ECE 5960-023/6960-025 – Advanced Programming for Computer Design Problems

In-class Practice 2 (due 2020/01/23 in class)

1. Report the difference in terms of the lines of assembly code between the recursive version and iterative version of the big-mod problems using compiler explorer https://godbolt.org/. Did you see any difference when the optimization flag -O2 is enabled?

2. Based on the "*empirical analysis*" we taught in class, answer the following questions:
   - ❑ **A Problem with N = 1000, Time Limit = 1s**
       - ❑ Could a O(N) algorithm pass the time limit constraint?
       - ❑ What about the O(NlogN)?
       - ❑ What about the $O(N^2)$?
       - ❑ What about the $O(N^2 \log N)$?
       - ❑ What about the O(1) time ?
   - ❑ **A Problem with N = 1000000, Time Limit = 1s**
       - ❑ Could a O(N) algorithm pass the time limit constraint?
       - ❑ What about the O(logN) ?
       - ❑ What about the O(logN logN) ?
       - ❑ What about the $O(N^2)$?

3. What is the big-O complexity of the following code in computing Fibonacci numbers?
```
int fib(int n)
{
    if (n <= 1)
        return n;
    return fib(n-1) + fib(n-2);
}
```
A common solution to avoid duplicate computations in the above code is the "tabular method" we discussed in class. What is the big-O complexity of the tabular method in computing Fibonacci numbers?

4. Finish the Hanoi problem (hanio.cpp) using the tabular method and report the runtime difference from the one without tabular method in passing all testcases.

Name: uid:

5. What is the advantage of using Divide and Conquer to solve the problem of finding a minimal value of a sequence, as we taught in the class?

6. Finish the 2D maximal point finding problem using divide and conquer (`divide_and_conquer` function in maximal-points.cpp). Make sure your solution is correct first, and then report the following:

N	Brute-Force Runtime (ms)	Divide-and-Conquer Runtime (ms)
10		
100		
1000		
10000		
100000		

Note:

To compile a .cpp program: `g++ simple.cpp -O2 -o simple`
To feed a program with a test file from the standard output: `./simple < test.txt`
To measure the runtime of a program: `time -p ./simple < test.txt`