Problem J. Gold Transfer

Time limit 4500 ms
Mem limit 262144 kB

You are given a rooted tree. Each vertex contains a_i tons of gold, which costs c_i per one ton. Initially, the tree consists only a root numbered 0 with a_0 tons of gold and price c_0 per ton.

There are q queries. Each query has one of two types:

1. Add vertex i (where i is an index of query) as a son to some vertex p_i; vertex i will have a_i tons of gold with c_i per ton. It's guaranteed that $c_i > c_{p_i}$.
2. For a given vertex v_i consider the simple path from v_i to the root. We need to purchase w_i tons of gold from vertices on this path, spending the minimum amount of money. If there isn't enough gold on the path, **we buy all we can**.

If we buy x tons of gold in some vertex v the remaining amount of gold in it decreases by x (of course, we can't buy more gold that vertex has at the moment). For each query of the second type, calculate the resulting amount of gold we bought and the amount of money we should spend.

Note that you should solve the problem in `online` mode. It means that you can't read the whole input at once. You can read each query only after writing the answer for the last query, so don't forget to flush output after printing answers. You can use functions like `fflush(stdout)` in `C++` and `BufferedWriter.flush` in `Java` or similar after each writing in your program. In standard (if you don't tweak I/O), `endl` flushes `cout` in `C++` and `System.out.println` in `Java` (or `println` in `Kotlin`) makes automatic flush as well.

Input

The first line contains three integers q, a_0 and c_0 ($1 \le q \le 3 \cdot 10^5$; $1 \le a_0, c_0 < 10^6$) — the number of queries, the amount of gold in the root and its price.

Next q lines contain descriptions of queries; The i-th query has one of two types:

- "1 p_i a_i c_i" ($0 \le p_i < i$; $1 \le a_i, c_i < 10^6$): add vertex i as a son to vertex p_i. The vertex i will have a_i tons of gold with price c_i per one ton. It's guaranteed that p_i exists and $c_i > c_{p_i}$.
- "2 v_i w_i" ($0 \le v_i < i$; $1 \le w_i < 10^6$): buy w_i tons of gold from vertices on path from v_i to 0 spending the minimum amount of money. If there isn't enough gold, we buy as much as we can. It's guaranteed that vertex v_i exist.

It's guaranteed that there is at least one query of the second type.

Output

For each query of the second type, print the resulting amount of gold we bought and the minimum amount of money we should spend.

Sample 1

Input	Output
5 5 2 2 0 2 1 0 3 4 2 2 4 1 0 1 3 2 4 2	2 4 4 10 1 3

Note

Explanation of the sample:

At the first query, the tree consist of root, so we purchase 2 tons of gold and pay $2 \cdot 2 = 4$. 3 tons remain in the root.

At the second query, we add vertex 2 as a son of vertex 0. Vertex 2 now has 3 tons of gold with price 4 per one ton.

At the third query, a path from 2 to 0 consists of only vertices 0 and 2 and since $c_0 < c_2$ we buy 3 remaining tons of gold in vertex 0 and 1 ton in vertex 2. So we bought $3 + 1 = 4$ tons and paid $3 \cdot 2 + 1 \cdot 4 = 10$. Now, in vertex 0 no gold left and 2 tons of gold remain in vertex 2.

At the fourth query, we add vertex 4 as a son of vertex 0. Vertex 4 now has 1 ton of gold with price 3.

At the fifth query, a path from 4 to 0 consists of only vertices 0 and 4. But since no gold left in vertex 0 and only 1 ton is in vertex 4, we buy 1 ton of gold in vertex 4 and spend $1 \cdot 3 = 3$. Now, in vertex 4 no gold left.