1. What is the BigO upper bound of the code snippet below as a function of n assuming that the Func1(n) is O(n)?

   ```c
   int j = n;
   while (j >= 1)
   {
       for (int i = 1; i <= j; i++)
       {
           int val = Func1(n);
       }
       j = j / 2;
   }
   ```

2. Determine the BigO upper bound as a function of n for the following code snippet. Do this by finding a formula, g(n) for the number of times Func2() is called. Assume Func2 is completed in constant time, C. Prove your answer.

   ```c
   for (int i = 9; i <= n; i *= 3)
   {
       for (int j = -n; j < n; j++)
       {
           if (j % 2)
           {
               Func2(i, j);
           }
       }
   }
   ```
3. A Catalan number is defined by the following recursive formula:

\[ C_0 = 1 \quad \text{and} \quad C_{n+1} = \sum_{i=0}^{n} C_i \, C_{n-i} \quad \text{for} \ n \geq 0; \]

Write a recursive function which computes the nth Catalan number.
4. Assume a List class which is a doubly linked-list of ordered nodes. The node and class is defined as follows:

```cpp
Class List
{
public:
  List();
  ~List();
  ... member functions ...
  bool InsertNewItem(Item &it);
private:
  struct Node
  {
    Item *pItem;
    Node *next;
    Node *prev;
  }
  Node *head;
  Node *tail;
}
```

Write the member function `InsertNewItem` which takes an item by reference and inserts it into the proper place in the list. Duplicates are not allowed. Return true if the item was inserted; false if the item already is in the list and does not need to be inserted.