CSCI 4041, Spring 2019, Written Assignment 1
Due Tuesday, 1/29/19, 1:00 PM (submission link on Canvas)

Group17 member:
Thomas Tang - tang0376
Jackie Ji - ji000011
Song Liu - liux4169

This is a collaborative assignment; you may work in a group of 1-3 students. However, you may not consult or discuss the solutions with anyone other than the course instructor, the TAs, or the other members of your group, nor may you use material found from outside sources as part of your solutions. In addition, if you do choose to work in a group, each group member must participate in coming up with the solution to each problem and must be able to explain the group's answer if asked: dividing the problems amongst the group members is not acceptable.
+
Complete the following problems and submit your solutions in a single pdf file to the Written Assignment 1 submission link on Canvas. If you're working in a group, only one person should submit your answers, but make sure that you include the name and x500 of each group member at the top of the file, and that you are all in one of the Written Assignment Groups in Canvas. Typed solutions are preferred, but pictures or scans of a handwritten solutions in pdf form are acceptable so long as your solutions are clearly legible.

This assignment contains 4 problems, and each is worth 10 points, for a total of 40 points.

Your solutions to these problems must be clearly explained in a step-by-step manner; for most problems, the explanation will be worth far more points than the actual answer.

1. (Based on Problem 3-2) Indicate, for each pair of functions (f,g) in the table below, whether f(n) is O, Θ, or Ω of g(n). Your answer should be in the form of "yes" or "no" for each box; you do not need to show any other work.

f(n)	g(n)	f(n)=O(g(n))	f(n)=Θ(g(n))	f(n)=Ω(g(n))
5n - 78	3lg(n) + 2	Yes	No	No
$29n^2$	0.5*n!	No	No	Yes
7n + n*lg(n)	9n*lg(n) + 12	Yes	Yes	Yes
2^n	$100n^7$	Yes	No	No
$n^{1.5}$	n*lg(n)	Yes	No	No

2.

```
Mystery(num_ls)
1     diff = 0
```

```
2     for i = 1 to num_ls.len
3           for j = 1 to num_ls.len
4                 if(num_ls[i] - num_ls[j] > diff)
5                       diff = num_ls[i] - num_ls[j]
6     return diff
```

a. The algorithm above takes in as input an array of numbers, num_ls. Explain what the value it returns represents in a single sentence. You should assume that array indexing starts at 1.

It will return the biggest difference between two numbers in the array.

b. Analyze the runtime of each line of the algorithm, similar to the method shown in lecture and section 2.2 of the textbook, and then give a tight (big-Θ) bound on the asymptotic runtime of the algorithm.

Mystery(num_ls)	line cost	Time run
1 diff = 0	c_1	1
2 for i = 1 to num_ls.len	c_2	n+1
3 for j = i+1 to num_ls.len	c_3	n(n+1)
4 if(num_ls[i] - num_ls[j] > diff)	c_4	n^2
5 diff = num_ls[i] - num_ls[j]	c_5.	n^2
6 return diff	c_6.	1

Worst case runtime: $c_1+c_2+c_6+n(c_2+c_3)+n^2(c_3+c_4+c_5)$

$\Theta(n^2)$

c. Write pseudocode (or Python/Java) for another algorithm that has the same output for any given input, but is asymptotically faster than the above algorithm. Then give a tight (big-Θ) bound on the asymptotic runtime of the new algorithm.

```
Mystery(num_ls)
1     small = num_ls[0]
2     big = num_ls[0]
3     for i = 1 to num_ls.len
4           if(num_ls[i]>big)
5                 big = num_ls[i]
6           if(num_ls[i]<small)
7                 small = num_ls[i]
8     return big-small
```

$\Theta(n)$

3. (Problem 3-4f in the textbook) Let f(n) and g(n) be asymptotically positive functions. Prove or disprove the following: f(n) = O(g(n)) implies g(n) = Ω(f(n)). You **must** use the definition of big-O and big-Ω notation, given on pages 47 and 48 of the textbook, to construct your argument.

Given: f(n) = O(g(n)) which means there exists a positive c_1 and n_0 such that $0 \le f(n) \le c_1 g(n)$ for all $n \ge n_0$.

Ω (f(n)) = g(n) which needs there exists a positive c_2 and n_1 such that $0 <= c_2 f(n) <= g(n)$ for all $n >= n_1$.

Ask: f(n) = O(g(n)) implies g(n) = Ω(f(n))

Proof: $0 <= f(n) <= c_1 g(n)$

Divide c_1

$0 <= f(n)/c_1 <= g(n)$ for all $n >= n_0$

$0 <= f(n)(1/c_1) <= g(n)$ for all $n >= n_0$

$0 <= c_2 f(n) <= g(n)$ for all $n >= n_0$

c_2 can be (1/c1), which is a positive number.

$n_1 = n_0$

Therefore, f(n) = O(g(n)) which means there exists c_2 and n_1 such that $0 <= c_2 f(n) <= g(n)$ for all $n >= n_1$.

f(n) = O(g(n)) implies g(n) = Ω(f(n))

4. (Based on Exercise 2-1.1) Show the operation of insertion sort on the array A = [23, 9, 26, 1, 18, 4]. You must show the state of the array at the end of each iteration of the outer for loop, but no other work is required.

1: When j = 2, A = [9,23,26,1,18,4]
2: When j = 3, A = [9,23,26,1,18,4]
3: When j = 4, A = [1,9,23,26,18,4]
4: When j = 5, A = [1,9,18,23,26,4]
5: When j = 6, A = [1,4,9,18,23,26]