COMP 322 Written Assignment

1.1 Recursive Fibonacci

1.1.1 Recursive Fibonacci Complexity.

n	fib(n)	WORK(n)
0	0	1
1	1	1
2	2	3
3	3	5
4	5	9
5	8	15
6	13	25

$Work(k) = Work(k-1) + Work(k-2) + 1.$

So $\boxed{WORK(n) = 2fib(n+1) - 1}$, and $Work(k) = Work(k-1) + Work(k-2) + 1$.

According to Binet's formula, we know $fib(n+1)$

$$= \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^{n+1} - \left(\frac{1-\sqrt{5}}{2}\right)^{n+1}\right)$$

$$WORK(n) = \frac{2}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^{n+1} - \left(\frac{1-\sqrt{5}}{2}\right)^{n+1}\right) - 1$$

Prove by reduction:

Base case: $Work(0) = 1$, $2fib(1) - 1 = 2 - 1 = 1$, so $Work(0) = 2fib(1) - 1$

Recursive case: Assume $n = 0, 1, 2, 3, \ldots, k-1$ holds true for $Work(n) = 2fib(n+1) - 1$,

we want to show that $Work(k) = 2fib(k+1) - 1$.

Proof: $2fib(k+1) - 1 = 2fib(k+1-1) + 2f(k+1-2) - 1$

$= (2fib(k) - 1) + (2fib(k-1) - 1) + 1$

$= Work(k-1) + Work(k-2) + 1 = Work(k)$

Thus proved that $Work(n) = 2fib(n+1) - 1$

1.1.2

- Because memorization allows the result of each call to fib recorded, when we call fib(n) for the first time, we perform fib(0), fib(1), ..., fib(n-1) only once, so we need to call fib for n times. Since each call to fib has a total Work of 1. it takes $\boxed{O(n)}$ work to call fib(n).

- Since fib(k_1), fib(k_2), ..., fib(k_m), $k \in [0, MaxMemo]$ are already called, their results are recorded. So when we call fib(n), $n < MaxMemo$, we can get the result from the previously recorded work, thus it only takes $\boxed{O(1)}$ of Work.