339. Nested List Weight Sum

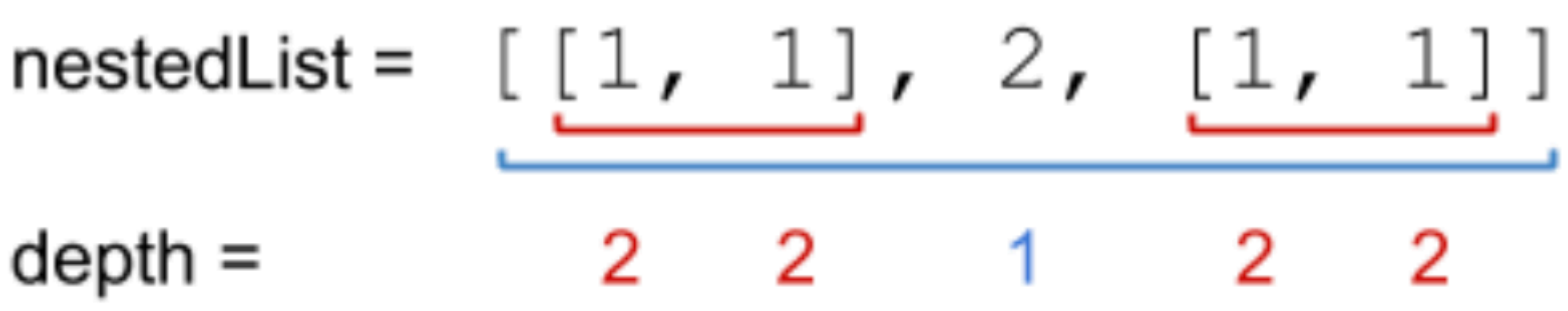
Medium 1383 315 Add to List Share

You are given a nested list of integers `nestedList`. Each element is either an integer or a list whose elements may also be integers or other lists.

The **depth** of an integer is the number of lists that it is inside of. For example, the nested list `[1,[2,2],[[3],2],1]` has each integer's value set to its **depth**.

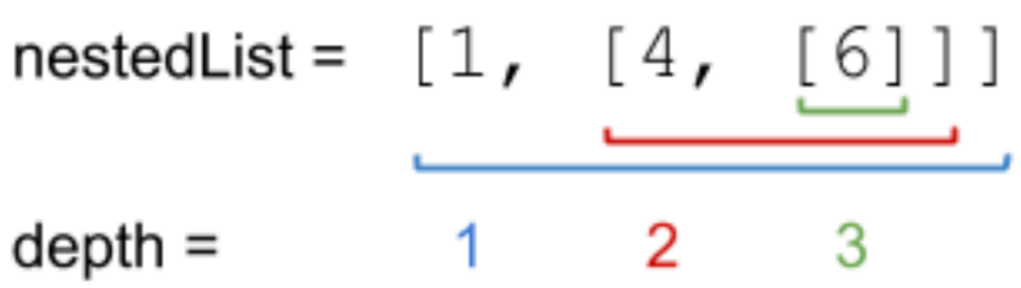
Return *the sum of each integer in* `nestedList` *multiplied by its* ***depth***.

Example 1:



```
Input: nestedList = [[1,1],2,[1,1]]
Output: 10
Explanation: Four 1's at depth 2, one 2 at depth 1. 1*2 + 1*2
+ 2*1 + 1*2 + 1*2 = 10.
```

Example 2:



```
Input: nestedList = [1,[4,[6]]]
Output: 27
Explanation: One 1 at depth 1, one 4 at depth 2, and one 6 at
depth 3. 1*1 + 4*2 + 6*3 = 27.
```

Example 3:

```
Input: nestedList = [0]
Output: 0
```

Constraints:

- `1 <= nestedList.length <= 50`
- The values of the integers in the nested list is in the range `[-100, 100]`.
- The maximum **depth** of any integer is less than or equal to `50`.

[1, [4, [6]]]

