CIS 121 — Data Structures and Algorithms
Homework Assignment 3

Assigned: February 3, 2020 **Due:** February 10, 2020

Note: The homework is due **electronically on Gradescope** on February 10, 2020 by 11:59 pm ET. For late submissions, please refer to the Late Submission Policy on the course webpage. You may submit this assignment up to 2 days late.

A. **Gradescope**: You must select the appropriate pages on Gradescope. Gradescope makes this easy for you: before you submit, it asks you to associate pages with the homework questions. Forgetting to do so will incur a 5% penalty, which cannot be argued against after the fact.

B. **LaTeX**: You must use the LaTex template provided on the course website, or a 5% penalty will be incurred. Handwritten solutions or solutions not typeset in LaTex will not be accepted.

C. **Solutions**: Please write concise and clear solutions; you will get only a partial credit for correct solutions that are either unnecessarily long or not clear. Please refer to the Written Homework Guidelines for all the requirements.

D. **Algorithms**: Whenever you present an algorithm, your answer must include 3 separate sections.

1. A precise description of your algorithm in English. No pseudocode, no code.
2. Proof of correctness of your algorithm
3. Analysis of the running time complexity of your algorithm

E. **Collaboration**: You are allowed to discuss **ideas** for solving homework problems in groups of up to 3 people but *you must write your solutions independently*. Also, you must write on your homework the names of the people with whom you discussed. For more on the collaboration policy, please see the course webpage.

F. **Outside Resources**: Finally, you are not allowed to use *any* material outside of the class notes and the textbook. Any violation of this policy may seriously affect your grade in the class. If you're unsure if something violates our policy, please ask.

Note the following: $\lg n$ means $\log_2 n$.

1. [5 pts] Max on Max on Max

Let $f(n)$ and $g(n)$ be asymptotically non-negative functions. Prove that

$$\max\{f(n),\ g(n)\} = \Theta(f(n) + g(n))$$

Note that $\max\{f(n),\ g(n)\}$ refers to the maximum value at any given point in time, meaning that for some given value of $n = i$, we take the maximum of $\max\{f(i),\ g(i)\}$

2. [10 pts] Rush Big ΩΘO

For each of the parts below, list the functions in increasing asymptotic order. In some cases functions may be asymptotically equivalent (that is $f(n)$ is $\Theta(g(n))$). In such cases indicate this by writing $f(n) = g(n)$. When one is asymptotically strictly less than the other (that is, $f(n)$ is $O(g(n))$ but $f(n)$ is not $\Theta(g(n))$), express this as $f(n) < g(n)$. For example, given the set:

$$n^4 \qquad n \log n \qquad \frac{1}{4}n + n\log n,$$

the first function is $\Theta(n^4)$ and the other two are $\Theta(n \log n)$, and therefore the answer would be

$$n \log n \ = \ \frac{1}{4}n + n \log n \ < \ n^4.$$

You are *not* required to show your work; just writing your answer suffices.

(a)	$3^{(n/4)}$	$4^{(n/3)}$	$(4/3)^n$
(b)	$\lg n$	$\ln n$	$\lg(n^{100})$
(c)	$2^{(4 \lg n)}$	$2^{\lg n}$	$n^{\lg 32}$
(d)	$\max(2000n^2, 0.002n^3)$	$\min(2000n^2, 0.002n^3)$	$2000n^2 + 0.002n^3$
(e)	$\lceil n^2/11 \rceil$	$\lfloor n^2/11 \rfloor$	$n^2/11$

3. [15 pts] Prime Time

Consider the following IsPrime algorithm for determining if a given input, n where $n > 1$, is a prime.

```
IsPrime($n$):
  for $i = 2 \to \sqrt{n}$:
      if $n \bmod i = 0$ then
          return NO
  return YES
```

(a) Prove that the algorithm correctly determines if the input integer $n > 1$ is a prime or not.

(b) Give a Θ-bound of the worst-case running time of the algorithm in terms of the input integer n.

(c) Let b denote the *size* of the input to IsPrime, i.e., b is the number of bits needed to represent n. Give a Θ-bound of the worst-case running time of the algorithm as a function of b.

(d) Is IsPrime a polynomial-time algorithm or exponential-time algorithm in the size of the input b?

4. [18 pts] Ode to Code

For each of the code fragments given below, give a bound of the form $\Theta(f(n))$ on its running time on an input of size n. Justify your answer.

