Homework 3

ECE 358 Foundations of Computing
Fall Semester, 2023

Due: Friday, October 27, 5:00pm

- All page numbers are from the 2022 edition of Cormen, Leiserson, Rivest and Stein.
- For each algorithm you asked to design you should give a detailed *description* of the idea, proof of algorithm correctness, termination, analysis of time and space complexity. If not, your answer will be incomplete and you will miss credit. You are allowed to refer to pages in the textbook.
- Do not write C code! When asked to describe an algorithm give analytical pseudocode.
- **Read the handout with the instructions on how to submit your Homework online. Failure to adhere to those instructions may disqualify you and you may receive a mark of zero on your HW.**
- Write *clearly*, if we cannot understand what you write you may not get credit for the question. Be as formal as possible in your answers. Don't forget to include your name(s) and student number(s) on the front page!
- No Junk Clause: For any question, if you don't know the answer, you may write "I DON'T KNOW" in order to receive 20% of the marks.

---

1. [**Data structure comparison, 15 points**]

   Suppose that you have to write your own dictionary implementation, which has to support $insert(k)$, $search(k)$, $delete(k)$ and $rank(k)$ operations. $rank(k)$ finds how many keys in the dictionary are smaller than k, and $insert(k)$, $search(k)$, $delete(k)$ stand for insertion of key in dictionary, searching and deletion respectively.

   Which one of the following data structure would you use: linked list, array, hashmap, or a self-balancing binary search tree? Provide a brief argument either for or against for each one of them. Assume that no operation is called less frequently than the others.

2. [**Hashing, 10 points**]

   Demonstrate the insertion of keys 7, 9, 88, 11, 25, 23, 22, 28, 14, and 21 into a doubly-hashed table where collisions are resolved with open addressing. Let the table have 11 slots, let the *primary* hash function be $h_p(k) = k \mod 11$ and let the *secondary* hash function be $h_s(k) = 3k \mod 4$.

3. **[Greedy Algorithms, 5+5+5+5 points]**

   At a restaurant, the clients reduce the tip they would give to the waiter according to the time it took for him to reach their table and take the order. Further, each client has their own impatience. Therefore, if client i waited for w minutes and has impatience t_i, they will deduct $t_i \cdot w$ from the tip. Help the waiter to decide the order in which to take orders from n clients, so as to minimize his loses, if each client takes o_i time to give their order and the waiter goes directly from client to client.

   (a) Find an expression for the loses, propose a greedy algorithm that will minimize them, and analyze its runtime.

   (b) After making your greedy choice, show that the problem reduces to a smaller subproblem.

   (c) Prove the greedy choice property by showing that there is an optimal solution that agrees with the first greedy choice your algorithm makes.

   (d) Prove the problem's optimal substructure by showing that the optimal solution to the subproblem combined with your greedy choice leads to an optimal solution.

4. **[Dynamic Programming, 10+10 points]**

   Xun and Yuntao are trying to complete their group projects at UofT in the following N days.

   Being very polite, both Xun and Yuntao don't want the other person to complete the project for them, meaning that each of them wants to handle as many deadlines as possible by themselves. However, they need to rest, too, so they've decided to take turns when it comes to working on projects, meaning that every day, either Xun or Yuntao is working. On each day $i = 1 \dots N$, there are a number of deadlines $d[i]$ that need to be completed. After flipping a cryptocoin, they decided that Xun starts working on deadlines first, and Yuntao can rest in the meantime.

   To balance the load, they agreed to the following schedule: the person that is working should work on all deadlines that are due in the next X days, where $1 \le X \le 3D$ (i.e. everyone is free to choose any X that lies within these bounds, and they will have to work X days on all the deadlines that need to be handled on these days). In the beginning of the deadline rush, $D = 1$. After any person finishes their shift, they set $D = \max(X, D)$ for the next person.

   Example:

   d = [1, 3, 3, 2, 5]

   If Xun chooses $X = 1$, she will have to complete 1 deadline that is due today, and only then Yuntao chooses the next X value from 1 to $3D = 3$. If Xun chooses $X = 3$, she would have to complete $1 + 3 + 3$ deadlines, and only then Yuntao chooses the next X value from 1 to $3D = 9$.

   Xun and Yuntao always complete all deadlines in time and always choose X optimally. Find the person who will handle more deadlines.

   (a) Define the recurrence relationship for the number of deadlines that Xun handles. Answers without a proper explanation will be given partial or no credit.

   (*Hint*: The way that Xun and Yuntao handle the deadlines is a zero-sum game. An important realization is that Xun tries to maximize her score while Yuntao tries to minimize it (try to convince yourself that this is true by doing a pen-and-paper example on small inputs). For Yuntao, the argument is symmetric. This principle is called minimax, or maximin. Using this hint, try to create a function that describes the way each of the players implement their strategies.)

(b) Describe the implementation of your solution clearly (pseudocode is not mandatory). Try to make it as efficient as you can; marks will be deduced for inefficient solutions. Analyze its running time and space complexity with and without dynamic programming optimization (correctness and termination are not required).