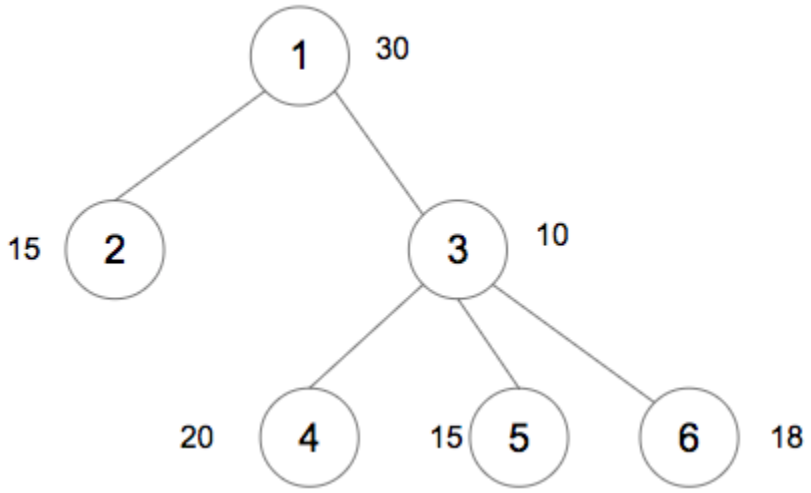
Good Node-subsets

Time Limit	Memory Limit
1 second (See Below)	512 MB

Description

Alice and Bob like to play a game using trees. Let there be a tree with n nodes that are labeled from 1 to n. Let p[i] be the parent of node i, and let p[i] = 0 if node i is the root. Lastly, let v[i] be the integer value of node i assigned to it.



For instance, the tree above describes the case where n = 6, v = [30, 15, 10, 20, 15, 18] and p = [0, 1, 1, 3, 3, 3]. The number next to each node is the integer value assigned to it. In this tree, p[1] = 0 indicates that node 1 is the root node. Notice that every node except for the root node has a parent node.

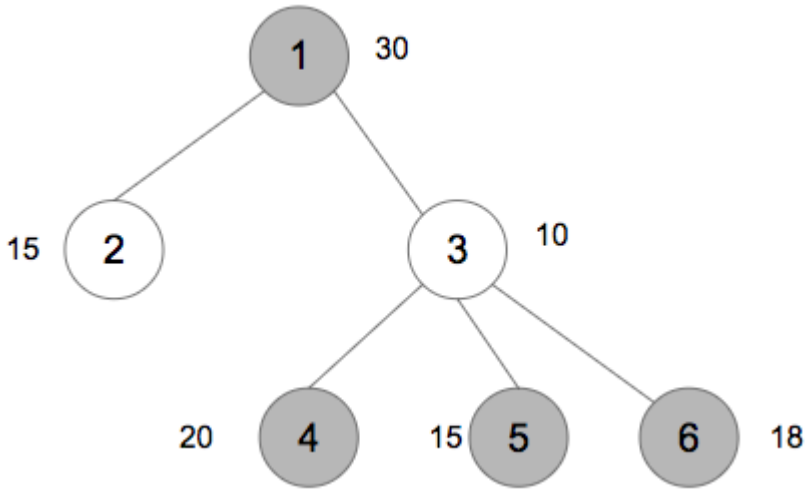
If a node-subset S of the tree satisfies the following conditions, then we call it a "good node-subset":

- Condition 1: If x is in S, then child/parent node(s) of x (that are connected to x) are NOT in S.
- Condition 2: If x with 1 or more child node(s) is NOT in S, then at least one child node of x is in S.

In the tree above, consider the following node-subsets:

- S = {}: Condition 2 is not satisfied -- Node 3 is not in S, and thus at least one of {4, 5, 6} must be in S. Hence it is not a good node-subset.
- S = {3}: This is a good node-subset as it satisfies both conditions.
- S = {1, 2, 3}: Condition 2 is satisfied, but condition 1 is not.
- S = {1, 4, 6}: This is a good node-subset as it satisfies both conditions.
- S = {2, 3}: This is a good node-subset as it satisfies both conditions.

Let Score(S) be the score of a good node-subset S where it is the sum of the values assigned to the nodes in S. In the above example, if S = {1, 4, 5, 6}, then Score(S) = 30 + 20 + 15 + 18 = 83, which is the highest score achievable.



