Introduction to Algorithms	**Problem Set 4**
CS 4820, Spring 2019	**Due 11:59pm Friday, March 1**

Hand in your solutions electronically using CMS. Each solution should be submitted as a separate file. Collaboration is encouraged while solving the problems, but:

1. **list the names of those with whom you collaborated;**
2. **you must write up the solutions in your own words;**
3. **you must write your own code.**

Remember that when a problem asks you to design an algorithm, you must also prove the algorithm's correctness and analyze its running time. The running time must be bounded by a polynomial function of the input size.

(1) Parentheses *(10 points)*
You are helping out a sloppy friend with a math problem. He has written an expression of the form $a_1O_1a_2O_2\dots O_{n-1}a_n$, where each a_i is either FALSE denoted by 0 or TRUE denoted by 1, and each O_i is the logical OR denoted by $\vee$ or the logical XOR denoted by $\oplus$. Unfortunately, he has forgotten to insert in parentheses in the expression which leads you to think of the following problem: how many ways of parenthesizing the expression leads the evaluation to be 0? Develop an efficient algorithm that takes as input an expression E of the form $a_1O_1a_2O_2\dots O_{n-1}a_n$ and outputs the number of ways of parenthesizing E such that it evaluates to 0 (i.e, FALSE).

Example: Let the input expression E be: $0 \vee 1 \oplus 1 \vee 1$. Below are all the ways of parenthesizing E:

- $(((0 \vee 1) \oplus 1) \vee 1)$ which evaluates to 1.
- $((0 \vee (1 \oplus 1)) \vee 1)$ which evaluates to 1.
- $((0 \vee 1) \oplus (1 \vee 1))$which evaluates to 0.
- $(0 \vee ((1 \oplus 1) \vee 1))$ which evaluates to 1.
- $(0 \vee (1 \oplus (1 \vee 1)))$ which evaluates to 0.

Thus the output of your algorithm on this instance should be 2.