CS29003 ALGORITHMS LABORATORY
Greedy Algorithms
Last Date of Submission: 4 – October – 2018

General Instruction

1. Please do not use any global variable unless you are explicitly instructed so.
2. Please use proper indentation in your code.
3. Please name your file as `<roll_no>_<assignment_no>`. For example, if your roll number is 14CS10001 and you are submitting assignment 3, then name your file as `14CS10001_3.c or 14CS10001_3.cpp` as applicable.
4. Please write your name and roll number at the beginning of your program.

An information technology firm undertakes a big project. The project requires the company to hire professionals with various categories of expertise. The higher management analyzed the project and found it to consist of K subparts. Each subpart requires a group of professionals specially skilled for that subpart. Each such group of professionals takes a certain number of days.

The whole work is looked over by a single consultant. He can supervise only one subpart at a time; hence only one group of skilled professionals can be deployed at a time. However, professionals with such specialized skills may not be readily available all the time. In order to avoid delays due to non-availability of required professionals when needed, the company decides to hire all the professionals at the start of the project itself, paying them from the beginning until their subpart is done, at which point, they are let go. Each group of professionals costs the company a certain amount of money per day from the day they are hired till they finish their subpart.

The company wants to schedule the tasks so as to minimize its expenditure. In Part-I, we assume that the subparts have the advantage that they can be done in any order. So the company is at complete liberty to order the tasks at its will. You will help the company to pick an order that minimizes its total expenditure.

In part-II, the situation is compounded by the fact that there are dependencies between subparts. A subpart p_2 is dependent on a subpart p_1 if p_2 can begin only after p_1 ends. However it is observed that each subpart can depend on at most one subpart and can have at most one subpart dependent on it. As before, you will help the company to pick an order that minimizes its total expenditure subject to these *precedence constraints*.

Part-I

Represent information about each subpart by the following structure:

```
typedef struct{
        int subpartID;
        int cost_per_day;
        int duration;
```

}subpart_data;

Write a function with following prototype that takes in the number K of subparts and an array A of information about the subparts, and prints the total cost of an optimal schedule. Assume that in the array A, the **subpartID** field of each structure holds the index of the structure.

void print_schedule(subpart_data *A, **int** K**);**

The above function must be implemented by a greedy algorithm that runs in $O(K \log K)$ time.

Part-II

Store dependence informations in structures of the following type:

```
typedef struct{
        int predecessorID;
        int successorID;
}dependency_info;
```

Store the data such that subpart **successorID** is dependent on subpart **predecessorID**. Write a function with following prototype that takes in the number K of subparts, an array A of information about the subparts, the number ℓ of dependencies, an array B of dependency information, and prints the total cost of an optimal schedule subject to the precedence constraints. Like Part-I, identifier of a subpart is simply its index in the array A.

void print_schedule1(subpart_data *A, **int** K, **dependency_info** *B, **int** ℓ**);**

The above function must be implemented by a greedy algorithm that runs in $O(K^2)$ time. Note that if each subpart can depend on at most one subpart and can have at most one subpart dependent on it, then $\ell < K$.

main

Enter subpart information in the following format:

- ▷ The first line is the number of subparts (K). Assume that $K \leqslant 100$.
- ▷ The second line contains a sequence of K integers indicating the duration of each subpart.
- ▷ The third line contains a sequence of K integers indicating the hiring cost per day of each subpart.

Run **print_schedule** and print the cost of an optimal schedule.
Enter dependency inputs (for part-II) in the following format:

- ▷ The first line contains the number of dependencies ℓ.
- ▷ Then there are ℓ lines, each containing two integers x and y (in this order) where subpart y depends on subpart x (i.e., y can start only after x has finished).

Run **print_schedule1** to print the cost of an optimal schedule respecting precedence constraints.

Sample input and output

Note: The sample outputs are not necessarily the optimal costs. Please refer to them only for the format.

```
5
21 27 12 33 5
100 200 50 25 75
cost=24375
3
1 2
3 5
5 4
cost=21125
```