NYU, Tandon School of Engineering
Bridge to Computer Science Program — Summer 2023 Extended

Third Midterm Exam

- You have two hours
- There are 100 points total.
- Note that there are longer 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.
- Write your name, netID and NYU ID at the head of the solutions file.
- For editing this file, you are allowed to use only plain text editors (Notepad for Windows users, or textEdit for Mac users).
- You are permitted to use Visual Studio (C++) or XCode as compilers. And Textedit/Notepad for text editing but should copy/paste your answers to the TXT file.
- 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.
- **When done, please upload your answer file to Brightspace.nyu.edu, Gradescope and email to dkatz@nyu.edu**

1. (5 pts)

```
int * first = new int(0);
*first = 100;
cout<<*first<<endl;
delete first;
```

What mistake has been made in the above code?

   A. Dereference null pointer;
   B. Dangling pointer;
   C. Memory leak;
   D. There is no mistake in the code;

2. (5 pts) Which of the following data structures would be helpful in sorting a set of data in less than quadratic time?
   A. Array/Vector
   B. Linked List
   C. Stack
   D. Tree

3. (5 pts) Which of the below is the correct declaration for an overloaded assignment operator for the Thing class which can support chained assignment (i.e. x=y=z)?
   A. `Thing& operator=(const Thing& rhs)`
   B. `Thing* operator=(const Thing rhs)`
   C. `Thing operator=(const Thing& rhs)`
   D. `void operator=(const Thing& rhs)`

4. (5pts) Given the following

```
class Thing {
public:
	void member();
};

int main() {
	Thing* ptr = new Thing();
	//Your code here
}
```

   Write one line of code to call the member function using ptr.

5. (5 pts) Convert the infix expression "**2+3*4+(5+6/7)**" to postfix form.

6. (5 pts) In a binary tree in which the root has only a right child, provide one line of code to print the data value in the child of the root. (variable names aren't important here, I'll understand what you mean)

7. (10 pts) Given a pointer to the first node of a binary search tree in which duplicates are allowed, explain in English an efficient solution for determining if there exist any duplicates in the tree. In your answer, provide the big-theta runtime of your algorithm and discuss memory usage.

8. (10 pts) Given a pointer to the first node of a doubly linked list (everything in the list class and node class are public so you can access everything) explain, in English, how you would determine if the list has any defects in its structure (any pointers incorrect).

9. (20 pts) A file "exam3.txt" contains information for each student's performance on exam 3 where each line contains one student's id (int) and their grade (double) separated by a space. The file exists on the hard drive you can trust it is there, no need to check.

   You are asked to produce two output files with the data sorted. The first file, "sortedbyID.txt" will have all of the data sorted by ID number. The second, "sortedbyGrade.txt" will have the data sorted by grade. Both files should maintain the format of student ID followed by grade separated by a space.

   For this problem, you may use any STL data structures and algorithms you like, or you may design your own. Efficiency is somewhat of a concern but the file will never contain more than 350 students (that would be a BIG cohort!).

10. (30 pts) Driving anywhere these days tends to cause us to incur tolls. Tolls are commonplace on bridges, tunnels and roadways and are often variable in nature since cars will pay a different price than trucks and buses. We would like to design a system to model vehicles going through a toll booth and keep track of the total amount of money earned by the toll booth in a day.

    You will need to design the following classes:

    - The TollBooth class. This class will record how much was earned by the toll booth each day and should provide an accessor function (getter) to return the total of all of the tolls collected. The class will also have a function "processToll" which will accept a pointer to a vehicle as a parameter (right now we're concerned about Trucks and Cars, but we may add other vehicle types in the future) and will cause the vehicle to incur the toll and adjust the tolbooth's total appropriately.
    - The Vehicle class. This class will have functions to "incurToll" and "getTollTotal". incurToll will cause the vehicle's tollTotal to increase by the appropriate amount and getTollTotal will return the total amount incurred at all TollBooths this Vehicle went through.
    - The Car class. To make this question a bit easier, we know that Cars are charged $5 at all tollbooths.
    - The Truck class. To make this question a bit easier, we know that Trucks are charged $25 at all tollbooths.