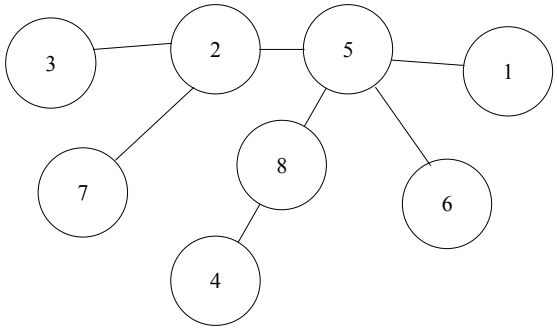
A Prüf Problem

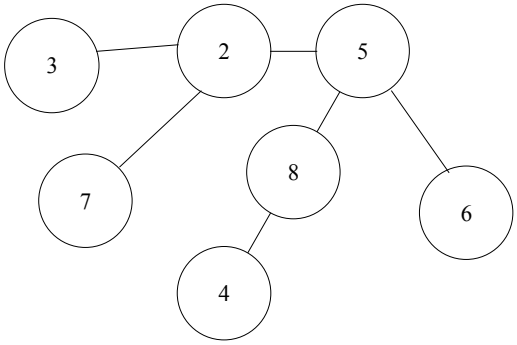
input file: `pruf.in` output file: `pruf.out`

Prüfer codes are used to represent labeled free trees. A free tree is a connected, acyclic, undirected graph. The labeling for a free tree with n nodes gives each node a label from 1 to n, using each value exactly one time. One possible labeled free tree with 8 nodes is:



A Prüfer code for a free tree with n nodes is a sequence of n-2 labels that completely represents the free tree.

To construct a Prüfer code from a free tree, first identify all leaves. A leaf is a node with only one adjacent node. In the tree above, the nodes 1, 3, 4, 6, and 7 are leaves. Then remove the leaf with the lowest label (in this case 1) and add the label of the node adjacent to it (in this case 5). This will give a new free tree:



Continue to find the leaf with the smallest label, remove it, and add the label of the node adjacent to it to the Prüfer code, until there are only two nodes left in the free tree. The Prüfer code for the original free tree is 5, 2, 8, 5, 2, 5.

Notice that in the Prüfer code, each node appears *k*-1 times, where *k* is the number of edges adjacent to a node. For example, there are 4 edges adjacent to node 5 in the tree and 5 appears in the Prüfer code 3 times. Therefore, any node that is a leaf in the original free tree will not appear in the Prüfer code.

You should write a program to generate the original tree from its Prüfer code.

Input

The input to your program will be a number of different Prüfer codes. Each code will take one line of data. The first integer on the line will be n, the number of nodes in the free tree. You may assume $3 \le n \le 20$. The next n-2 numbers will be the Prüfer code for the free tree.

The last line of input will consist of just value 0 for n. You should not process this line.

Output

For each input set, write a line with the number of the input set. Then, on the next line, list the edges (as ordered pairs) in the tree in free tree in canonical order. To canonically order a set of edges, arrange each as (i, j), where i and j are the nodes they connect and $i < j$. Then, to compare two edges (i, j) and (m, n), the edge (i, j) comes before (m, n) if and only if $i < m$, or $i = m$ and $j < n$.

Have one blank line after each output set.

Sample Input

```
3 2
8 5 2 8 5 2 5
5 3 3 5
0
```

Sample Output (corresponding to sample input)

```
Input set 1:
(1, 2) (2, 3)

Input set 2:
(1, 5) (2, 3) (2, 5) (2, 7) (4, 8) (5, 6) (5, 8)

Input set 3:
(1, 3) (2, 3) (3, 5) (4, 5)
```