# Problem H
Automatic Marking

You must be familiar with binary trees and all their operations, but this problem deals with a less popular structure, which I shall call *myTree*. There are three possible organizations of a *myTree*:

1. an empty tree. That is, a tree with no nodes.
2. a tree whose root node has a single data item, say K, and two children. Each of its two children is a *myTree*. Any values in the left subtree are less than or equal to K, and any values in the right subtree are larger than K.
3. a tree whose root node has two data items, say K_1 and K_2 , and three children. $K_1 < K_2$. Each of its three children is a *myTree*. Any values in the left subtree are less than or equal to K_1, any values in the middle subtree are larger than K_1 and smaller than or equal to K_2 , and any values in the right subtree are larger than K_2.

All internal (non-terminal) nodes have two or three children, although some may be empty. One way to represent such a tree is to use level-order traversal, starting at the root node, with the content of each node enclosed in parentheses. An empty tree is represented by a pair of parentheses that encloses nothing. The following figure demonstrates an example of such a tree, along with its representation, with values in the nodes being uppercase characters chosen in the range of "A" to "Z", inclusive.
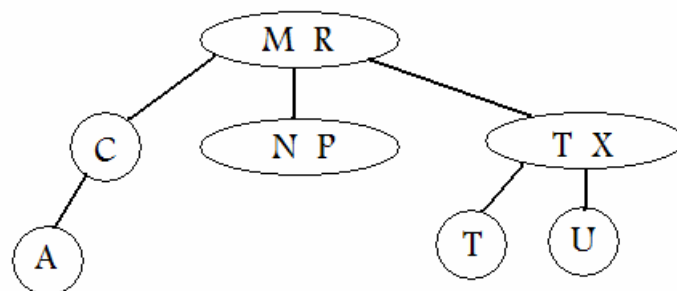
(M R) (C) (N P) (T X) (A) () () () () (T) (U) () () () () () () ()

A lecturer of "Data Structures 101" likes to test her students understanding of the *myTree* structure by asking them to identify all possible ways to assign a missing value in a given *myTree*.

Examples of such a question would be:

1. a tree "(M R) (C) (N P) (? U) () () () () () () () ()", for which the answer should be the letters "S" and "T".
2. a tree "(M R) (X) (N O) (? U) () () () () () () () ()", for which the answer should be "This is not a *myTree*". The reason is that X > M.
3. a tree "(M R) (N O) (? U)", for which the answer should be "This is not a myTree". The reason is that nodes with two values should have three children in a *myTree* structure, which is violated in this question.

Your task is to write a program to answer such a question.

INPUT:

The input contains descriptions of a number of *myTree structures* to be processed. The information for each tree is given in a single line as a series of properly matched parentheses. Each pair of matched parentheses encloses zero, one, or two characters selected from uppercase letters in the range of "A" to "Z" and "?". Each line contains exactly one "?". The selected characters and parentheses are separated by single spaces.

The input is terminated by a line of a set of matched parentheses, which encloses a zero, for which no output is to be produced.

OUTPUT:

For each input tree the output is a single line with:

1. a listing of possible uppercase letters sorted in increasing order that can replace the "?" in the given tree, or
2. the string "This is not a myTree", if the given data does not conform to the given specifications.

EXAMPLE INPUT:
```
( M R ) ( C ) ( N P ) ( ? U ) ( ) ( ) ( ) ( ) ( ) ( ) ( )
( M R ) ( X ) ( N P ) ( ? U )
( M R ) ( C ) ( ? U )
( M R ) ( C ) ( N P ) ( ? X ) ( A ) ( ) ( ) ( ) ( ) ( S ) ( U ) ( ) ( ) ( ) ( ) ( ) ( ) ( )
( 0 )
```

EXAMPLE OUTPUT:
```
S T
This is not a myTree
This is not a myTree
S T
```