Exciting Tournament

Objective
Give practice with BST insertion.
Give practice with tree traversal.

Story
You are setting up a tournament to promote your newly opened arcade. There has not been a tournament in Gameland for quite some time. Tournaments in Gameland are a big spectacle. The tournament is a (sometimes unbalanced) single elimination bracket tournament. However, each round only consists of one game. The single game per round format allows the entire audience to focus their attention on the match.

In our tournament we will have N competitors. There will always be N - 1 games because of the single elimination factor. At the beginning of the tournament there will be N - 1 virtual tables lined up in a row. There is exactly 1 player spot between each adjacent pairs of virtual tables. There are two additional player spots at the beginning/end of the row of virtual tables. Each player will take one of these spots; no spot will be occupied by two players. In each round a single table will turn on and the player closest on the left and the player closest on the right will play a game to determine a victor (there are never draws). After which the losing player will leave the stage and victor will remain for any remaining games. Given any execution and win/lose order there will always be two players that can play each game. We have also ensured that no player will play more than 1000 games.

The games themselves are not very interesting to you. You know that each player has some unique, measurable skill level, and the player with the highest skill always wins their match. You could probably make a fair bit of money by betting on the matches, but you are more focussed on promoting your arcade. You know that the excitement of a match is computed using the absolute value of the difference of the skills of the players. You want to write a program that can determine the sum of the excitement of a given tournament (order of players and order of game tables).

Problem
Given the table activation order and the skill order of the players determine the sum of the excitement across all games in the tournament.

Input
Input will begin with a line containing a single integer, n ($1 \leq n \leq 500{,}000$), representing the number of players in the tournament. The following line contains n - 1 integers, $a_1, a_2, \ldots, a_{n-1}$ ($1 \leq a_i \leq n$ - 1; each of which will be unique), representing the table activation order. Following this will be a second line containing n integers, $b_1, b_2, \ldots, b_n$ ($0 \leq b_i \leq 1{,}000{,}000{,}000$; each of which will be unique) representing the skill of the players in the order at the start of the tournament.

Output

Output will contain 1 integer representing the sum of the excitement of all the games in the tournament

Sample Input	Sample Output
5 1 3 4 2 5 1 4 2 3	8
6 4 3 5 2 1 3 19 10 20 13 6	49

Explanation

Case 1

We have the following arrangement at the start

Skill 5	Table 1	Skill 1	Table 2	Skill 4	Table 3	Skill 2	Table 4	Skill 3

Table 1 is the first to activate according to the second line of input ("**1** 3 4 2").

Skill 5	Table 1	Skill 1	Table 2	Skill 4	Table 3	Skill 2	Table 4	Skill 3

The player with skill of 1 is eliminated and the player with skill of 5 stays. We generate 5-1 = 4 excitement.

Skill 5		Skill 1	Table 2	Skill 4	Table 3	Skill 2	Table 4	Skill 3

The next table to activate is 3 again according to the second line of input ("1 **3** 4 2")

Skill 5		Skill 1	Table 2	Skill 4	Table 3	Skill 2	Table 4	Skill 3

The player with skill of 2 is eliminated and the player with skill of 4 stays. We generate 4-2 = 2 excitement

Skill 5		Skill 1	Table 2	Skill 4		Skill 2	Table 4	Skill 3

The next table to activate is 4 again according to the second line of input ("1 3 **4** 2")

Skill 5		Skill 1	Table 2	Skill 4		Skill 2	Table 4	Skill 3

The player with skill of 3 is eliminated and the player with skill of 4 stays. We generate 4-3 = 1 excitement

Skill 5		Skill 1	Table 2	Skill 4		Skill 2		Skill 3

The last game is played at table 2 between the player with skill 5 and the player with skill 4. The excitement of the game is 1, and the player with skill 4 is eliminated.

Skill 5		Skill 1	Table 2	Skill 4		Skill 2		Skill 3

Skill 5		Skill 1		Skill 4		Skill 2		Skill 3

The total excitement of the tournament is 4 + 2 + 1 + 1 = **8**.

Case 2

We have the following arrangement at the start

S3	T1	S10	T2	S19	T3	S20	T4	S13	T5	S6

To put the process into a different perspective let's imagine removing the tables and players as the rounds proceed. The first table to activate is 4 (because of the second line "**4** 3 5 2 1").