Level(1) —-> 1

Level(2) —-> 4

Level(3) —-> 6

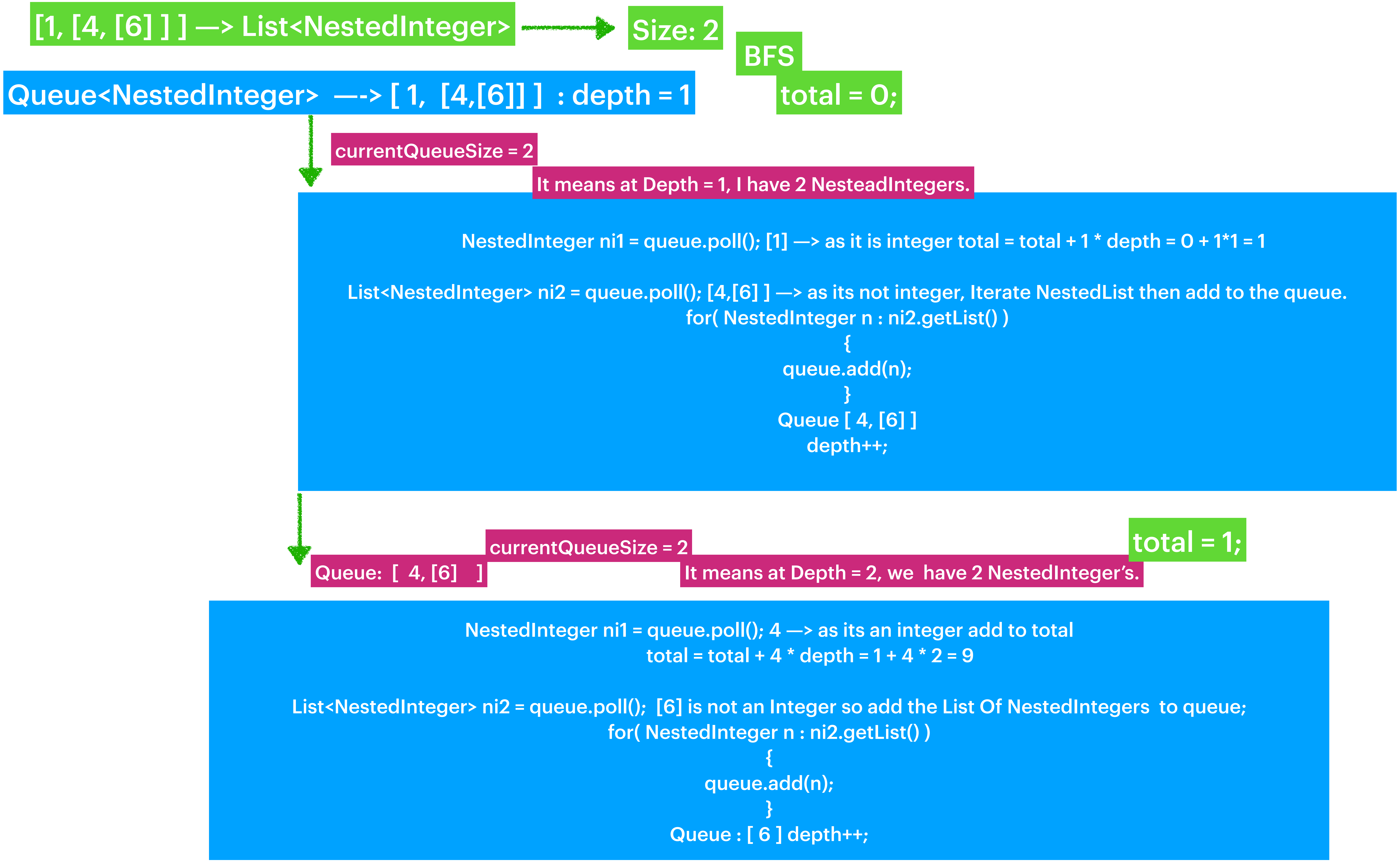
Summation by multiple of depth -> 1*1 + 2 * 4 + 3 * 6 = 27

[[1,1], 2, [1,1]]

Level(1) —-> 2

Level(2) —-> {1,1,1,1}

Summation by multiple of depth ->
1 * 2 + 2 * 1 + 2 * 1 + 2*1 + 2*1 = 10



[1, [4, [6]]] —> List<NestedInteger>

Size: 2

BFS

total = 0;

Queue<NestedInteger> —-> [1, [4,[6]]] : depth = 1

currentQueueSize = 2

It means at Depth = 1, I have 2 NesteadIntegers.

NestedInteger ni1 = queue.poll(); [1] —> as it is integer total = total + 1 * depth = 0 + 1*1 = 1

List<NestedInteger> ni2 = queue.poll(); [4,[6]] —> as its not integer, Iterate NestedList then add to the queue.

```
for( NestedInteger n : ni2.getList() )
{
queue.add(n);
}
Queue [ 4, [6] ]
depth++;
```

currentQueueSize = 2

total = 1;

Queue: [4, [6]]

It means at Depth = 2, we have 2 NestedInteger's.

