Bài toán tìm các khoảng không giao nhau

**I. Bài toán**

Có n công việc cần thực hiện; ai – thời điểm bắt đầu, bi – thời điểm kết thúc công việc i (i=1...n)

Hãy chọn ra các công việc để một người có thể thực hiện được nhiều việc nhất.

**II. Mô tả chi tiết thuật toán**

1. Ý tưởng (tham lam)

- Gọi C là tập các công việc ban đầu

- Gọi S là tập các công việc được lựa chọn

- Sắp xếp các công việc theo thứ tự không giảm của đầu mút phải (bi)

- Lần lượt xét các đoạn trong danh sách theo thứ tự đã sắp xếp và bổ sung đoạn thẳng đang xét vào S nếu nó không có điểm chung với bất cứ đoạn nào trong S.

2. Chi tiết

**procedure Greedy3;**

**begin**

S:=; (\* S là tập các đoạn thẳng cần tìm \*)

<Sắp xếp các đoạn trong C theo thứ tự không giảm của mút phải>

**while** (C ≠) **do**

**begin**

đoạn đầu tiên trong C;

C:=C \ (;

**if** <( không giao với bất cứ đoạn nào trong S>

**then** S:= S

**end**;

<**S** là tập cần tìm>

**end**;

**III. Thực hiện thuật toán**

\*Bộ dữ liệu 1

|  |  |  |
| --- | --- | --- |
| Công việc | Bắt đầu | Kết thúc |
| a1 | 2 | 3 |
| a2 | 4 | 6 |
| a3 | 1 | 7 |
| a4 | 9 | 10 |
| a5 | 5 | 12 |
| a6 | 8 | 14 |
| a7 | 13 | 16 |
| a8 | 11 | 15 |
| a9 | 3 | 8 |
| a10 | 15 | 20 |

Sắp xếp các công việc theo thứ tự không giảm của mút phải ta được thứ tự các công việc

C = {a1, a2, a3, a9, a4, a5, a6, a8, a7, a10}

Khởi tạo:

C = {a1, a2, a3, a9, a4, a5, a6, a8, a7, a10}

S= {}

Lặp:

Chọn công việc a1

S = {a1}

Xét công việc a2 không có đoạn chung với a1 -> chọn a2

S = {a1, a2}

Xét công việc a3 có đoạn chung với a2 -> không chọn a3

S = {a1, a2}

Xét công việc a9 có đoạn chung với a2 -> không chọn a9

S = {a1, a2}

Xét công việc a4 không có đoạn chung với a2 -> chọn a4

S = {a1, a2, a4}

Xét công việc a5 có đoạn chung với a4 -> không chọn a5

S = {a1, a2, a4}

Xét công việc a6 có đoạn chung với a4 -> không chọn a6

S = {a1, a3, a4}

Xét công việc a8 không có đoạn chung với a4 -> chọn a8

S = {a1, a2, a4, a8}

Xét công việc a7 có đoạn chung với a8 -> không chọn a7

S = {a1, a2, a4, a8}

Xét công việc a10 không có đoạn chung với a8 -> chọn a10

S = {a1, a2, a4, a8, a10}

\*Bộ 2:

|  |  |  |
| --- | --- | --- |
| Công việc | Bắt đầu | Kết thúc |
| a1 | 1 | 3 |
| a2 | 4 | 6 |
| a3 | 1 | 7 |
| a4 | 11 | 15 |
| a5 | 5 | 12 |
| a6 | 8 | 14 |
| a7 | 13 | 16 |
| a8 | 11 | 15 |
| a9 | 3 | 9 |
| a10 | 18 | 25 |

Bộ 3:

|  |  |  |
| --- | --- | --- |
| Công việc | Bắt đầu | Kết thúc |
| a1 | 2 | 3 |
| a2 | 4 | 6 |
| a3 | 1 | 7 |
| a4 | 9 | 10 |
| a5 | 5 | 12 |
| a6 | 8 | 14 |
| a7 | 15 | 16 |
| a8 | 1 | 15 |
| a9 | 3 | 9 |
| a10 | 17 | 30 |

Bộ 4:

|  |  |  |
| --- | --- | --- |
| Công việc | Bắt đầu | Kết thúc |
| a1 | 2 | 3 |
| a2 | 4 | 6 |
| a3 | 1 | 7 |
| a4 | 2 | 10 |
| a5 | 5 | 12 |
| a6 | 9 | 19 |
| a7 | 1 | 16 |
| a8 | 11 | 15 |
| a9 | 3 | 10 |
| a10 | 20 | 25 |

Bộ 5:

|  |  |  |
| --- | --- | --- |
| Công việc | Bắt đầu | Kết thúc |
| a1 | 2 | 3 |
| a2 | 4 | 6 |
| a3 | 1 | 7 |
| a4 | 9 | 10 |
| a5 | 5 | 12 |
| a6 | 8 | 14 |
| a7 | 11 | 16 |
| a8 | 1 | 15 |
| a9 | 3 | 8 |
| a10 | 15 | 20 |

**IV. Chương trình sử dụng C++**

#include <iostream>

#include <fstream>

#include <conio.h>

#include <sstream>

using namespace std;

#define max 100

struct Activity {

string id = "a";

int start;

int finish;

} Plan[100];

int n;

void setPlan();

void Init();

int main()

{

Init();

setPlan();

return 0;

}

// Nhap tu file

void Init(){

fstream in;

in.open("input.txt");

in >> n; // Doc so hoat dong

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

stringstream ss;

string str;

ss << i+1;

ss >> str;

Plan[i].id += str;

in >> Plan[i].start >> Plan[i].finish; // Doc khoang thoi gian

}

}

// Chon hoat dong

void setPlan()

{

int i, j;

Activity temp;

//step 1

//sort the Plan as per finishing time in ascending order

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

for (j = 0; j < n - 1; j++) {

if (Plan[j].finish > Plan[j + 1].finish) {

temp = Plan[j];

Plan[j] = Plan[j + 1];

Plan[j + 1] = temp;

}

}

}

//sorted

cout << "Sorted Plan as per finish time (ascending order)\n";

cout << "Activity" << "\tStart" << "\tFinish" << endl;

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

cout << "\t" << Plan[i].id << "\t" << Plan[i].start << "\t" << Plan[i].finish << endl;

}

//step 2

//select the first activity

cout << "-----Selected Plan-----\n";

cout << "Activity" << "\tStart" << "\tFinish" << endl;

cout << "\t" << Plan[0].id << "\t" << Plan[0].start << "\t" << Plan[0].finish << endl;

//step 3

//select next activity whose start time is greater than

//or equal to the finish time of the previously selected activity

i = 0;

for (j = 1; j < n; j++) {

if (Plan[j].start >= Plan[i].finish) {

cout << "\t" << Plan[j].id << "\t" << Plan[j].start << "\t" << Plan[j].finish << endl;

i = j;

}

}

}

**V. Đánh giá độ phức tạp của thuật toán**