Monkey Business

Objective

Give practice with Trees (or tree-like input) in C.
Give practice with Stacks in C.

Story

Your park has a new layout. The park is modeled as a collection of exhibits (including the entrance of the park) each with their own unique ID. There are also paths in your park. Each path connects a pair of exhibits. Additionally, between each pair of exhibits there is exactly 1 sequence of unique paths that connects. One aspect of the park that draws in a lot of guests are the monkeys that hang around each exhibit (including the entrance of the park).

Guests that visit your park love to see a lot of monkeys. However, the guests are very busy, so they typically have time to walk from the entrance to one particular exhibit, then walk back to the entrance to leave. They don't visit any more exhibits than necessary. You were going to give the visitors a map so that they could plan out the visits that would enable them to visit the most monkeys, but someone swiped your reference map. Now you want to settle for telling the guests how many distinct monkeys they will see in their visit to your park.

Luckily, you have a dedicated employee that walked the entire park to give you a report of the existing pathways and the monkeys at each location. The employee started at the entrance and crossed all pathways exactly twice. The first time the employee reached a location they wrote down a unique integer ID for the location and the number of monkeys they saw. Everytime they used a pathway a second time they reached an exhibit they had already seen, so they noted that the exhibit had already been recorded in their notes. The employee may have visited an exhibit multiple times.

Problem

Given the employee's traversal of the park, and the exhibits that the visitors want to visit determine how many unique monkeys the visitor will see on their way from the entrance to the exhibit and then back.

Input

Input will begin with a line containing 1 integer, $\boldsymbol{n}$ ($1 \leq \boldsymbol{n} \leq 200{,}000$), representing the number of exhibits in the park (including the entrance). The following line contains 2 integers, $\boldsymbol{i}$ and $\boldsymbol{m}$ ($\boldsymbol{i} = 1$; $1 \leq \boldsymbol{m} \leq 5{,}000$), representing the ID of the entrance and the number of monkeys at the entrance. Each of the following $2\boldsymbol{n} - 2$ lines will contain a path description recorded by our employee. If this is the first time the employee used the path, then the line will contain two integers, $\boldsymbol{i}$ and $\boldsymbol{m}$ ($2 \leq \boldsymbol{i} \leq \boldsymbol{n}$; $1 \leq \boldsymbol{m} \leq 5{,}000$), representing the ID of the destination exhibit and the monkeys at the destination exhibit respectively. If this is the second time the path is used, then the line will contain the integer -1 instead, since the ID and number of monkeys were already recorded. You can assume that the monkeys don't move between exhibits.

The following line contains a single integer, $\boldsymbol{v}$ ($1 \le \boldsymbol{v} \le 500{,}000$), representing the number of visitors to the park. The following $\boldsymbol{v}$ lines will each contain a single integer, $\boldsymbol{e}$ ($1 \le \boldsymbol{e} \le \boldsymbol{n}$), representing the ID of the exhibit that the guest is going to visit.

Output

Output should contain $\boldsymbol{v}$ lines. Each line will contain a corresponding integer representing the number of unique monkeys seen in the corresponding visitor's park visit. The order of the output should correspond to the order of the input.

Sample Input	**Sample Output**

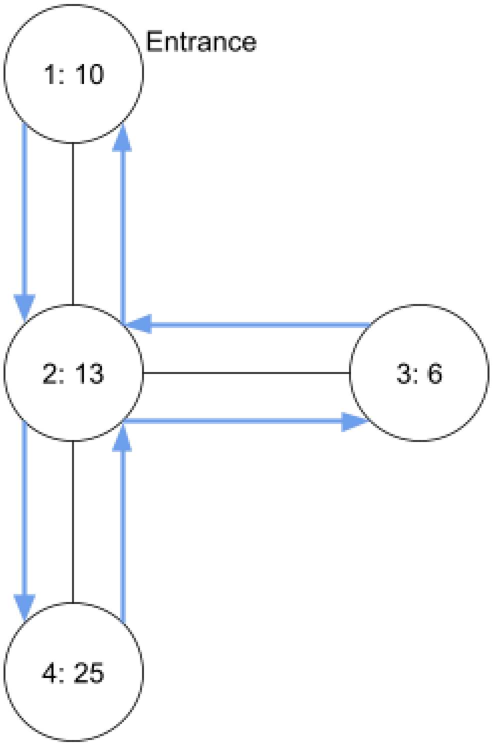
`4`
`1 10`
`2 13`
`4 25`
`-1`
`3 6`
`-1`
`-1`
`3`
`4`
`3`
`2`
	`48`
	`29`
	`23`

`5`
`1 1`
`2 2`
`3 3`
`-1`
`-1`
`4 4`
`5 5`
`-1`
`-1`
`2`
`3`
`5`
	`6`
	`10`

Explanation

Case 1

Below is the diagram of the park with a blue line demonstrating the path that our dedicated employee followed to map out the park.



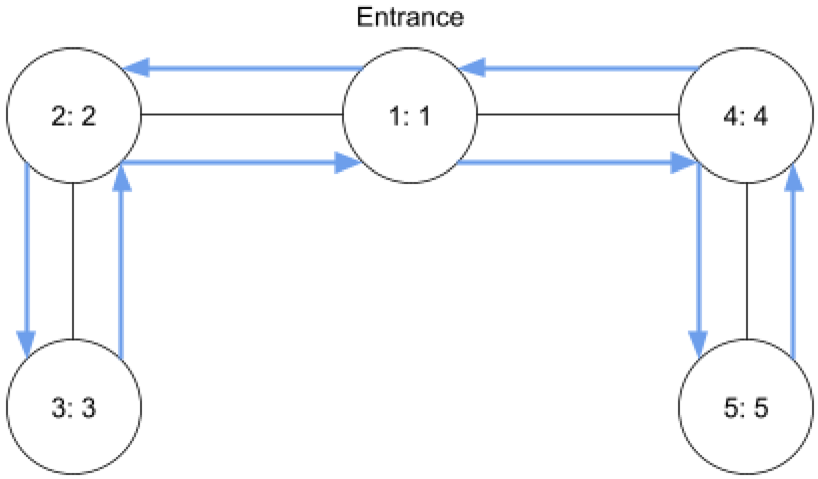
The values in the circles represent the exhibit ID and the monkeys in that exhibit.

