COMP 285 (NC A&T, Fall ‘22) Homework 1 (100 pt)

Fun with Algorithms

Due 09/06/22 @ 1:59PM ET

Submitting

In order to submit this assignment, you must download your assignment as a ZIP file and upload it to Gradescope (setup instructions below). There will be a written component and a coding component.

For the written component, you will write your answers in the corresponding `.txt` files.

For the coding component, you will write your solution in `answers.cpp`. We will use an autograder to test your solutions - the results of which can be seen after submitting to Gradescope. Note that you have unlimited tries to submit. In order to receive full credit, you must also provide documentation on your approach, see the commented sections in the `answers.cpp` file.

Question 1: Interview Practice: Who's Missing! (15 pt)

Makayla is just wrapping up her software engineering internship at Foogle. She recently developed an algorithm, generateIDs, that gives unique IDs, starting from 0 and in increasing order, to each Foogle account in a particular region. This is how she defined her algorithm, in a `.h` file. However, she’s running into a few issues with her code. It seems that the returned vector is always too small - precisely, it seems to always be *exactly* one element too small. For example, Makayla knows that in the country of Algo-landia, there are exactly 2^{20} = 1,048,576 accounts. However, when she runs the following lines of code: It prints 1,048,575, which is precisely one short. Confused, Makayla decides to ask Tiffani, a fellow intern, for some help.

1. (5 pt) Is it sorted?

Tiffani suggests to Makayla that they first write a program to check if the IDs they're getting back are **sorted in descending order**. She thinks that if they’re sorted, it’ll be easy for them to find which one is missing!

She starts writing out the program to check if a vector of integers is sorted in descending order, but accidently forgot to save when she was done coding. Below is what she was able to recover. Flabergasted, Tiffani turns to you for help!

a. (1 pt) What should go in the first set of ‘???’ in line 5 above? *Be sure to read the comment for the function carefully!*

b. (2 pt) What should go in the second set of ‘???’ in line 7 above? *Be sure to consider edge cases!*

c. (2 pt) What is the big-Oh running time of the isSortedDescending function, if we let `n` be the number of elements?

Write your answers in `q1.txt`.

2. (5 pt) So what account is missing?

Sadly, it turns out the lists are not sorted! I guess the documentation was right this time! You, Makayla, and Tiffani start brainstorming other solutions. How can we figure out which ID is the one that’s missing?

Makayla gives another suggestion. On the white board, She writes the numbers:

0, 1, 2, 3, 4, 5, 6, 7

Then asks:

a. (1 pt) What do the numbers above add up to?

She then erases the numbers, and writes:

(0, 7); (1, 6); (2, 5); (3, 4)

She turns to you, and asks:

b. (2 pt) What do each of the pairs add up to? How many pairs are there?

After thinking about the above for a bit, all three of you jump up excitedly! Tiffani goes to the board and writes:

0, 1, 2, 3, 4, . . . , n − 3, n − 2, n − 1, n

and then writes:

(0, n); (1, n − 1); (2, n − 2); (3, n − 3); (4, n − 4), . . .

Both Makayla and Tiffani are quite smart! They turn to you, and ask:

c. (2 pt) If we add up all the numbers from 0 to n, what do they add up to? Is there a closed-form formula? *As a hint, check out this video.*

Write your answers in `q1.txt`.

3. (5 pt) Find the missing ID

Having figured out the above, the three of you start coding! You come up with the following pseudo-code: Now, Makayla and Tiffani look at you, and say: "Can you code it up for us?"

Write your solution in `answers.cpp`.

Question 2: Interview Practice: The Best Basketball Player (15 pt)

Stanton has been following the Denver Nuggets very closely this season.

He’s a bit disappointed with their performance this season, so he reaches out to his friend Austin.

Austin is a data scientist working for the U.S. Census. He tells Stanton that a big issue with the Denver Nuggets this season is that their players are too short! What they really need is a tall player - wait, not a tall player, *the tallest player in America*.

Thankfully, Austin has just the data for the job! He shows Stanton that the census collects every person’s height as an integer in inches. They always round to the nearest inch and the dataset is in a random order.

1. (10 pt) Finding the tallest American

Stanton and Austin feel like they’ve done a good job so far, so they reach out to you. They know you’re an expert in C++, do they ask you to write a function - `findTallestPerson` - which takes in a vector of integers correspondong to each person’s height from the census data Austin has, and returns the index of the largest height. If there are multiple largest heights, it returns the smaller index.

They give you the following examples: You look at them a bit funny, and wonder to yourself: "Who in the world has negative height? And why would the list ever be empty?" Still, you want to do a good job, so you do what they ask.

Write your solution in `answers.cpp`.

2. (5 pt) How fast is it?

You also want to show everyone how good you are at analyzing the runtime of an algorithm. - **a. (3 pt)** What is the big-Oh runtime of the pseudo-code shown above?

Austin realizes he has another dataset that might come in handy. After digging around in the census databases, he excitedly shows you the new dataset. This particular dataset contains every person's height sorted in *ascending order*! Eager to show Stanton and Austin your runtime knowledge, you ask yourself:

- **b. (2 pt)** What is the fastest big-Oh runtime of a `findTallestPerson` algorithm we could write if we used Austin's new dataset instead? (You don't need to implement it, just give the optimal runtime)

Write your answers in `q2.txt`.