(a)

```
for ($i = 1$; $i \le \frac{n}{2}$; $i = i * 2$) do
    for ($j = i$; $j \le 4 * i$; $j = j * 2$) do
        print ("ihdnag vijar")
```

(b)

```
for ($i = 2$; $i \le n$; $i = i^2$) do
    for ($j = \frac{n}{1024}$; $j \le n^2$; $j = j * 2$) do
        print ("hi")
```

(c)

```
for ($i = 1$; $i < n$; $i = i * 3$) do
    for ($j = n$; $j > i$; $j = j - 1$) do
        print ("beep boop boop bop");
```

5. [21 pts] Expansions

Solve the following recurrences using the method of expansion (iteration), giving your answer in Θ notation. In answering these questions make sure to include a generalized expression for $T(n)$ (i.e., $T(n)$ expressed in terms of k), the value of k for which the recursion bottoms out, and then any algebra to simplify your answer. In other words, show your work.

(a) $T(n) = 3T(n/3) + 2n$. Assume that $T(n) = 1$, for all $n \le 3$.

(b) $T(n) = 6T(n-3)$. Assume that $T(n) = 1$, for all $n \le 3$.

(c) $T(n) = \sqrt{n}T(\sqrt{n}) + n$. Assume $T(n) = 1$, for all $n \le 2$.

6. [15 pts] What Not to Wear

Robin is a new TA this semester. To celebrate him, the 121 staff is throwing him a party. JJ is nice enough to let Robin borrow clothes from his closet. In JJ's closet, there are n T-shirts to choose from (you may assume $n > 2$), some of which are stylish and the rest of which are ugly. Additionally, he learns that either the stylish T-shirts or ugly T-shirts make up at least 85% of JJ's closet.

Robin wants to impress his fellow TAs so he needs to determine whether JJ's closet contains mostly stylish or mostly ugly T-shirts.

(a) Give a deterministic algorithm such that Robin only removes the fewest number of T-shirts from JJ's closet such that he can definitively determine whether or not it contains mostly stylish or ugly T-shirts. Note that Robin cannot see the T-shirts in the closet without removing them from the closet. Provide a proof of correctness and runtime analysis for your algorithm. Use Θ notation to express your answer.

(b) Consider the following randomized algorithm for the problem, in which the function randomShirt chooses a random T-shirt from JJ's closet, where each T-shirt is equally likely to be chosen. The T-shirt is then returned to JJ's closet. The function `isStylish` returns 1 if the T-shirt is stylish and 0 if the T-shirt is ugly. State the runtime of the algorithm. You may assume that the functions `randomShirt` and `isStylish` both take $O(1)$ time.

```
$i \leftarrow$ randomShirt
$j \leftarrow$ randomShirt
$k \leftarrow$ randomShirt
if isStylish($i$) + isStylish($j$) + isStylish($k$) $\leq 1$ then
  return "mostly ugly T-shirts"
else
  return "mostly stylish T-shirts"
```

(c) The algorithm can output an incorrect answer. Explain how this can happen.

(d) Suppose *exactly* 85% of T-shirts in JJ's closet are ugly. Find the probability that the randomized algorithm will output "`mostly stylish T-shirts`." Justify your answer.

7. [16 pts] Arrays for Days

Consider the following problem. You're given an array A consisting of n integers $A[1], A[2], \ldots, A[n]$. You'd like to output a two-dimensional n-by-n array B in which $B[i, j]$ (for $i < j$) contains the sum of array entries $A[i]$ through $A[j]$ – that is, the sum $A[i] + A[i+1] + \cdots + A[j]$. (The value of the array entry $B[i, j]$ is left unspecified whenever $i \geq j$, so it doesn't matter what is the output of these values. You should ignore these values when summing the array entries.)

Here is a simple algorithm to solve this problem.

```
for ($i = 1; i <= n; i++$)
  for ($j = i + 1; j <= n; j++$)
      Add up array entries $A[i]$ through $A[j]$
      Store the result in $B[i, j]$
```

(a) For the above code fragment find the Θ bound of the running time in terms of the input size of n by giving both a Big-Oh and Big-Omega bound. That is, show that the runtime is $\Theta(f(n))$ by showing it is both $O(f(n))$ and $\Omega(f(n))$ for some function $f(n)$. Justify your answer.

(b) Although the algorithm in part (a) is the most natural way to solve the problem – after all, it just iterates through the relevant entries of the array B, filling in a value for each – it contains some highly unnecessary sources of inefficiency. Give a different algorithm to solve this problem, with an asymptotically better running time. In other words, you should design an algorithm with running time $O(g(n))$, where $\lim_{n\to\infty} g(n)/f(n) = 0$. Prove why your algorithm works and also analyze the run time.

[0 pts] Feedback: No feedback form for this assignment. Feel free to reach out to the Head TAs if you have any comments/concerns.