NestedInteger ni1 = queue.poll(); 4 —> as its an integer add to total
total = total + 4 * depth = 1 + 4 * 2 = 9

List<NestedInteger> ni2 = queue.poll(); [6] is not an Integer so add the List Of NestedIntegers to queue;

```
for( NestedInteger n : ni2.getList() )
{
queue.add(n);
}
Queue : [ 6 ] depth++;
```

total = 9;

currentQueueSize = 1

Queue: [6]

It means at Depth = 3, we have 1 NestedInteger.

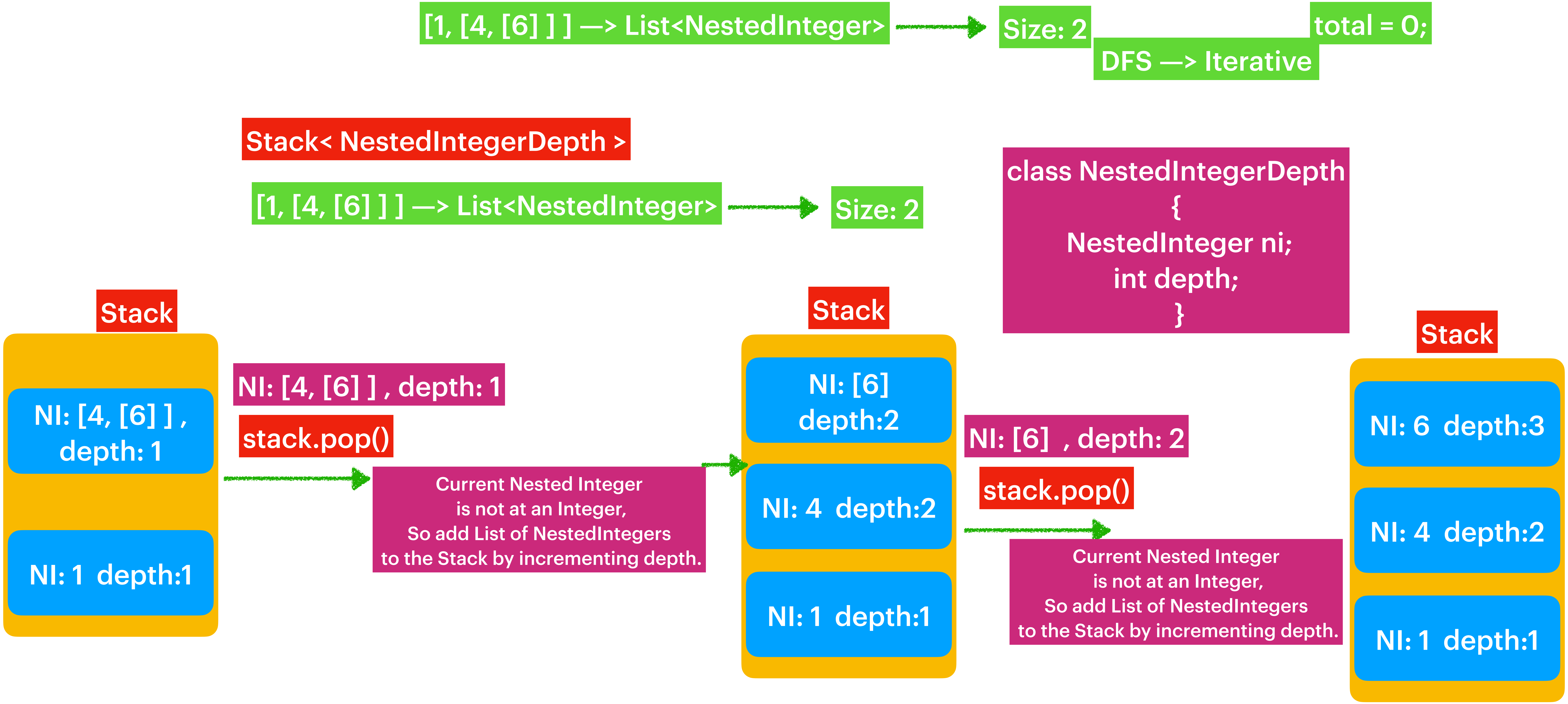
NestedInteger ni1 = queue.poll(); 6 —> as its an integer add to total
total = total + 6 * depth = 9 + 6*3 = 27

queue[]

As Queue is Empty Return the Total = 27

Time Complexity : O(n)

Space Complexity : O(n)



[1, [4, [6]]] —> List<NestedInteger>

Size: 2

DFS —> Iterative

total = 0;

Stack< NestedIntegerDepth >

[1, [4, [6]]] —> List<NestedInteger>

Size: 2

```
class NestedIntegerDepth
{
NestedInteger ni;
int depth;
}
```

Stack

NI: [4, [6]] , depth: 1

NI: 1 depth:1

NI: [4, [6]] , depth: 1

stack.pop()

Current Nested Integer is not at an Integer, So add List of NestedIntegers to the Stack by incrementing depth.

