1. For each of the following recursive functions:

- (1 point) Describe what the function computes (careful, some of these are tricky!)
- (1 point) Give a recurrence relation that describes the running time of the function (Give both base and recursive cases)
- (2 point) Solve the recurrence to get a $\Theta$ running time for the function. Use either the repeated substitution method, or the recursion tree method (which is essentially the same as the repeated substitution method, just a little more graphical). *Do not* use the master method for this question (you will have a chance to use the master method on later questions!)

(a) `int recursive1(int n)`
{
    if (n == 0)
        return 0;
    else
        return 1 + recursive1(n-1);
}

(b) `int recursive2(int n)`
{
    if (n == 0)
        return 1;
    else
        return recursive2(n-1) + recursive2(n-1);
}

(c) `int recursive3(int n)`
{
    if (n == 0)
        return 1;
    else
        return 2 * recursive3(n-1);
}

(d) `int recursive4(int n)`
{
    int no_op;
    if (n > 1)
    {
        for (int i = 0; i < n; i++)
        {
            no_op++;
        }
        return recursive4(n/2) * recursive4(n/2);
    }
    else
    {
        return 0;
    }
}
2. Use the substitution method (that is, proof by induction) to prove the following bounds:

(a) (3 points) $O(n^2)$ bound for:

\[
\begin{align*}
T(0) & = C_1 \\
T(1) & = C_1 \\
T(n) & = T(n - 2) + C_2 n
\end{align*}
\]