Assignment 3
Deadline: 11:59PM Dec 6

Partner Requirement
You can work in a group of up to 3. Your partner can be from a different set.

Requirements
In this assignment, you are required to write a function that reads string inputs from a file. The name contains a first and last name. Your program adds, deletes, and searches a node from a binary search tree. It should be able to perform in-order traversal of the tree and print it to the output. The pre-order is done in a way that the output is ascending order by the first name, then the last name.

For add operation, there is no output (a node is added since we allow a duplicate. More to come later). For search operation, if a name is found "Found" is the output. If not found, "Not Found" is written to the output. One design requirement is that there can be duplicates, so make sure to handle this.

The delete operation is different from conventional delete operations. If a data is found in the tree, then the node is deleted. If the data is not found, then the node is added. For example, a user is trying to delete "Steve Roger" and it is not currently in tree, then "Steve Roger" node will be added.

Operations are specified after the name by an integer:
1: add operation
2: delete operation
3: search operation
4: perform the traversal

Option 4 does not need any input, so it will just traverses the tree and print the names in the correct order.

Below is a sample run of the program.

```
>> cat input.txt
John Kim 1
John Kim 3
4
Chris Patel 1
John Kim 2
Chris Patel 1
Chris Patel 3
4
Chris Patel 2
Chris Patel 3
Chris Batel 1
4
Lisa Hampton 3
>> ./<executable> input.txt output.txt
>> cat output.txt
```

```
Found
John Kim
Found
Chris Patel
Chris Patel
Found
Chris Batel
Chris Patel
Not Found
```

Below are the specifications of the lab.

1. The run command will be
   `>> ./<executable> input.txt output.txt`
2. There can be duplicates.
3. Values in the input file are not sorted.
4. There will be corner case testing in this assignment.
5. Guaranteed to have no empty lines, no integer outside integer range and no data other than int type in the input file
6. First and last names will be guaranteed to be present. Both names are case-sensitive and must be used in the ordering using ASCII value. For example, Roger comes before roger since R is 82 and r is 114.
7. In any case an error is detected, print any message that contains "Error" and exit the program.
8. Do not add any random trailing new lines or spaces. I will allow up to one new line character at the end of an output file. Example below:

   ```
   >> cat output.txt
   Found
   Found <- having a new line character here is please
   >>
   ```

   Below examples will be incorrect

   ```
   >> cat output.txt
   Found
   Found >>    <- no new line character
   ```

   Or

   ```
   >> cat output.txt
   Found
   Found
   Found
       <- extra new line
   >>
   ```

Grading

I will provide sample input and output files. Make sure to test against those to ensure that your program output format works. Any grading failure due to not following specifications will result in 0. For full marks this week, you must:

- (6 point) Generate a correct solution to the problem(s) in this lab

- (2 point) handle error cases

Extra Bonus
Your job is to make the above operations as fast as possible. I will allow you to change the data structures or operations as long as it lets do the operation as specified. I will not count space complexity into account. I will just run the program and get the time. If you are interested, you can run the following to see how long it takes to run the program.

```
>> time ./<name of executable> <input file> <output file>
```

It will report the time it takes to run the program. I will rank the speed from the shortest runtime to the longest runtime. Top 10 students will get the bonus. If the output is incorrect, it will not be considered in the competition. The following is the bonus that will be applied:

- Midterm grade 50% and below: change the midterm grade to 51%
- Midterm grade between 50.1% and 60: add 10% to the midterm grade
- Midterm grade between 60.1% and 70: add 8% to the midterm grade
- Midterm grade between 70.1% and 80: add 6% to the midterm grade
- Midterm grade between 80.1% and 100: add 5% to the midterm grade

Unfortunately, the remaining students will not get any bonus point. I will announce the winners by listing all A numbers of the class from the fastest to the lowest. This is an optional part of the assignment, and it will not impact any of your existing grades.

By entering the competition, you agree that your code can be released to the class since top performer's code will be released to the class. If you do not want to enter the competition, please email me.

Submission Files

- You must deliver only one file named: **assignment3.c** (do not capitalize)
- Submit your file to the learning hub before the deadline.
- Make sure to update your A numbers by inserting the following in the main function:

```
// A numbers of everyone. AXXXX_AXXXX_AXXX format.
char *ANum = "A0000";

FILE *outputFile = fopen(ANum, "w");

if (outputFile == NULL) {
     printf("Failed to create the output file.\n");
     return 1;
}

fclose(outputFile);
```