CIS 121 — Data Structures and Algorithms
Homework Assignment 4

Assigned: February 10, 2020 **Due:** February 17, 2020

Note: The homework is due **electronically on Gradescope** on February 17, 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.

1. [10 pts] 2020: A Love Odyssey

With February 14th coming up, Marshall has been flooded with requests from people who want to be his Valentine. Unfortunately, due to his busy schedule, Marshall can't keep track of the k requests he's received. With time running out, Marshall needs to find out the value of k so he knows how many people to get back to. He decides to ask Liz for help, and so she creates a robot with artificial semi-intelligence named VAL 9000 to help him process his requests.

Each time Marshall guesses an integer, VAL will tell him whether his guess is too high, too low, or exactly correct. Unfortunately, there's a bug in Liz's code, and VAL will only allow Marshall to make $O(\lg k)$ guesses. Note that Marshall may not guess in terms of k and only knows that it is a positive integer.

Give an algorithm that will allow Marshall to correctly determine the number of Valentine's he's received using $O(\lg k)$ guesses. Be sure to justify the correctness and runtime of your algorithm.

2. [20 pts] Roses are red, violets are blue, Rakuten is for you

Seeing the new promotions on Rakuten, Steven decides to buy n (you can suppose n is a power of two) roses. However, when he receives his bouqet, he finds out they accidentally sent him a multi-colored bouquet of roses (you may not assume any type of ordering amongst the roses). He observes that there are k different types of roses, each distinguished by a unique color.

He thinks only a unicolor bouquet is pretty and worthy of giving to someone. So, he decides that if he can make a pretty bouquet of strictly greater than $n/2$ roses of the same color, he will select those roses and make the bouquet to give to a special someone for Valentine's Day. Otherwise, he will give up and throw out all the roses.

Steven wants to figure out if he can make a pretty bouquet. However, still recovering from his last failed Valentine's Day, Steven's mind is a little unfocused. Therefore, all Steven is able to do is pick up two roses and determine in constant time whether or not those roses are the same color.

(a) Give an $O(n \lg n)$ divide-and-conquer algorithm to determine if Steven will be able to make a pretty bouquet in time to impress his special friend.

(b) Design a linear-time algorithm to solve the same problem. (Hint: consider pairing up the roses. How could you discard some of these roses such that the majority color rose is maintained?)

Make sure to include a proof of correctness and runtime analysis for your algorithms in both parts.

3. [25 pts] Smart Cookies

Yunhee is making Valentine's day cookies for all n CIS 121 TAs. Each TA has a **distinct** sweetness preference for their cookies and will not eat any cookie that is sweeter or more bland than that exact sweetness. Yunhee made these cookies so that there is exactly one cookie for each TA, made to their exact preferred sweetness. However, when making the cookies, Yunhee forgot to write down which cookie belongs to which TA. She also doesn't have any way to quantify the sweetness of each cookie, and none of the TAs know how to verbalize their preferences and/or organize amongst themselves. **The only test that Yunhee has is to have some TA try a particular cookie, and tell her whether the cookie is too bland, too sweet, or just right for them.** You can assume that each test takes a constant amount of time and there are an equal number of TAs and cookies. In this problem, we want to design an efficient algorithm that Yunhee can use to give each cookie to the correct TA.

(a) Consider the following algorithm to figure out which cookie is for which TA. Let T denote the group of TAs and C be the set of cookies. What is the *expected* running time of this algorithm? What is its *worst case* running time? Justify your answer.

```
COOKIE($T, C$):
if $|T| = |C| = 1$ then:
    Give the cookie to the TA
else:
    Pick a cookie $c$ uniformly at random from $C$
    $T' = T$
    $C' = C$
    for each TA $t$ in $T$ do:
        TA $t$ tries cookie $c$
        if it is just right then:
            Give cookie $c$ to TA $t$
            $T' = T' - \{t\}$
            $C' = C' - \{c\}$
            break out of the for loop
    COOKIE($T', C'$)
```

(b) Design an algorithm with a better bound than the one in part (a). Be sure to explain your algorithm well, analyze its expected running time, and analyze its worst case running time.

