A *Binary Search Tree (BST)* is a tree data structure, which obeys the following properties:

- Each nodes has at most 2 children.
- Each node except the root has exactly one parent.
- There is exactly one root node. The root node does not have any parents.
- If a node has a left child, all values in the left subtree are less than or equal to the value of the value of the node itself.
- If a node has a right child, all values in the right subtree are strictly greater than the value of the node itself.

A kuPellaKeS BST is a BST which have the following properties:

- The difference between sum of all values in the left subtree and sum of all values in the right subtree is minimum.
- If there is more than one way to minimize the difference, sum of all values in the left subtree is maximum.
- The right and left subtrees are also kuPellaKeS BST's.

The following rules describe a representation way for a BST:

- A BST with no subtrees is denoted by
  "value_of_root" (Quotes included for clarity).
- A BST which have only one subtree (Either left or right subtree), is denoted by
  "value_of_root(representation_of_subtree)" (Quotes included for clarity).
- A BST which have both of its subtrees, is denoted by
  "value_of_root(representation_of_left_subtree,representation_of_right_subtree)".

Given a list of integer numbers, output the representation of the kuPellaKeS BST having those numbers as node values.

Input

The first line of input gives the number of cases, T. Then, T test cases follow. Each one starts with a line containing number of values $1 \leq n \leq 1000$. Next line contains n integer numbers separated by a single space.

Output

For the x-th test case, your program must output the line containing '`Case #`x`:`', followed by representation of the kuPellaKeS BST having the given numbers as node values.

Sample Input

```
3
10
1 2 3 4 5 6 7 8 9 10
5
1 0 1 0 1
4
-1 -2 -3 -4
```

Sample Output

```
Case #1: 7(5(3(2(1),4),6),9(8,10))
Case #2: 1(1(1(0(0))))
Case #3: -3(-4,-2(-1))
```