

**CS 325 Fall 16**  
**Homework Assignment 1**

The following problems are from the 3<sup>rd</sup> edition of Introduction to Algorithms, CLRS. Attempt to solve these problems independently and then discuss the solutions in your Homework discussion groups. Submit a “professional” looking individual solution in Canvas. A subset of the problems will be graded for correctness.

- 1) (CLRS) 1.2-2. Suppose we are comparing implementations of insertion sort and merge sort on the same machine. For inputs of size  $n$ , insertion sort runs in  $8n^2$  steps, while merge sort runs in  $64n \lg n$  steps. For which values of  $n$  does insertion sort beat merge sort?

**Note:**  $\lg n$  is  $\log$  “base 2” of  $n$  or  $\log_2 n$ . There is a review of logarithm definitions on page 56. For most calculators you would use the change of base theorem to numerically calculate  $\lg n$ .

- 2) (CLRS) Problem 1-1 on pages 14-15. Fill in the given table. Hint: It may be helpful to use a spreadsheet or Wolfram Alpha to find the values.
- 3) (CLRS) 2.3-3 on page 39. Use mathematical induction to show that when  $n$  is an exact power of 2, the solution of the recurrence

$$T(n) = \begin{cases} 2, & \text{if } n = 2 \\ 2T\left(\frac{n}{2}\right) + n, & \text{if } n = 2^k, \text{ for } k > 1 \end{cases}$$

is  $T(n) = n \lg n$ .

- 4) For each of the following pairs of functions, either  $f(n)$  is  $O(g(n))$ ,  $f(n)$  is  $\Omega(g(n))$ , or  $f(n) = \Theta(g(n))$ . Determine which relationship is correct and explain.

- |                        |                    |
|------------------------|--------------------|
| a. $f(n) = n^{0.75}$ ; | $g(n) = n^{0.5}$   |
| b. $f(n) = n$ ;        | $g(n) = \log^2 n$  |
| c. $f(n) = \log n$ ;   | $g(n) = \lg n$     |
| d. $f(n) = e^n$ ;      | $g(n) = 2^n$       |
| e. $f(n) = 2^n$ ;      | $g(n) = 2^{n-1}$   |
| f. $f(n) = 2^n$ ;      | $g(n) = 2^{2^n}$   |
| g. $f(n) = 2^n$ ;      | $g(n) = n!$        |
| h. $f(n) = n \lg n$ ;  | $g(n) = n\sqrt{n}$ |

- 5) Design an algorithm that given a list of  $n$  numbers, returns the largest and smallest numbers in the list. How many comparisons does your algorithm do in the worst case? Instead of asymptotic behavior suppose we are concerned about the coefficients of the running time, can you design an algorithm that performs at most  $1.5n$  comparisons? Demonstrate the execution of the algorithm with the input  $A = [9, 3, 5, 10, 1, 7, 12]$ .

**CS 325 Fall 16**  
**Homework Assignment 1**

6) Let  $f_1$  and  $f_2$  be asymptotically positive functions. Prove or disprove each of the following conjectures. To disprove give a counter example.

a. If  $f_1(n) = O(g_1(n))$  and  $f_2(n) = O(g_2(n))$  then  $f_1(n) + f_2(n) = O(g_1(n) + g_2(n))$  .

b. If  $f_1(n) = O(g_1(n))$  and  $f_2(n) = O(g_2(n))$ , then  $\frac{f_1(n)}{f_2(n)} = O\left(\frac{g_1(n)}{g_2(n)}\right)$

7) **Fibonacci Numbers:**

The Fibonacci sequence is given by : 0, 1, 1, 2, 3, 5, 8, 13, 21, ..... By definition the Fibonacci sequence starts at 0 and 1 and each subsequent number is the sum of the previous two. In mathematical terms, the sequence  $F_n$  of Fibonacci number is defined by the recurrence relation

$$F_n = F_{n-1} + F_{n-2} \text{ with } F_0=0 \text{ and } F_1=1$$

An algorithm for calculating the  $n^{\text{th}}$  Fibonacci number can be implemented either recursively or iteratively.

**Example Recursive:**

```
fib (n) {  
    if (n = 0) {  
        return 0;  
    } else if (n = 1) {  
        return 1;  
    } else {  
        return fib(n-1) + fib(n-2);  
    }  
}
```

**Example Iterative:**

```
fib (n) {  
    fib = 0;  
    a = 1;  
    t = 0;  
    for(k = 1 to n) {  
        t = fib + a;  
        a = fib;  
        fib = t;  
    }  
    return fib;  
}
```

**CS 325 Fall 16**  
**Homework Assignment 1**

- a) Implement both recursive and iterative algorithms to calculate Fibonacci Numbers in the programming language of your choice. Provide a copy of your code with your HW pdf. We will not be executing the code for this assignment. You are not required to use the flip server for this assignment.
- b) Use the system clock to record the running times of each algorithm for  $n = 5, 10, 15, 20, 30, 50, 100, 1000, 2000, 5000, 10,000, \dots$ . You may need to modify the values of  $n$  if an algorithm runs too fast or too slow to collect the running time data. If you program in C your algorithm will run faster than if you use python. The goal of this exercise is to collect run time data. You will have to adjust the values of  $n$  so that you get times greater than 0.
- c) Plot the running time data you collected on graphs with  $n$  on the x-axis and time on the y-axis. You may use Excel, Matlab, R or any other software.
- d) What type of function (curve) best fits each data set? Again you can use Excel, Matlab, any software or a graphing calculator to calculate the regression curve. Give the equation of the function that best "fits" the data and draw that curve on the data plot. Why is there a difference in running times?