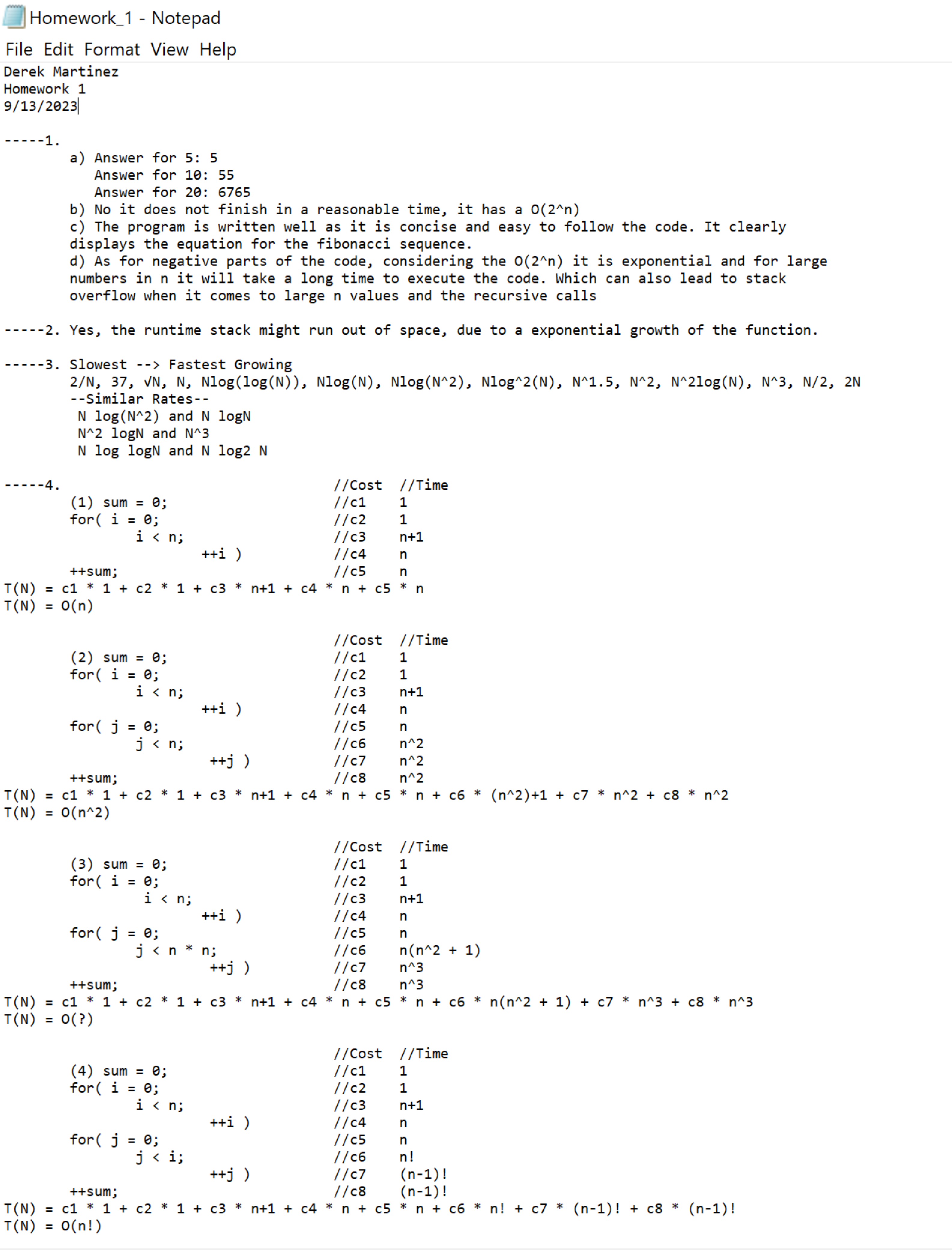

```
Derek Martinez
Homework 1
9/13/2023

-----1.
        a) Answer for 5: 5
           Answer for 10: 55
           Answer for 20: 6765
        b) No it does not finish in a reasonable time, it has a O(2^n)
        c) The program is written well as it is concise and easy to follow the code. It clearly
        displays the equation for the fibonacci sequence.
        d) As for negative parts of the code, considering the O(2^n) it is exponential and for large
        numbers in n it will take a long time to execute the code. Which can also lead to stack
        overflow when it comes to large n values and the recursive calls

-----2. Yes, the runtime stack might run out of space, due to a exponential growth of the function.

-----3. Slowest --> Fastest Growing
        2/N, 37, √N, N, Nlog(log(N)), Nlog(N), Nlog(N^2), Nlog^2(N), N^1.5, N^2, N^2log(N), N^3, N/2, 2N
        --Similar Rates--
         N log(N^2) and N logN
         N^2 logN and N^3
         N log logN and N log2 N

-----4.                                 //Cost  //Time
        (1) sum = 0;                    //c1    1
        for( i = 0;                     //c2    1
                i < n;                  //c3    n+1
                        ++i )           //c4    n
        ++sum;                          //c5    n
T(N) = c1 * 1 + c2 * 1 + c3 * n+1 + c4 * n + c5 * n
T(N) = O(n)

                                        //Cost  //Time
        (2) sum = 0;                    //c1    1
        for( i = 0;                     //c2    1
                i < n;                  //c3    n+1
                        ++i )           //c4    n
        for( j = 0;                     //c5    n
                j < n;                  //c6    n^2
                         ++j )          //c7    n^2
        ++sum;                          //c8    n^2
T(N) = c1 * 1 + c2 * 1 + c3 * n+1 + c4 * n + c5 * n + c6 * (n^2)+1 + c7 * n^2 + c8 * n^2
T(N) = O(n^2)

                                        //Cost  //Time
        (3) sum = 0;                    //c1    1
        for( i = 0;                     //c2    1
                 i < n;                 //c3    n+1
                        ++i )           //c4    n
        for( j = 0;                     //c5    n
                j < n * n;              //c6    n(n^2 + 1)
                         ++j )          //c7    n^3
        ++sum;                          //c8    n^3
T(N) = c1 * 1 + c2 * 1 + c3 * n+1 + c4 * n + c5 * n + c6 * n(n^2 + 1) + c7 * n^3 + c8 * n^3
T(N) = O(?)

                                        //Cost  //Time
        (4) sum = 0;                    //c1    1
        for( i = 0;                     //c2    1
                i < n;                  //c3    n+1
                         ++i )          //c4    n
        for( j = 0;                     //c5    n
                j < i;                  //c6    n!
                         ++j )          //c7    (n-1)!
        ++sum;                          //c8    (n-1)!
T(N) = c1 * 1 + c2 * 1 + c3 * n+1 + c4 * n + c5 * n + c6 * n! + c7 * (n-1)! + c8 * (n-1)!
T(N) = O(n!)
```