Given a tree, compute the largest score of a good node-subset that Alice and Bob can achieve.

Input

The first line of the input will contain T, the number of test cases.

The first line of each test case will contain n, the number of nodes. The second line will contain n integers separated by whitespace, describing the values assigned to nodes (v[1], ..., v[n]). The third line will contain n integers separated by whitespace, describing the parent of each node (p[1], ..., p[n]).

Output

Output the answer for each test case in each line.

Limit

- $1 \le T \le 10$
- $2 \le n \le 100{,}000$
- $-10^9 \le v[i] \le 10^9$ (i = 1, 2, ..., n)
- $0 \le p[i] \le n$ (i = 1, 2, ..., n)
- For each test case:
    - Among i = 1, 2, ..., n, p[i] = 0 holds for exactly one i.
    - For all i, $p[i] \neq i$.

Sample Input 1 Copy

```
5
6
30 15 10 20 15 18
```

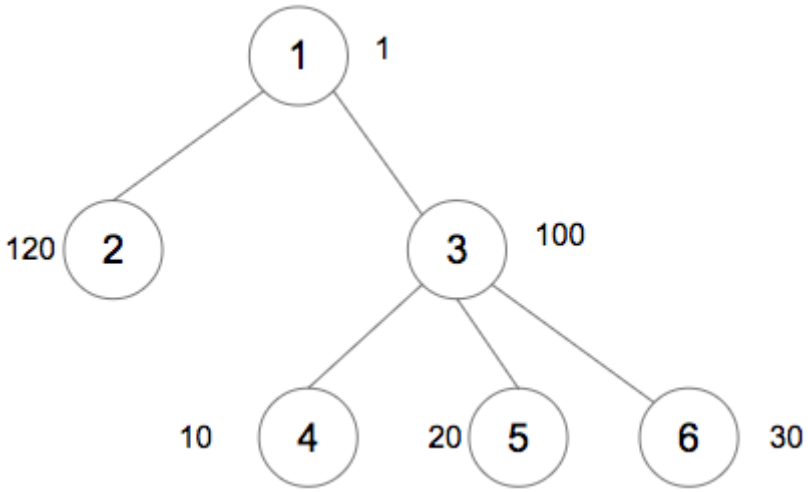
```
0 1 1 3 3 3
6
1 120 100 10 20 30
0 1 1 3 3 3
6
100 8 5 -20 -30 15
0 1 1 3 3 3
5
-1 -2 -3 -4 -5
2 3 4 5 0
2
-2022 2022
0 1
```

Sample Output 1 Copy

```
83
220
115
-6
2022
```

Case 1: Discussed in the problem statement.

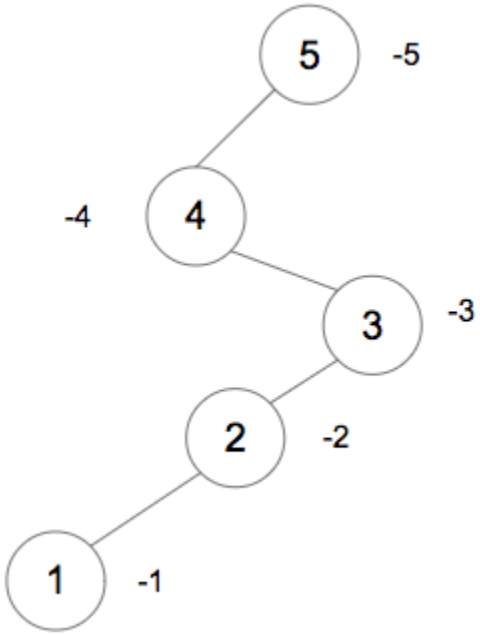
Case 2: It's the same tree as in Case 1, but the values assigned to nodes are different.



In this case, S = {2, 3} maximizes the score.

Case 3: No explanation.

Case 4: The tree is shown below. S = {4, 2} maximizes the score.



Case 5: No explanation.

Time Limit

- Java 8: 3 seconds
- Python 3: 6 seconds

- PyPy3: 6 seconds
- Java 8 (OpenJDK): 3 seconds
- Java 11: 3 seconds
- Kotlin (JVM): 3 seconds
- Java 15: 3 seconds