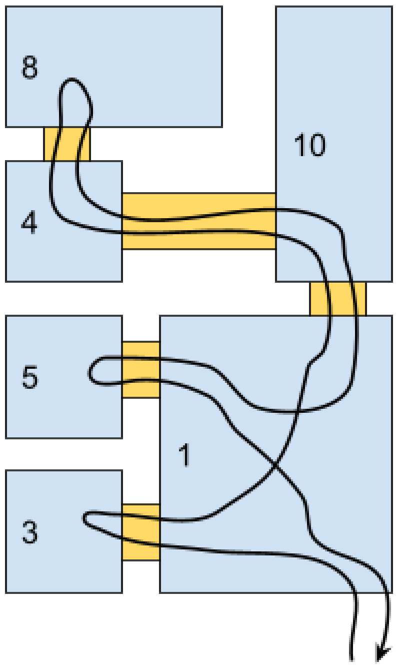
New Layout

Objective
Give practice with stacks input in C.
Give practice with trees in C.

Story
Well the ship is fixed (sort of). The repair crew just left, but they did something awful. The ship has a completely different interior. Hallways have become blocked off and some walls were removed. You have been in communication with the repair crew. They apparently had to make the ship up to code or something. You are exhausted and want this ship to set sail as soon as possible. Now you are left with the daunting task of determining how many rooms your new ship's layout can support.



The new layout can be thought of as a series of rooms connected by corridors. Each corridor connects 2 rooms; no room is connected to itself via a corridor. There is exactly 1 series of distinct corridors that connects any pair of rooms.

You are going to assume that no one should walk through someone's quarters to reach another room. Because of this you will assume that (with the exception of the ship's entrance) each possible room will be connected to exactly 1 corridor.

You don't have time to walk through the ship yourself, so you hired an employee to walk through each of the rooms. He came up with a unique room ID for each room they visited. Rather than telling you all the corridors that connect the rooms, your employee walked through all the corridors (each one exactly twice), and they kept a list of the rooms they visited by the IDs in the order in which they visited the room. Additionally each room ID was noted each time they entered the room. This means a room ID can appear multiple times.

You are also worried that the rooms might be very far from the main entrance, so you will also track total distance (in terms of number of corridors) from all possible quarters to the main entrance.

Problem
Given a list of room IDs representing the order in which the employee visited the rooms, determine how many possible sleeping quarters the ship can have an the sum of the distances to each possible quarter.

Input

Input will consist of a series of integers each one on their own line. The series ends with the integer -1. No earlier integer in the series will be -1. The integer series represents the room IDs visited in the traversal that the employee executed. The employee started in the main entrance. It is guaranteed that the main entrance will be the last room ID before the -1. There will always be at least one ID that is not -1 in the input.

Output

Output should contain a single line containing 2 integers. The first integer represents the number of possible quarters the second integer represents the total distance (in terms of corridors) from each room to the main entrance.

Sample Input	**Sample Output**

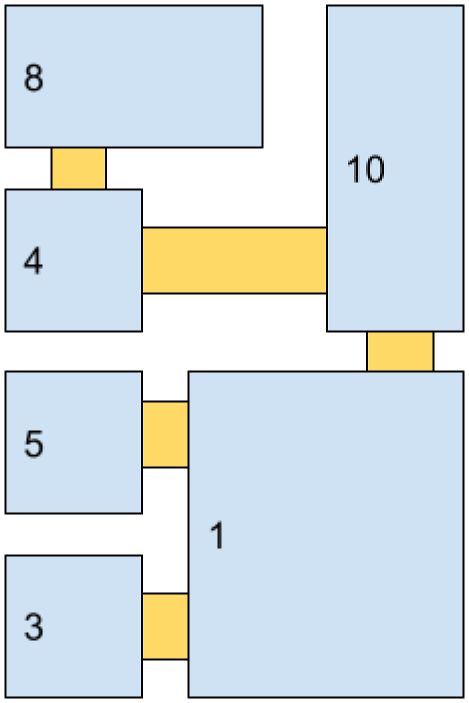
`1`
`3`
`1`
`10`
`4`
`8`
`4`
`10`
`1`
`5`
`1`
`-1`
	`3 5`

