NYU, Tandon School of Engineering
Bridge to Computer Science Program — Winter 2021

Third Midterm Exam

*****PLEASE REMEMBER TO SUBMIT ON CLASSES, GRADESCOPE and EMAIL*******

- There are 100 points total.
- Note that there are longer programming problems at the end. Be sure to allow enough time for these.
- We supplied you with a file, named 'solutions.txt', where you should type all your answers in.
- For editing this file, you are allowed to use compilers such as Visual Studio, XCODE, CLion, textedit or notepad
- You may use 2 scratch papers.
- Calculators are not allowed.
- This is a closed-book exam. No additional resourced are allowed.
- Pay special attention to the style of your code. Indent your code correctly, choose meaningful names for your variables, define constants where needed, choose most suitable control statements, etc.
- In all questions you may assume that the users enter inputs as they are asked. For example, if the program expects a positive integer, you may assume that users will enter positive integers.
- No need to document your code in this exam, but you may add comments if you think they are needed for clarity.
- Read every question completely before answering it.

1. (3 pts) You are going to write a function which will be given the first node of a linked list and are asked to return the sum of the linked list (assume the values stored in the list can be added using the + operator). Which of the following would be the best function signature to use?

a) int sum(LListNode<t>* head)	c) int sum(LListNode head)
b) int sum(LListNode<t>*& head)	d) int sum(const LListNode<t>* head)

2. (3 pts) We would like to define a function which will exist in a base class and derived classes, the function will be overridden in the derived classes. How should we define this function?

   a. Any normal function will do
   b. The function must be virtual
   c. The function must be pure virtual
   d. The function must be abstract

3. (3 pts) Polymorphism allows for copying from a derived object to a base object, however how can we copy from a base class object to a derived?

   a. Overload operator= in the base class to accept a derived object as a parameter
   b. Overload operator= in the derived class to accept a base object as a parameter
   c. Implement a constructor in the derived class that accepts a base object as a parameter
   d. This cannot be done in C++.

4. (3 pts) One method for sorting items from an array would be to remove them from the array, store them in a _________________ and then return them to the array in order.
   a. A Queue
   b. A Stack
   c. A Binary Search Tree

5. (3 pts) Given a perfectly balanced, full, binary search tree of integers (leaves are all on the same level and all levels are populated fully), write a member function to find and return the median element of the tree. Use the given function header:

```
class BST{
        TreeNode* head;
        int getMedian(){
        }
}
```

6. (10 pts) You are given a stack of unsorted integers and asked to sort the stack using only one other stack for storage (no additional storage besides scalar variables). You are not concerned with the time complexity but are concerned with not using additional space! Explain, in English, not code, how you would perform this task.

7. (10 pts) Explain in English, you may mention functions and other snippets of code, how you would read an entire file and then determine which character is at the exact midpoint of the file. You must do so in constant space (i.e. you may NOT store the file in memory, it's too large!)

8. (20 pts) You are given a file, on the hard drive (file.txt; you are guaranteed that this file exists) which contains a set of names, one on each line. Names could be just first name (i.e. Madonna, Cher, etc) or multiple names (i.e. "John Dough" or "Wiley E. Coyote"). Write a program to determine the percentage of names that are first name only.

9. (25 pts) Given two pointers to the root nodes of two different, unbalanced, binary search trees (TreeNode<T>* ptr), write a function "compare" to determine if the two trees contain the same values. You may assume each node has a data section and left, right and parent pointer, but may not assume any other items exist in the nodes. Your solution should be as efficient as possible both in terms of space and time complexity.

10. (20 pts) We will be designing an inventory management system for a book store. Books come in two forms, paperback and hardcover. We are asked to design a set of classes which will be used to represent these books. Every book has a cost, but its cost is dependent on two factors, the production cost and the author cost. The author's cost is for writing the book and doesn't matter which format the book comes in. Production of the paperback book adds $5 whereas the hardcover book adds $25 to the cost.

Design three classes, one for a GenericBook which should store the author's cost as a double variable that can be accessed by any of the classes, one for PaperBackBook and one for HardCoverBook. All three should have a "price" function which returns the total cost of the book in the appropriate form, however, a GenericBook cannot produce a price since it doesn't have a format (hard cover or paperback), you must enforce this! The constructors for all classes should take the author's cost as a parameter.

You should create these three classes as efficiently as possible and the code below should work properly.

```
int main() {
       double totalValue = 0;
       vector<GenericBook*> inventory;
       inventory.push_back(new PaperBackBook(5.25)); //Actual cost is $10.25
       inventory.push_back(new HardCoverBook(3)); //Actual cost is $28
       inventory.push_back(new PaperBackBook(7)); //Actual cost is $12

       for (int i = 0; i < inventory.size(); i++)
              totalValue += inventory[i]->price();
       cout << "Total value of the inventory is: " << totalValue << endl;
       //Prints: $50.25
}
```