Name:________________________ Tufts ID: ____________

COMP160: ALGORITHMS, SPRING 2020, **Exam 2**

Instructions

- You have 3 hours to answer the 4 questions in the exam.
- The exam is designed to take only 75 minutes so that you will have plenty of time for uploading at the end, plus extra time to work on the problems.
- Please read **each** question carefully, specify any assumptions that you make, and **briefly justify** all your answers unless otherwise indicated.
- Please use a different page to answer each question. Your answers may be handwritten or typewritten or any combination as long as they are easily readable.
- When you are done, upload your answers to Gradescope **before** the end of the 3 hour time period. Late submissions will be penalized 1 pt per minute late.
- This exam is open-book. You may consult any books, notes, webpages, online videos, etc that do not involve interacting with a live person. You may **not** consult a live person in any way.
- If you need any clarifications during the exam make a **private** question on Piazza
-
-
-

Please write the following affirmation and sign with your name. This will be uploaded as problem 5 on Gradescope and is necessary in order for you to receive credit on the exam:

"I affirm that I have not consulted with any other person (except the instructors) in the course of answering this exam. I understand that doing so may result in a grade of zero on the exam. Signed, "

1. (25 points) We want to have a hash table to store grades. The key of the table will be the student's id, and we will store their grade. All of the questions below assume we have n elements in a table with m buckets and that collisions are handled with chaining:

   (a) (3 points) Look through your notes. What is the expected runtime of a successful search in a hash table? What about an unsuccessful search? Just list runtimes as a function of n and m (no justification necessary).

   (b) (5 points) Say we want expected runtime for an unsuccessful search to be $\Theta(\log n)$. Give a value of m that we could choose (as a function of n). What if we wanted expected runtime to be $\Theta(1)$? No justification is necessary.

   (c) (5 points) In the classic hash table with chaining, elements which collide (hash to the same bucket) are stored in a linked list. Suppose we still have m buckets, but store elements which hash to the same bucket in an AVL tree instead (each bucket will have its own AVL tree, indexed by tufts id). Draw an example with 5 buckets, 3 of which are empty, 1 only has a single item, and 1 has 7 items

   (d) (7 points) Give the pseudocode of the Search(id) function for the modified hash table (where we use AVL trees instead of linked lists). This function should return the grade of the student whose Tufts id is `id` (or -1 if such a student does not exist). For simplicity, assume that the hash function of a student whose id is `id` can be are computed by function $h(\texttt{id})$.

   (e) (5 points) Since we modified the hash structure, the runtime of an unsuccessful search probably changed. What is the new expected runtime? Give your answer in terms of n and m (for a general table, do not use the values of your answer to question (b)). Justify your answer with 1-2 sentences.

2. (20 points, 4 points each) Suppose that we insert **some** integers in the range 1 to 1000 stored in a binary search tree in an unknown order (the only thing we know is that no number is repeated). We then execute a search for number 267 that is in the tree. We are interested in the sequence of nodes that are traversed before reaching that number.

   For each of the sequences below, decide whether it **could** or **could not** be a proper sequence. Justify your answer with 1 sentence (a drawing/description of the tree in which it happens is good enough).

   Example (no need to answer this): 267

   Answer: Yes, this could happen if the root of the BST is number 267

   Example 2 (no need to answer this): $100, 345, 100, 267$

   Answer: No. This could not happen because once we see a node we do not revisit it.

   (a) $911, 369, 269, 268, 267$

   (b) $911, 169, 860, 238, 269, 267$

   (c) $264, 265, 266, 267$

   (d) $925, 806, 240, 808, 267$

   (e) Would any of your answers change if we are searching in an AVL tree instead? If so, which one(s)? Why?

3. (25 points) You have a bunch of kids forming a line in front of a teacher before their respective parents pick them up. Each kid is represented by the tuple `(id, height, pos)` where `id` is its unique identifier (say, a name), `height` is their height and `pos` listing its location (an strictly positive integer representing the distance to the teacher, in meters).

   Design a data structure that can perform the following operations:

   - ADD(id, height, pos) Record that a new kid has been added to the line
   - PICKUP(id) Record that a kid has been picked up by its parents (thus, it should be removed from the data structure)
   - ISVISIBLE(id) returns true if the teacher can see the kid. Since all kids are on a line, a kid is visible if and only if there is no other kid that is taller and is closer to the teacher.

   Describe your data structure, invariants and an algorithm for performing each operation. Discuss correctness of the algorithm and justify runtime of each operation with 1-2 sentences (for full credit all operations must be executed in $O(\log n)$ worst case time).

4. (25 points)

   CONGRATULATIONS! After a long fight you defeated the dragon from homework 7! Now it is time to escape from the collapsing dragon's lair while collecting as much treasure as you can. The dragon's lair has a rectangular grid-like structure: As soon as you enter a cell you automatically pick up any coins in that cell. You look at the scriptures for advice and you read:

   When exiting the dragon's lair don't be greedy. Go to the escape by only walking down or to the right of your current position.

   The scriptures magically contain an array A with the amount of gold in each position (that is $A[i, j]$ is equal to the amount of gold in position (i, j) of the dragon's lair). Also, you can assume that you start in position $(1, 1)$ and must reach the exit in position (n, m). Moving right means you increase the x coordinate and by moving down you increase the y coordinate. Your goal is to design an algorithm that returns the maximum number of coins that you can collect while preserving the above constraints. Answer the following questions:

   (a) What technique will you use to solve the problem? Just say the name and explain why you think it is better than other possible options (2-3 sentences is enough)

   (b) Describe in detail your solution and justify correctness

   (c) What is the runtime of your algorithm?

   Note: as usual, points will be given for clear structure, efficiency, and arguing correctness.

5. (5 points) Write the following affirmation and sign with your name. This is necessary in order for you to receive credit on the exam:

   "I affirm that I have not consulted with any other person (except the instructors) in the course of answering this exam. I understand that doing so may result in a grade of zero on the exam. Signed, "