Question 3: Interview Practice: Game Development (15 pt)

Hassan is a world-renowed game developer working for Big Corp Studios. His game works by randomly generating pets, each which costs a random amount of coins (only integer values). The player is then allowed to purchase precisely two pets.

Being an awesome developer, Hassan wants to make sure that no matter how many coins the player has, there are always two pets that we can purchase at the cost of all his coins.

1. (13 pt) Finding the most expensive pets

Having been stumped with this problem for quite a bit, he reaches out to Teco, whose an avid gamer and who recently took COMP 285 for help. He asks Teco to write a C++ function `findTwoPets`, which takes as input a vector of integers (corresponding to the cost of each pet) as well as a target value (corresponding to how many coins the player currently has), and returns a pair that represents two **distinct** indices of pet costs that sum up to the target value. *We need distinct indices since we a player cannot buy the same pet twice!*

For some reason, Hassan also wants the smaller index to be first. To be super clear, he gives a few examples to Teco: In the first example, `vec[1] + vec[3] = 2 + 4 = 6` . In the second example, we returned `-1, -1` because there are not two distinct elements within the vector that sum to 10 (you can't use 5 twice).

For context, there's a straight-forward O(n^2) algorithm that you can start with; this might be the first one you think of. There is also an O(n log n) algorithm (Hint: what if the list was sorted?), and even a far more efficient O(n) algorithm.

For this homework, there are no explicit time complexity constraints here (i.e. algorithm just needs to work as intended). You can receive bonus points if you implement one of the faster ones, though.

Help Teco by writing an algorithm (in C++) that solves this problem.

Write your solution in `answers.cpp`.

2. (2 pt) Runtime of the algorithm

- **a. (2 pt)** For whichiever algorithm you implemented, what is the big-Oh runtime?

Write your answers in `q3.txt`.

Question 4: Exercise: Refresher on Nodes, Pointers, etc. (15 pt)

1. (13 pt) Adding two numbers

Almost every programming languages already has `+` and `*` implemented for you, on integers. So to make this more interesting, your task is to write the function add, which given two numbers represented as linked lists of single-digit integers, returns their sum as a linked list of integers. Return the head node.

The linked list is represented as follows: We have provided a util function `makeList` to convert an integer to its corresponding list representation. Feel free to take a look at the function in `util.cpp` from the starter code and note how we extract the digits (`num % 10` and `num / 10` in the loop).

For convenience, the list stores the digits from the lowest to the highest, e.g., `makeList(123)` => `head -> [3 -> 2 -> 1]` .

Some examples for add:

Refresher on pointers, memory allocation, and deallocation for your reference: - To create a Node object, use `Node<int>* ptr = new Node<int>()`. The keyword `new` dynamically allocates memory for the node. - We provided a `freeList` function that will traverse the linked list and free every node. `freeList` is all we need here, but always remember when you have allocated memory (i.e. with `new`) to free it when finished (i.e. `delete ptr`). - To check whether there are "memory leaks" (i.e., allocated memory that's not deallocated by the time the program finishes running), you can run valgrind on repl.it with `valgrind –leak-check-full ./main`

Please note that this question is at the upper-end of the C++ intricacies we will learn on in the course. Having familiarity with pointers here will serve us well later on (e.g. with trees).

Write your solution in `answers.cpp`.

2. (2 pt) Runtime of addition

The algorithm above forces you to implement addition the way you would normally do it by hand (i.e., add numbers together then carry when they add to more than 10). In this context, let us define "operation" as being a 1-digit addition.

- **a. (2 pt)** What is the big-Oh running time of this algorithm, in terms of n where n is the length of the input lists (i.e., numbers)?

Write your answer in `q4.txt`.

Question 5: Resumes, Career Prep Module #1 (40 pt)

I hope most of you are preparing yourselves for an amazing career - be it going to graduate school, getting a job in tech, or doing whatever makes the most sense for you (any entrepreneurs?).

As we learned in lecture, everyone can write a polished and compelling resume - so let's jump right in!

The end deliverable for this question will be a single, 2-page document (Microsoft Word doc or Google Docs only).

a. (5 pt) Please make the first page of this document a copy of your current resume (which you submitted as part of HW0).

b.(35 pt) Let's start writing/editing!

- Ensure your resume follows a template similar to this one (has my more in-depth comments), this two, this three, or this four. It does not have to be any of these exact templates, but we're looking for something that is stylistically similar. This means clear headings for every section, bolding/italics to differentiate roles or names, and indentation schemes that make the document skimmable. If you are unsure if your current formatting aligns, feel free to email a copy to me.
- Now we get to the meat of the resume... remember our formula from lecture. For bullets in each section of your resume (Experience, Course or Independent Projects, Volunteering, etc.), apply the following formula to the best extent possible.

Think deeply about what you did, built, completed, worked on, or achieved in each circumstance and why it was significant or impactful. Be sure to quantify when possible, but understand this is more of an art than a science.

Did you discover a problem and take initiative to solve it? Did you do something that you're proud of at a part-time job, internship or in a student club? Did a manager give you positive feedback on something you did or built?

Every experience you've had during college counts; every experience can be valuable when positioned correctly.

Polishing your resume and structuring bullets to maximize impact can be *very time intesive*. However, consider it a necessary upfront investment in your future career that will pay off ten-fold. After completing this part of the assignment, you're even better positioned for the next step in your career!

As always, my door is open to provide support. We can brainstorm or workshop content should you want the extra pair of eyes!

Please upload the single document with your original and revised resume here; we will follow this checklist when evaluating.