S3	T1	S10	T2	S19	T3	S20	T4	S13	T5	S6

Player with skill 20 beats the player with skill 13. We gain 20-13 = 7 excitement and produce the following result.

S3	T1	S10	T2	S19	T3	S20	T5	S6

The second table to activate is 3 (because of "4 **3** 5 2 1").

S3	T1	S10	T2	S19	T3	S20	T5	S6

Player with skill 20 beats the player with skill 19. We gain 20-19 = 1 excitement and produce the following result.

S3	T1	S10	T2	S20	T5	S6

The third table to activate is 5.

S3	T1	S10	T2	S20	T5	S6

Player with skill 20 beats the player with skill 6. We gain 20-6 = 14 excitement and produce the following result.

S3	T1	S10	T2	S20

The next table to activate is 2.

S3	T1	S10	T2	S20

Player with skill 20 beats the player with skill 10. We gain 20-10 = 10 excitement and produce the following arrangement.

S3	T1	S20

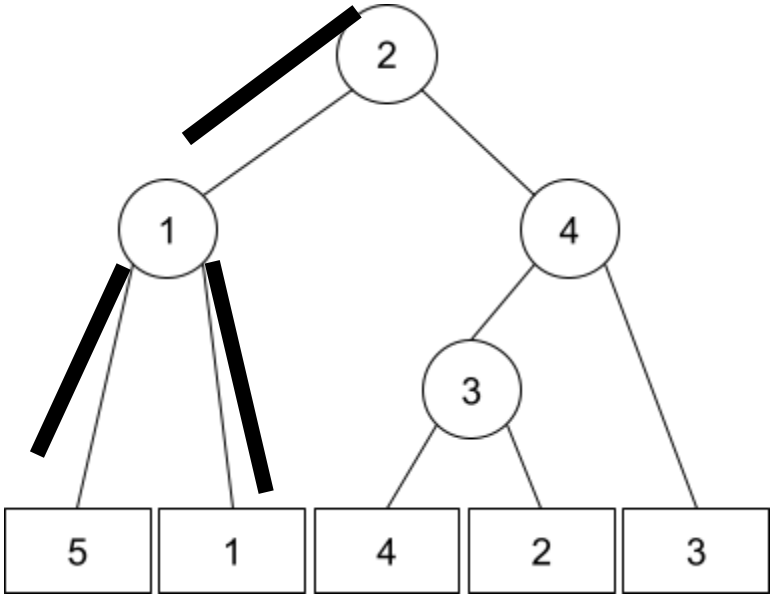
The last table to activate is table 1. We gain 20-3 = 17 excitement and the player with skill 20 is the champion.

The total excitement is 17 + 10 + 14 + 1 + 7 = 49.

Hints

The Tree: The way in which the games play out forms a binary search tree. Each node in the tree is one of the tables. The top (root) of the tree is the last table at which a game is played.

Here is a picture of the tree in the first sample



We need to build the trees, but the last node listed is the last table played at. This is similar to a post order traversal. It's not exactly a post-order traversal, but the tree can be constructed using the reverse order of the tables listed.

The Node: Each table (node) will have a victor in addition to an identifier. The two people that play in a match are the victors from the children's tables (or a person if they have not played a game yet. I recommend storing this information in the node struct in the C code.

Finding victors: We can traverse the tree and compute the victors for each node by using a post order method (compute the victor of a node after computing the victors of the sub-trees). We can use this same recursive method to return the resulting excitement for the sub-tree

NULL Nodes: When a null node is reached, we must grab one of the players from our list. As it so happens the first NULL node reached (in post-order traversal) is the first player in our line up. The second NULL node reached is the second player, and so on. If we extrapolate this patter we can keep track of how many NULL nodes seen (as a global variable) and use it to find which player in the list is to be used.

Grading Criteria

- Read/Write from/to **standard** input/output (e.g. scanf/printf and no FILE *)
    - 10 points
- Good comments, whitespace, and variable names
    - 15 points
- Construct a Tree
    - 10 points
- Use a post order to find all the
    - 10 points
- Store the number of NULL nodes and the skills of the players either as a global variable or as a reference
    - 5 points
- Programs will be tested on 10 cases
    - 5 points each

There will be a 10 point penalty for having extra output (e.g. input prompts).

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

Sometime a requested technique will be given, and solutions without the requested technique will have their maximum points total reduced. For this problem use a Binary Search Tree and a Tree traversal. ***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.