Stack

NI: [6] depth:2

NI: 4 depth:2

NI: 1 depth:1

NI: [6] , depth: 2

stack.pop()

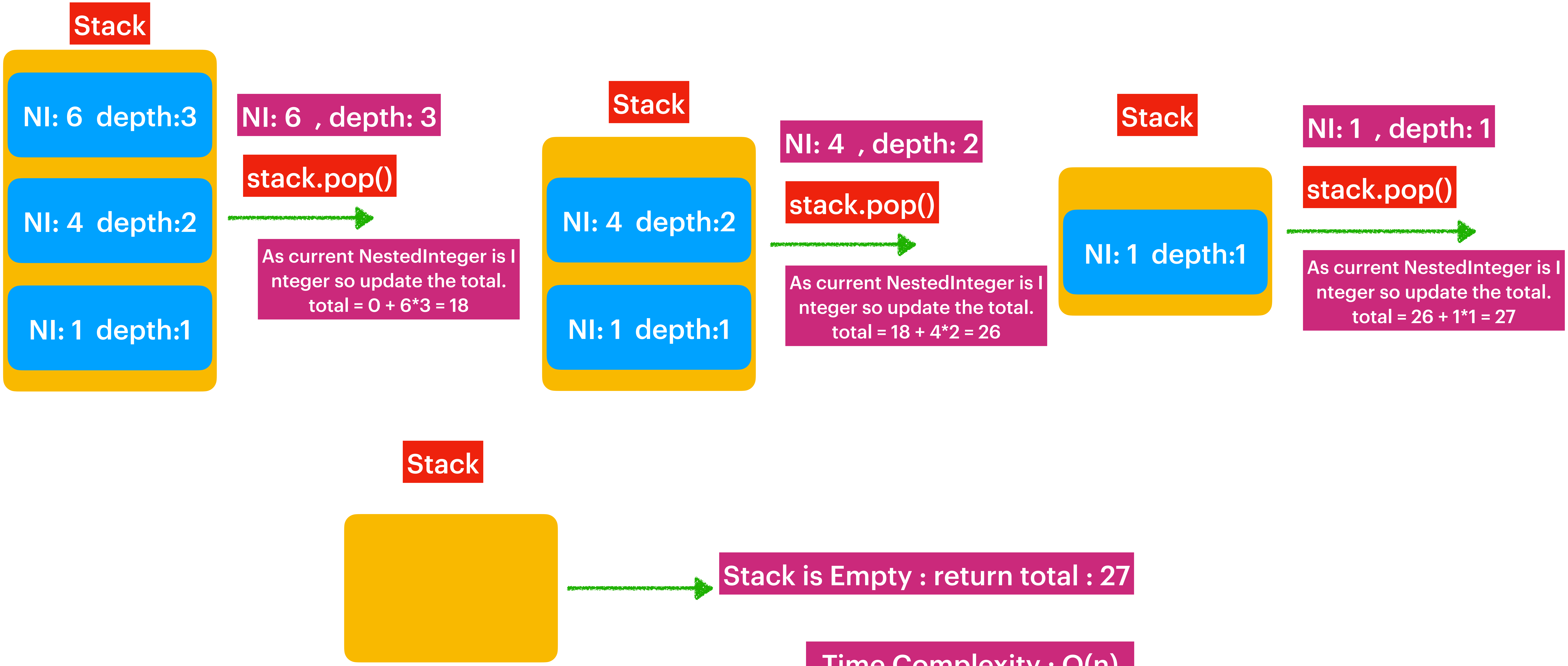
Current Nested Integer is not at an Integer, So add List of NestedIntegers to the Stack by incrementing depth.

Stack

NI: 6 depth:3

NI: 4 depth:2

NI: 1 depth:1



Stack

NI: 6 depth:3

NI: 4 depth:2

NI: 1 depth:1

NI: 6 , depth: 3

stack.pop()

As current NestedInteger is Integer so update the total.
total = 0 + 6*3 = 18

Stack

NI: 4 depth:2

NI: 1 depth:1

NI: 4 , depth: 2

stack.pop()

As current NestedInteger is Integer so update the total.
total = 18 + 4*2 = 26

Stack

NI: 1 depth:1

NI: 1 , depth: 1

stack.pop()

As current NestedInteger is Integer so update the total.
total = 26 + 1*1 = 27

Stack

Stack is Empty : return total : 27

Time Complexity : O(n)
Space Complexity : O(n)