Problem 6

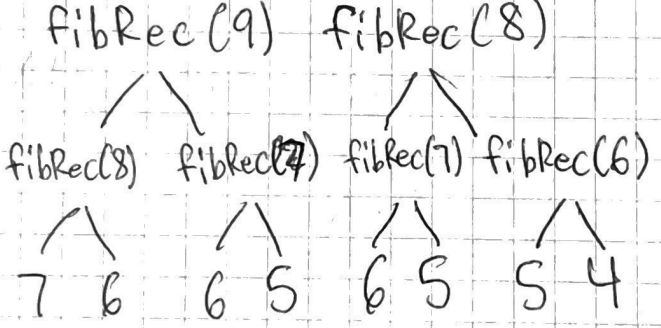
Derive the order for the Recursive and non-Recursive Fibonacci Function.

Recursive:

```
int fibRec(int n) {
  if (n<=0) return 0;
  if (n==1) return 1;
  return fibRec(n-1) + fibRec(n-2);
}
```

$T(n) = T(n-1) + T(n-2) + O(1)$

Let $n = 10$



=> Each function call, fibRec gets called twice so function increases exponentially,

$$\boxed{T(n) = O(2^n)}$$

Non-Recursive;

```
int fibLoop(int n) {
    if (n<=0) return 0;
    if (n==1) return 1;
    int fib1=1, fib2=0, fi=fib1+fib2;
    for(int i=2; i<n; i++) {
        fib2=fib1;
        fib1=fi;
        fi=fib1+fib2;
    }
    return fi;
}
```

Non-Recursive has a lot of O(1) statements so they don't depend on input size, The for loop goes to size N so time complexity: $\boxed{T(n)=O(N)}$