NYU, Tandon School of Engineering
Bridge to Computer Science Program — Summer 2020

Third Midterm Exam

- 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 and CLion
- 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) We would like to create a recursive function which allows us to pass the root of a tree and allows us to delete all of the nodes of the tree, while updating the root, which of the following would be acceptable (assume the BST and BSTNode class are similar to what was defined in our webinar)
   a. void deleteRoot(BSTNode* root)
   b. void deleteRoot(BSTNode*& root)
   c. void BSTNode::deleteRoot(BSTNode* root)
   [✓ d. void BST::deleteRoot(BSTNode* root)]

2. (3 pts) If we were modeling Cars in the current environment, we might model a Car as a base class but have derived classes for GasCar, HybridCar and ElectricCar. All cars consume energy but they do so differently, some electric, some gasoline, some a mixture of both. How should we define the "consume" function in the base class Car.

   a. void Car::consume(){cout<<"Using energy"<<endl;

   b. void Car::consume()=0;

   [✓ c. virtual void Car::consume()=0;]

   d. Don't define it in the base, just define it in the derived classes.

3. (3 pts) How can we make the following code work:
   Base bobj;
   Derived d = bobj;
   a. Overload operator= in the Base class
   b. [✓ Overload operator= in the Derived class]
   c. It is, automatically, allowed; nothing needs to be done
   d. This can never be done.

4. (3 pts) Convert the math expression 2+3+4*(6-5) to post fix form.

5. (3 pts) Evaluate the post fix expression "`3 2 3 * - 2 /`" to a value.

6. (15 pts) Given a binary search tree where only the pointers and the data value is stored in each node. Explain, in English not code, how you would, in Theta(N) time, determine if the tree is balanced. (Note: You DO NOT have a stored "height" parameter in each node)

7. (15 pts) Given pointers to the first nodes of two singularly linked lists (LListNode<T>*), provide a member function to produce a new list which is the concatenation of the two provided lists. You may assume that the function will return an LListNode<T>* pointer to the new list and should NOT modify the original lists.

8. (20 pts) Someone has given us a file filled with integer ID numbers for people who have visited the NYU campus. This file contains ONLY the ID numbers, one per line. Unfortunately, some of the ID numbers are listed multiple times. We would like to remove the duplicates but need to do so in better than Theta(N^2) time as there are many of them. Provide the code to
   a. Ask the user for a filename which you must ensure exists
   b. Open the file
   c. Read in all of the duplicates
   d. Remove the duplicates
   e. Write the results out to "output.txt"

   The steps above may be combined as you see fit, the goal is to produce output.txt which contains only unique integer ID values. You may use any STL data structure that you'd like.

9. (20 pts) Given a pointer to the root of a binary search tree and a given value, produce a STL List of all of the values in the tree that are LESS THAN or equal to the given value. Each node of the tree has left, right, and parent pointers as well as a data section. Your code should run in theta(N) time.

10. (15 pts) We're going to design a system that keeps track of Exams. Exam is a class that will implement a number of functions, not the least of which is getResults(). getResults will return an integer, but what it returns depends on the type of exam we're modeling. A WrittenExam results in positive points granted and so, will return a positive integer (use +1 for this question). A PracticalExam is measured in points lost and so, will return a negative number (use -1 for this question). Implement the three classes using inheritance.