Inverting Stacks (B)

Code the following function:

```
template <typename T>
void Stack<T>::invertBase(int m);
```

which receives an integer value `m` ($0 \leq m \leq s.size()$) and modifies the stack, so that the m values on top of the stack remain in the same order, and the $p.size() - m$ values at the bottom of the stack are inverted.

To solve this exercise, you must use the dynamic implementation of the stacks based on the use of linked lists. You cannot use auxiliary ADTs or arrays. You cannot allocate nor release dynamic memory (i.e. you cannot use the operators **new** or **delete**)

Input

The input has several test cases in separate lines. Each test case contains the number of elements of the stack, the value of m and the values stored in the stack, starting from the element at the top. The input ends when the number of elements in the stack is -1.

Output

For each test case, the output must be the elements of the stack (separated by blanks), starting from the element at the top, after inverting the m elements at the bottom.

Sample input

```
3 0 1 2 3
3 1 1 2 3
3 2 1 2 3
3 3 1 2 3
-1
```

Sample output

```
3 2 1
1 3 2
1 2 3
1 2 3
```

Notes

This exercise must be understood in the context of the *Data Structures and Algorithms* course, FDI-UCM 2016/2017 (prof. Gonzalo Méndez). Therefore, the only valid solutions are those that use the concepts studied in this course. Additional remarks may be provided in class.