There are 3 visitors in the first case.

- The first visitor wants to go to exhibit 4. To get to and leave from exhibit 4 the visitor will visit exhibits 1, 2, and 4. The total monkeys seen on that path will be 10 + 13 + 25 = 48.
- The second visitor wants to go to exhibit 3. To get to and leave from exhibit 3 the visitor will visit exhibits 1, 2, and 3. The total monkeys seen on that path will be 10 + 13 + 6 = 29.
- The third visitor wants to go to exhibit 2. To get to and leave from exhibit 2 the visitor will visit exhibits 1 and 2. The total monkeys seen on that path will be 10 + 13 = 23.

Case 2

Below is the diagram of the park with a blue line demonstrating the path that our dedicated employee followed to map out the park.



The values in the circles represent the exhibit ID and the monkeys in that exhibit.

There are 2 visitors in the second case.

- The first visitor wants to go to exhibit 3. To get to and leave from exhibit 3 the visitor will visit exhibits 1, 2, and 3. The total monkeys seen on that path will be 1 + 2 + 3 = 6.
- The second visitor wants to go to exhibit 5. To get to and leave from exhibit 5 the visitor will visit exhibits 1, 4, and 5. The total monkeys seen on that path will be 1 + 4 + 5 = 10.

Hints

Graph: It’s a graph. Treat the exhibits as nodes and the pathways as edges.

Tree: It’s really a tree since there is a unique path from each node to each other node. If there is a cycle in the graph then some pair of nodes would have more than 1 path between them.

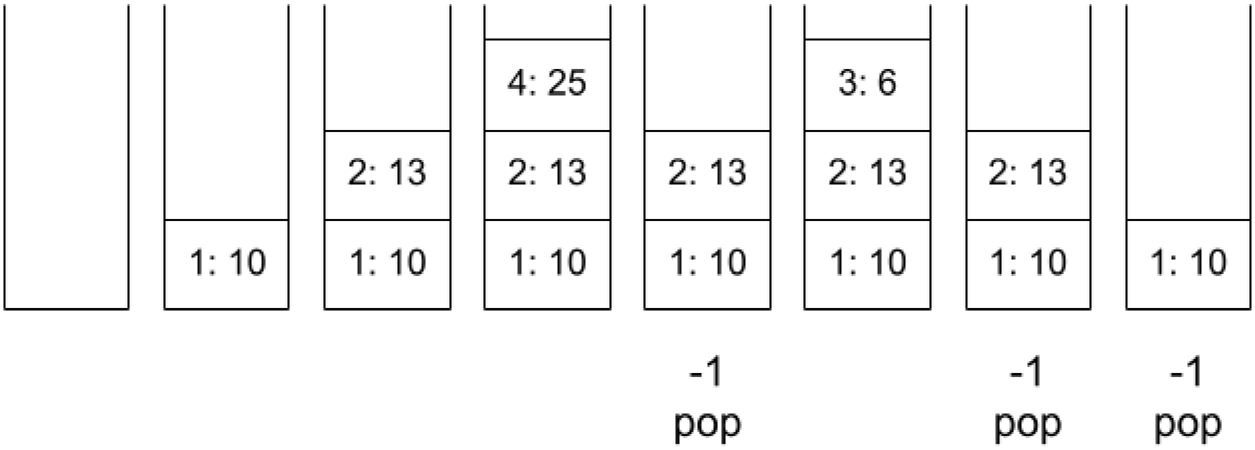
Rooted Tree: You can think of the tree as being rooted at the entrance.

Monkeys on a Path: The monkeys seen from the entrance to an exhibit are the values on the path from the root to a given node.

Store Parents: Knowing the parent of a node can help inform you where the employee ends up when they record a -1 as their path.

[Alternatively] Stack Usage: A stack could also be used to determine where and how many monkeys are at your location. When a location is left, you should pop the location off the stack.

Below is a demonstration of what the stack would look like after each path crossing in case 1.



Store Current Node: When the employee walks to some new nodes the parent should have been the previous node they were at.

Store the Answers: Keep an array of all the answers so that when a visitor asks for the number of monkeys on a path you can answer the query in O(1).

Grading Criteria

- Good comments, whitespace, and variable names
    - 15 points
- No extra input output (e.g. input prompts, “Please enter the number of friends:”)
    - 10 points
- “Traverse” the graph using the given order of nodes
    - 5 points
- Use a stack (or recursion) to keep track of which exhibits need to be visited to reach a particular exhibit
    - 10 points
- Store the number of monkeys for each exhibit
    - 5 points
- Use the stored values when answering queries for the visitors.
    - 5 points
- Programs will be tested on 10 cases
    - 5 points each

No points will be awarded to programs that do not compile using “gcc -std=gnu11 -lm”.

Sometimes a requested technique will be given, and solutions without the requested technique will have their maximum points total reduced. For this problem use a stack, recursion, or parent information. ***Without this programs will earn at most 50 points!***

Any case that causes a program to return a non-zero return code will be treated as wrong. Additionally, any case that takes longer than the maximum allowed time (the max of {5 times my solutions time, 10 seconds}) will also be treated as wrong.

No partial credit will be awarded for an incorrect case.