`8`
`2`
`4`
`5`
`4`
`6`
`4`
`2`
`8`
`-1`
	`2 6`

Explanation

Case 1

There are 6 rooms in the first case. The rooms have IDs, 1, 3, 4, 5, 8, and 10. The main entrance has an ID of 1. The rooms have a layout equivalent to the picture on the following page.



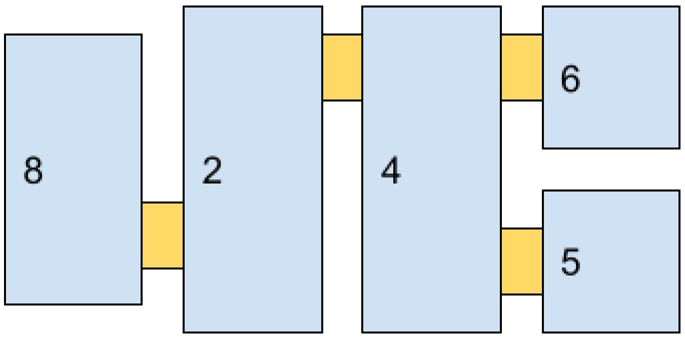
The possible quarters are ID'd 3, 5, and 8, since none of them have a room that has to pass through them to reach the main entrance (room ID'd 1).

The number of corridors to reach room 5 from the main entrance is 1.
The number of corridors to reach room 3 from the main entrance is also 1.
The number of corridors to reach room 8 from the main entrance is 3 (1 to 10, 10 to 4 and 4 to 8).

The total number of corridors for all possible quarters is 1+1+3 = 5.

Thus the output is 3 (number of possible quarters) and 5 (sum of corridors for the possible quarters).

Case 2
The layout looks like the following



With the main entrance being room 8. The number of possible quarters is 2 (room 6 and 5).

The number of corridors to get to said quarters is 3 (for room 6) + 3 (for room 5). The total is 6.

The output for the case is 2 (number of quarters) and 6 (corridor sum).

Hints

Reading Input: Use a while loop to process all the rooms.

Graph: The input is a graph, where the rooms are the nodes and the corridors are the edges.

Tree: The room graph forms a tree, because the number of paths between pairs of distinct nodes is exactly 1.

Parents: Track the room that was visited prior (on the path back to the main entrance) to each room. This prior room is the parent node in terms of a rooted tree. When a parent is revisited via the inputted room sequence that means that the employee was heading back to the main entrance and left the current room.

Once a room is left (back towards the main entrance) it will never be visited again in the employee's walk.

Possible Quarters: A room can only be a quarter, if the room has no children rooms.

Room Distance: *DO NOT* walk back up the tree to find the distance for a possible quarter to the main entrance.

Stack/Depth: Consider using a stack of nodes representing the path from the main entrance to the current room or storing the depth of a node to determine the distance from the main entrance to a possible quarter. The distance would be the size of the stack. The stack would also allow easy access to parent rooms.

Grading Criteria

- Read/Write from/to **standard** input/output (e.g. scanf/printf and no FILE *)
  - 5 points
- Good comments, whitespace, and variable names
  - 15 points
- No extra input output (e.g. input prompts, "Please enter the number of friends:")
  - 5 points
- Loop until a -1 is read in
  - 5 points
- "Store the depth" for a room
  - 10 points
- Track how many children a room has
  - 5 points
- Track the parent of each room
  - 5 points
- Programs will be tested on <u>10 cases</u>
  - 5 points <u>each</u>

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 or a tree. ***<u>Without this programs will earn at most 50 points!</u>***

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.

<u>No partial credit will be awarded for an incorrect case.</u>