Name: Muhammad Ishraf Shafiq Zainuddin

ID: 200342741

Assignment: 3

1. The running times predicted by the detailed model of the computer and the simplified model of the computer for each of the following program fragments:

(a). Run Time : O (n - 1).

```
int mode (int n)
{
    int i = 1,
    int k;
    if ( n = = 1)
    {
        return i;
    }
    for ( k = 1; k < n; k++)
    {
        i = i + mode (k) + mode (n - k);
    }
    return x;
}
```

(b) Run Time : O (n - 1).

```
int mode (int n)
{
    if ( n ≤ 1)
    {
        return 1;
    }
    else if ( n % 2 = = 0)
    {
        return ( n / 2)
    }
    else
    {
        return (3n - 1)
    }
}
```

2. Prove by induction the following summation formulas:

Let b(n) = formula;

When n = 1;

L.H.S = 1^2 = 1

R.H.S = (1(1 + 1)(2 + 1)) / 6 = 6 / 6 = 1.

- L.H.S = R.H.S

Hence, b(1) is true.

3. Solve the following recurrences by repeated substitution:

T (0) = 1 , T (n) = T (n - 1) + 1 , n > 0;

T (1) = T (0) + 1 = 1 + 1 = 2;

T (2) = T (1) + 1 = 2 + 1 = 3;

- T (n) = T (n - 1) + 1 = n + 1
- T (n) = n + 1.

4. Consider the function $f(n) = 3n^2 - n + 4$. Using Definition show that $f(n) = O(n^2)$.

By Definition, there is a positive real number (c) and positive integer (N) exist;

$[f(n) <= c*g(n)$ for all $n >= N]$

- $f(n) = 3n^2 - n + 4$, for $n >= 4$;

  $3n^2 - n + 4 <= 3n^2 - n + n <= 3n^2$

- $f(n) <= 3n^2$;

  Let $c = 3$, $N = 4$, $g(n) = n^2$

- Hence, $f(n) = O(n^2)$.

5. Consider the function $f(n) = 3n^2 - n + 4$. Using Definition show that $f(n) = \Omega(n^2)$

By Definition, there is a positive real number (c) and positive integer (N) exist;

$[f(n) <= c*g(n)$ for all $n >= N]$

- $f(n) = 3n^2 - n + 4$;

  $4 - n >= 0$;

  $3n^2 >= n^2$, for $n >= 0$;

- $3n^2 - n >= n^2$;

  $3n^2 - n + 4 >= n^2$

- $f(n) >= n^2$;

  Let $c = 1$, $N = 0$, $g(n) = n^2$;

- Hence, $f(n) = \Omega(n^2)$;

6. (a) Write a recursive version of the function Fibonacci;

```
if ( n = = 0)
    return 0;
else if ( n = = 1)
    return 1;
else
    return (RecursiveFibonacci ( n - 1) + RecursiveFibonacci ( n - 2));
```

6. (b) Write a non recursive version of the function Fibonacci

```
int a = 0;
int b = 1;
int fib;
for (int count = 0; count < n; count ++)
{
    if (count <= 1)
        fib = count;
    else
    {
        fib = a + b;
        a = b;
        b = fib;
    }
    cout << " " << fib << " ";
```