If you'd like (for this part only!) you may provide properly-formatted pseudocode to **support** your English explanation of your algorithm. That is, you still **must** provide an English explanation. You can easily format pseudocode using the `verbatim` environment.

(c) Suppose the following: each time Yunhee gives a cookie to a TA, the TA either says that the cookie is just right or it's not the correct sweetness, without giving any information on whether it is too sweet or too bland. Does the algorithm in part (a) still work? What about the one in part (b)? Give a brief (1-3 sentence) justification for your answer.

4. [15 pts] Michael's Median

Michael is looking for a restaurant to take that special someone to for Valentine's Day. Michael wants to go somewhere nice, but not *too* nice, to keep the dinner affordable. On Yelp, he finds a list, R, of n restaurants in Philadelphia, all with different price points. Then, to keep the price reasonable, he decides to make a list of the x restaurants which are closest to the median price. Note that $x < n$. Moreover, you may assume that n is odd, and any ties may be broken arbitrarily.

For example, if R contains five restaurants r_1, r_2, r_3, r_4, r_5, with prices given by $(10, 12, 5, 90, 3)$, and $x = 2$, then your algorithm should return r_1 and r_2.

Additionally, Michael is in a bit of a time crunch, as Valentine's Day is quickly approaching and he needs to secure a reservation. He doesn't have time to sort the list of restaurants and he thinks that it can be done faster. Describe a linear time algorithm (with respect to n) that he could use. Don't forget to analyze its runtime. Proof of correctness is not necessary.

5. [10 pts] Robin's Card Conundrum

With Valentine's Day approaching, Robin weeps at the fact that no one has written him a Valentine's Day card. Wiping away his tears, Robin suddenly gets an idea. He buys himself a total of n Valentine's Day cards at CVS, with $n \geq 2$. The cards are labeled $1, 2, ..., n$ and are stored in a special heart-stamped envelope E that acts as a queue. However, wishing to hide the fact from his friends that he bought the cards for himself, he decides to swap the order of some of his cards. Specifically, he tries to reverse the order of only the even-indexed cards. For instance, he swaps the second card with the card in the last even position, he swaps the fourth card with the card in the second-to-last even position, etc.

Having spent all of his money on Valentine's Day cards, he can only afford one stack S and one queue Q (separate from the special envelope), both of which are initially empty. Help Robin hide his loneliness from his friends by finding an algorithm that changes the order of the cards in E in $O(n)$ time and $O(1)$ extra space outside of the provided stack and queue. Be sure to justify the runtime and space usage of your algorithm. No proof of correctness is required.

6. [20 pts] Life is Like a Box of Chocolates

The 121 staff is standing in line at the grocery store checkout line in order to buy chocolates for Valentine's day. However, there are problems with the cash register and all n TAs are stuck in a single line. Since they are getting impatient, they want to know exactly how far they are from the front of the line. However, since some TAs are taller than others, the taller TAs may block the vision of shorter TAs. So, the each TA decides to get upset at the first taller TA in front of them. If there is no such taller TA in front of them, they decide to arbitrarily be jealous of the first TA in line. Note that the heights of the TAs are not necessarily distinct.

Naive $\Theta(n^2)$ Algorithm: For each TA t in line, keep checking the height of the TAs in front of t until we reach a TA taller than t or we reach the first TA in line. Denote this taller TA as u and conclude that TA t will be jealous of TA u.

As all the TAs are in a hurry to get on with their Valentine's plans, they need a more efficient algorithm. Help them come up with a more efficient algorithm to decide who to get upset at in $\Theta(n)$ time. Specifically, return an array A of size n where $A[i]$ contains the index of the TA that TA i decides to be upset with. Prove that your algorithm is correct and derive its running time.

[5 pts] Feedback: How long did you spend on this assignment? What did you think of it, and how can we improve written assignments in the future? As an incentive to complete the 2-3 minute form, 5 points are allocated to its completion. Let us know via the anonymous Canvas survey. The form will be due 3 days after the original homework deadline. We really appreciate your feedback!