Project Assignment 2	
Course: Algorithm Design and Analysis	**Semester**: Spring 2024
Instructor: Shi Li	**Due Date**: **2024/6/23**

Problem 1. An independent set of a graph $G = (V, E)$ is a set $U \subseteq V$ of vertices such that there are no edges between vertices in U. Given a graph with node weights, the maximum-weight independent set problem asks for the independent set of a given graph with the maximum total weight. In general, this problem is NP-hard.

For this programming problem, you need to solve the problem on trees: given a tree with node weights, find the independent set of the tree with the maximum total weight. For example, the maximum-weight independent set of the tree in Figure 1 has weight 47.

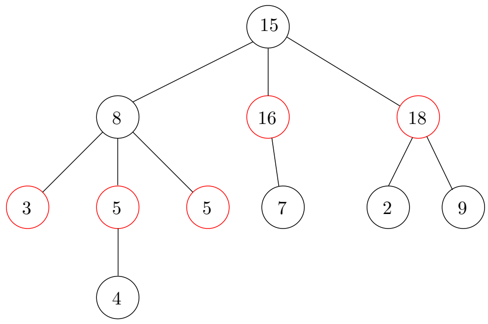

Figure 1: The maximum-weight indpendent set of the tree has weight 47. The red vertices give the independent set.

We assume that the nodes of the tree are $[n] = \{1, 2, 3, \cdots, n\}$. We root the tree at vertex 1, and for each vertex $i \in [2, n]$, the parent of i is a vertex $j < i$.

Input:

- The input is taken from the standard input (console).
- The first line of input contains one integer n, the number of vertices in the tree.
- The next n lines contain two integers each, where the i-th line contains two integers p_i and w_i, where p_i is the parent of i and w_i is the weight of i. We assume $p_1 = 0$, which is useless. For all $i \in [2, n]$, we have $1 \leq p_i < i$.

Output:

- The output is printed to the standard output (console).
- You only need to output one integer, the weight of the maximum-weight independent set.

Example Input:	**Example Input (Continued)**:	**Example Output**:	This is the example from the problem description.
11 0 15 1 8 1 16 1 18 2 3	2 5 2 5 3 7 4 2 4 9 6 4	47	

Constraints:

- $1 \le n \le 10^6$.
- $0 \le w_i \le 10^6$ for every $i \in [n]$.
- It is expected that your program terminates in 10 seconds.

Problem 2. You need to implement the minimum-weight arborescence problem. The input is a directed graph $G = (V, E)$ with edge weights $w \in \mathbb{Z}^E_{\ge 0}$. The vertices of the graph are indexed 1 to n. The root of the arborescence is 1, which does not have incoming edges in G. It is guaranteed that every vertex is reachable from 1 in G, and G does not contain parallel edges.

Input:

- The input is taken from the standard input (console).
- The first line of input contain two integers n and m, indicating the numbers of vertices and edges in G respectively.
- The next m lines give the description of the m edges. Each line contains three integers u, v and w, denoting an edge from u to v of weight w.

Output:

- The output is printed to the standard output (console). It contains a single integer, which is the weight of the minimum-weight arborescence in G rooted at 1.

Example Input:	**Example Input (Continued):**	**Example Output**:
11 23 1 2 7 1 3 10 1 4 5 2 3 6 4 3 8 5 2 3 3 5 8 3 6 2 4 6 5 7 4 3 4 8 7	6 5 6 6 7 10 8 7 6 5 9 5 5 10 10 6 10 7 7 10 8 11 8 9 9 10 8 10 9 10 10 11 4 11 10 6	54

Constraints:

- $1 \le n \le 1000, 1 \le m \le 10000$.
- The weights are integers between 0 and 10^6.
- It is expected that your program terminates in 10 seconds.

Problem 3. You need to implement the algorithm for the project selection problem. You are given a set of n projects, indexed from 1 to n. Each project $i \in [n]$ has a specific weight $w_i \in \mathbb{Z}$, which can be positive or negative. Additionally, there are precedence constraints between the projects: if project i precedes project j, then to select j, you must also select i. The precedence constraints do not induce cycles; that is, if we draw an directed edge (i, j) if i precedes j, then the resulting directed graph does not contain a directed cycle.

The goal of the problem is to select a subset of projects such that the total weight of the selected projects is maximized, while satisfying all the precedence constraints.

Input:

- The input is taken from the standard input (console).
- The first line of the input contains two integers n and m, indicating the the number of projects and the number of precedence constraints respectively.
- The second line contains n integers representing the weights of the projects, with the i-th number denoting the weight of project i.
- The next m lines contains the m precedence constraints. Each line contains two integers $i, j \in [n], i \neq j$, indicating a precedence constraint where project i precedes project j.

Output:

- The output is printed to the standard output (console).
- The output contains a single integer representing the maximum total weight of the selected subset of projects.

Example Input:	**Example Output**:	The optimum solution selects projects $1, 2, 4$ and 6.
6 7 -3 -2 -7 -1 6 10 1 2 1 3 1 4 2 6 3 5 4 5 4 6	4	

Constraints:

- $1 \leq n \leq 1000$.
- $1 \leq m \leq 10000$.
- $-10^6 \leq w_i \leq 10^6$.
- It is expected that your program terminates in 10 seconds.