Task 2. Garden

Sashka is responsible for the maintenance of the flowers in the city garden. The garden consists of N flower beds, numbered with the integers from 1 to N, where some of the flower beds are connected by pipes. The beds and the pipes form a connected, acyclic, undirected graph, in which the flower beds are the nodes and the pipes – the edges. In each bed there is an installed pump which can pump water from the underground. This water irrigates the bed in which the pump is located and sends water through the pipes to some of the beds that have a path of pipes to them. The pumps are numbered from 1 to N, with one pump's number being equal to the number of the bed it is located in. If the pump located in a bed with number x ($1 \leq x \leq N$) works for p minutes, the water will reach all beds that have distance within $p-1$ edges from x in the acyclic graph. The system of pumps is such that one pump runs first, then another one and so on. **Two pumps never work simultaneously**. **A pump can run only once.** A flower bed is considered irrigated if water has reached it, regardless of which pump. The pumps are designed in such a way, that if a pump with number m runs, it should run at most t_m minutes, otherwise it can malfunction. A run of a pump must continue an integer number of minutes. The pumps are identical and consume electricity as follows: 1 minute run costs c_1 euro, 2 minute run costs c_2 euro and so on. If a pump doesn't run, it won't consume electricity. Sashka is given the task of determining the minimum amount of money for electricity, when an appropriate set of pumps runs, so all the flower beds can be irrigated.

Task

Write a program `garden`, which solves the task Sashka is given.

Input

The first line of the standard input contains an integer N, equal to the number of beds (and pumps) in the garden. The second line contains also N non-negative integers $c_1, c_2, \ldots, c_N$, separated by intervals – the money for electricity, which a pump consumes for respectively $1,2, \ldots, N$ minute run. The third line contains N non-negative integers $t_1, t_2, \ldots, t_N$, separated by intervals – the maximum allowed run time for each pump. If some of these integers is equal to 0, this means that the respective pump cannot be used. Each of the last $N-1$ lines of the standard input contains two positive integers u and v, defining a pipe (edge) connecting the beds with numbers u and v. It's guaranteed that the flower beds and the pipes, connecting them, form a connected, acyclic graph.

Output

On the only line of the standard output, the program should output the minimum amount of money for electricity so all the flower beds can be irrigated. If it's impossible to irrigate all the flower beds, your program should output -1.

Constraints

$1 \leq N \leq 2\,000$
$0 \leq c_i \leq 10^6$
$0 \leq t_i \leq N$

Subtasks

№	Additional constraints			Points
	N	Other	Required subtasks	
1	–	The sample test cases	–	0
2	≤ 8	–	1	11
3	≤ 75	The graph is a chain $*$	–	12
4	≤ 500	The graph is a chain $*$	3	11
5	$\leq 2\,000$	The graph is a chain $*$	$3-4$	13
6	≤ 75	–	$1-3$	17
7	≤ 500	–	$1-4; 6$	14
8	$\leq 2\,000$	–	$1-7$	22

Points for a subtask are given only if all the tests for it and the required subtasks are passed.

$*$ A graph is a chain if and only if each node has at most 2 neighbors and there are exactly 2 vertices with 1 neighbor.

Examples

Input	Output
8 1 4 9 16 25 36 49 64 1 5 1 1 0 0 5 0 1 2 2 3 1 4 2 5 2 6 4 7 7 8	8
7 1 4 9 16 25 36 49 0 5 5 0 0 0 0 1 2 2 4 1 3 1 5 3 7 3 6	13

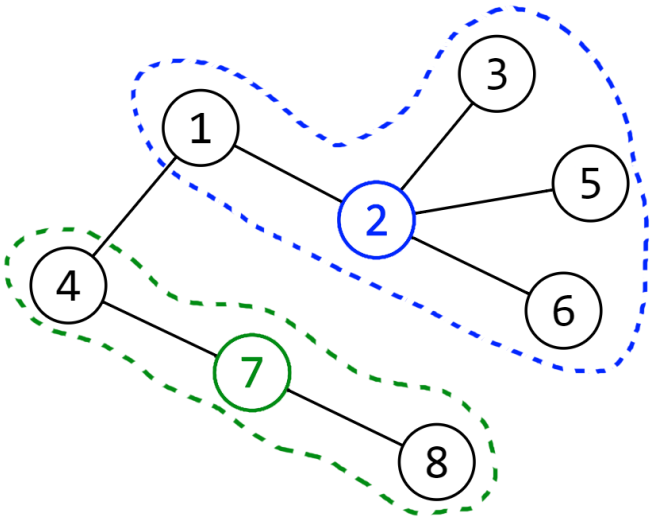
Explanation of the examples

Example №1: An irrigation with a minimum amount of money is achieved when pump №2 runs for 2 minutes and pump №7 runs for 2 minutes. The consumed electricity will cost $c_2 + c_2 = 4 + 4 = 8$ euro.

Example №2: An irrigation with a minimum amount of money is achieved when pump №3 runs for 3 minutes and pump №2 runs for 2 minutes. The consumed electricity will cost $c_2 + c_3 = 4 + 9 = 13$ euro.

Below are diagrams illustrating the graphs from the examples.

Example №1:



Example №2: