Flipping Burned Pancakes

The cook at the *Frobbozz Magic Pancake House* sometimes falls asleep on the job while cooking pancakes. As a result, one side of a stack of pancakes is often burned. Clearly, it is bad business to serve visibly burned pancakes to the patrons. Before serving, the waitress will arrange the stacks of pancakes so that the burned sides are facing down. You must write a program to aid the waitress in stacking the pancakes correctly.

We start with a stack of *N* pancakes of distinct sizes, each of which is burned on one side. The problem is to convert the stack to one in which the pancakes are in size order with the smallest on the top and the largest on the bottom and burned side down for each pancake. To do this, we are allowed to flip the top *k* pancakes over as a unit (so the *k-th* pancake is now on top and the pancake previously on top is now in the *k-th* position and the burned side goes from top to bottom and vice versa).

For example (+ indicates burned bottom, - a burned top):

> +1 -3 -2 [flip 2] ⇒ +3 -1 -2 [flip 1] ⇒ -3 -1 -2 [flip 3] ⇒
> +2 +1 +3 [flip 1] ⇒ -2 +1 +3 [flip 2] ⇒ -1 +2 +3 [flip 1] ⇒ +1 +2 +3

You must write a program which finds a sequence of at most *(3n – 2)* flips, which converts a given stack of pancakes to a sorted stack with burned sides down.

Input

The first line of the input contains a single decimal integer, N, the number of problem instances to follow. Each of the following N lines gives a separate dataset as a sequence of numbers separated by spaces. The first number on each line gives the number, M, of pancakes in the data set. The remainder of the data set is the numbers 1 through M in some order, each with a plus or minus sign, giving the initial pancake stack. The numbers indicate the relative sizes of the pancakes and the signs indicate whether the burned side is up (-) or down (+). M will be, at most, 30.

Output

For each dataset, you should generate one line of output with the following values: The dataset number as a decimal integer (start counting at one), a space, the number of flips (K, where K >= 0) required to sort the pancakes and a sequence of K numbers, each of which gives the number of pancakes to flip on the corresponding sorting step. There may be several correct solutions for some datasets. For instance 3 2 3 is also a solution to the first problem below.

Example

Input:
```
3
3 +1 –3 –2
4 –3 +1 –2 –4
5 +1 +2 +3 +4 -5
```

Output:

```
1 6 2 1 3 1 2 1
2 6 4 1 4 3 1 2
3 3 5 1 5
```