Special Block Tower

Problem Description

Bunny Ji finds a special tower that always keeps the biggest block on top. The tower starts empty. You will get a list of commands telling Bunny Ji what to do.

Here are the operations:

1. **Add a Block (`0 x`)**: Put a block with number x into the tower. The tower automatically rearranges itself, so the biggest block is always on top (maintaining the max-heap property).
2. **Battle and Merge (`1`)**: Remove the two largest blocks from the tower and fight: new block = first block - second block. The new block is placed back in the tower (even if the value is 0). The tower then re-balances to maintain the max-heap property.
3. **Change a Block (`2 idx delta`)**: Find a block at index `idx` (0 means the top block), and change its value by `delta`. After the adjustment, the tower automatically re-balances to maintain the max-heap property.

After all operations are performed, output the contents of the heap in **level-order**.

Since duplicate keys are allowed, sift-up swaps only when child > parent. For sift-down, pick the larger child (ties go to the left) and swap only if that child > parent.

Input Format

The first line contains a single integer N, representing the total number of operations to be performed on the heap. The following N lines each describe one operation. There are three possible operation types:

- `0 x`: Insert the integer x into the heap.
- `1`: Remove the two largest integers from the heap, subtract the second from the first, and insert the result back into the heap.
- `2 idx delta`: Increase the value at index `idx` (0-based) in the heap array by `delta` (or decrease if `delta` is negative), and then re-balance the heap to maintain the max-heap property.

It is guaranteed that every operation is valid. For example, operation `1` will only occur when there are at least two elements in the heap, and operation `2` will only occur when `idx` is a valid position in the heap.

Output Format

Output the final state of the heap in level-order after performing all the operations specified in the input. Numbers should be separated by a space.

Constraints

- $1 \leq N \leq 2.5 \times 10^5$
- $|x| \leq 10^9$
- $|delta| \leq 10^9$

Example Test Case

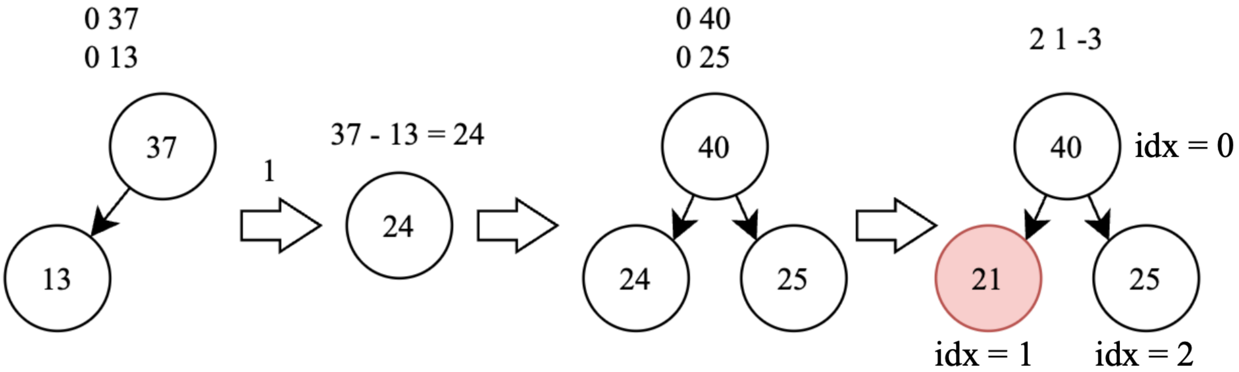
Sample Input 1

```
8
0 37
0 13
1
0 40
0 25
2 1 0
2 2 0
2 1 -3
```

Sample Output 1

```
40 21 25
```

Explanation of Sample 1



Sample Input 2

```
4
0 42
0 0
0 24
1
```

Sample Output 2

```
18 0
```

Explanation of Sample 2

- 0 42: Add block 42 → tower: [42]
- 0 0: Add block 0 → tower: [42, 0]
- 0 24: Add block 24 → tower reorders →tower: [42, 0, 42]
- 1: Remove 42 and 24 → $42 - 24 = 18$ → tower reorders →tower: [18, 0]

Sample Input 3

```
3
0 25
0 22
2 0 -4
```

Sample Output 3

```
22 21
```

Explanation of Sample 3

- 0 25: Add block 25 → tower: [25]
- 0 22: Add block 22 → tower: [25, 22]
- 2 0 -4: Decreases block at index 0 (25) by 4 → tower: [22, 21]