Boolean Function Reconstruction

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

Given the truth table of a boolean function with n boolean variables as input, construct an expression that satisfies this function. In the expression, you are only allowed to use the logical `and` (`&`) and logical `or` (`|`) operators.

Specifically, a truth table of a boolean function with n boolean variables gives all the 2^n outputs corresponding to the possible values of n input variables. A boolean expression `<expr>` has the following forms:

- `T`, `F`: Represents `True` and `False`.
- `a`, `b`, ..., `z`: Represents one of the variables. The i-th variable is represented by the i-th lowercase letter in alphabetical order.
- `(<expr>&<expr>)`: Represents the logical `and` operation applied to the results of two expressions.
- `(<expr>|<expr>)`: Represents the logical `or` operation applied to the results of two expressions.

The logical `and` operation and the logical `or` operation are defined as two boolean functions below that take two boolean values.

x_1	x_2	$x_1 \& x_2$	$x_1 \mid x_2$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Determine whether an expression exists that satisfies the conditions. If such an expression exists, ensure that the number of binary operators (`&` and `|`) does not exceed $2^{n-1} + 10$, and the depth of parentheses nesting does not exceed 100 layers.

It can be proven that if a solution exists, there is always one that meets the constraints of the problem.

Input

The input consists of multiple test cases. The first line contains an integer T ($1 \le T \le 2^{16}$), the number of test cases. For each test case, there are two lines:

- The first line contains an integer n ($1 \le n \le 15$).
- The second line contains a binary string s with length 2^n, indicating the truth table of the given function.

To interpret the input binary string, suppose the i-th variable has a value of x_i. Then, the corresponding function value, $f(x_1, x_2, \ldots, x_n)$, is equal to the $\left(\sum_{i=1}^{n} x_i \cdot 2^{i-1} + 1\right)$-th bit of the string s.

It is guaranteed that the sum of 2^{2n} over all test cases will not exceed 2^{30}.

Output

For each test case:

- Output `Yes` or `No` on the first line to indicate whether an expression satisfying the conditions exists.
- If an expression exists, output the expression on the second line. The expression must strictly adhere to the format given in the problem description, **without adding or omitting parentheses, and without adding extra spaces**.

Example

standard input	standard output
7 2 0001 2 0111 2 1111 3 00010111 1 10 2 0101 5 00000000000000000000000000000001	Yes (a&b) Yes (a|b) Yes T Yes ((a&(b|c))|(b&c)) No Yes a Yes (a&(b&(c&(d&e))))

Note

Below is the truth table interpretation for the fourth sample.

x_3	x_2	x_1	$f(x_1, x_2, x_3)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1