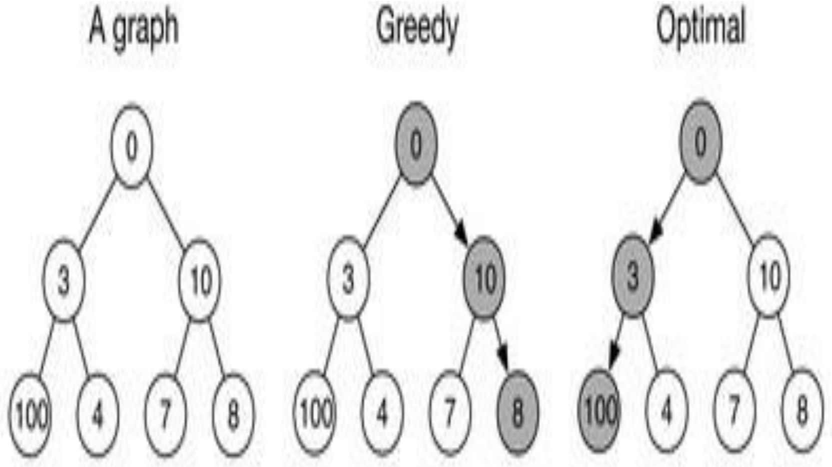
```
2
```

Description

The activity selection problem is a mathematical optimization problem. Our first illustration is the problem of scheduling a resource among several challenge activities. We find a greedy algorithm provides a well designed and simple method for selecting a maximum- size set of manually compatible activities.

Greedy Algorithm

Greedy is an algorithmic paradigm that builds up a solution piece by piece, always choosing the next piece that offers the most obvious and immediate benefit. Greedy algorithms are used for optimization problems. An optimization problem can be solved using Greedy if the problem has the following property: At every step, we can make a choice that looks best at the moment, and we get the optimal solution of the complete problem.



A greedy algorithm fails to maximise the sum of nodes along a path from the top to the bottom because it lacks the foresight to choose suboptimal solutions in the current iteration that will allow for better solutions later

Code:

```
#include<bits/stdc++.h>

using namespace std;

int main(){ int n; cin>>n;

        vector<vector<int>>

        v;

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

        int start, end;

        cin>>start>>end;

        v.push_back({start,end});

        }

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

        return a[1] <b[1];

        });

        int take = 1; int

        end = v[0][1];

        for(int i

        =1;i<n;i++){

        if(v[i][0] >=end){

        take++;

        end = v[i][1];

        }

        }
        cout<<take<<endl;
         return 0;
}
```

Sample Test Cases:

Input:

3

10 20

12 15

20 30

Output:

2

Constraints:

1 <= n <= 1e5 1

<=start, end <= 1e9

start <= end

Complexity Analysis

For unsorted inputs : O(NlogN)

For sorted inputs : O(N)

Conclusion

At the end we were able to figure out the solution to the problem statement, the inputs were sorted and subsequently we got an output, in our sample case.