**Problem #1**

**Count Valid Parenthesization**

An integer, *n*, is given as input. Output the number of valid parenthesizations that can be formed by using *n* pair of parentheses (i.e. pair of 1st brackets). For example, 5 valid parenthesizations can be formed using 3 pair of parentheses; namely “((()))”, “(()())”, “(())()”, “()(())” and “()()()”.

**Do not use arrays for this task.** Use a recursive method with the following prototype:

int validParenthesizationCount(int n)

|  |  |
| --- | --- |
| **Sample Input(s)** | **Corresponding Output(s)** |
| 1  2  3  4  6  8 | 1  2  5  14  132  1430 |

**Problem #2**

**Count Binary String**

A binary string is a string containing only two characters: ‘0’ and ‘1’. For this problem, you are given an integer, *n*, as input. Output the number of binary strings of length *n*, that do not contain three or more consecutive 1’s. For example, there are 16 binary strings of length 4. Three of them, namely “0111”, “1110” and “1111” contain three or more consecutive 1’s. So there are 13 binary string of length 4, that do not contain three or more consecutive 1’s.

**Arrays cannot be used for this task.** Use a recursive method with the following prototype:

int binaryStringCount(int n)

|  |  |
| --- | --- |
| **Sample Input(s)** | **Corresponding Output(s)** |
| 1  2  3  4  6  8 | 2  4  7  13  44  149 |