FabricBreakup Puzzle

Problem Statement:
You just had a breakup and your ex has dumped all your shirts in a big heap in the centre of your tiny hostel room. And man! These are way too many shirts. No wonder (s)he was fed up helping you choose the best among them for every date :).

Anyways, whenever you find time, you start arranging the shirts from the big heap one by one. We will call the arranged shirts (that are no longer in the heap) as good shirts. Since you have limited space in your hostel room, you arrange the good shirts in a big pile (tower) on a side.

But, of course, every now and then you also want to party with friends to get over your ex. And at any time you go out, you wish to wear your most favourite shirt from among the good shirts in the pile. Unfortunately, you are clumsy! So, in trying to take out that chosen shirt, all shirts on top of that shirt get toppled and get mixed in the big heap again. Your goal is to programmatically find how many shirts get toppled every time you party with friends.

Input:
The input to your software is a sequence of operations. The first line is the total number of operations N. And every successive line is operation #i with i ranging from 1 to N. Each operation is action id 1 or 2. Id 1 means a move from heap to pile. Id 2 means party with friends. For move action there is one additional number, a score which represents how much you like that shirt. It is an integer between 0 and 10000, where 10000 means max liking and 0 means hate it! As an example suppose you perform 8 operations: move shirt with score 400, move shirt with score 500, party, move shirt with score 300, party, party, move shirt with score 20, party. The input will be:

```
8
```

```
1 1 400
2 1 500
3 2
4 1 300
5 2
6 2
7 1 20
8 2
```

Output:
The output is, for each operation #, which is a party operation, return the number of shirts that get toppled back into the heap. In case there are multiple shirts with the same best score in the pile, you always pick that one which topples the least number of shirts. Also, if you try to party without any shirts in the pile you should output -1 to indicate error. In the example above, the output will be:

```
3 0
5 1
6 -1
8 0
```

That is, when you party first, you pick a shirt with score 500 and, since it's at top, no other shirt gets toppled. Second time, you topple one shirt and pick the shirt with score 400 to wear. You are out of shirts the third time you try to party the third time, and so on.