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

Third Midterm Exam

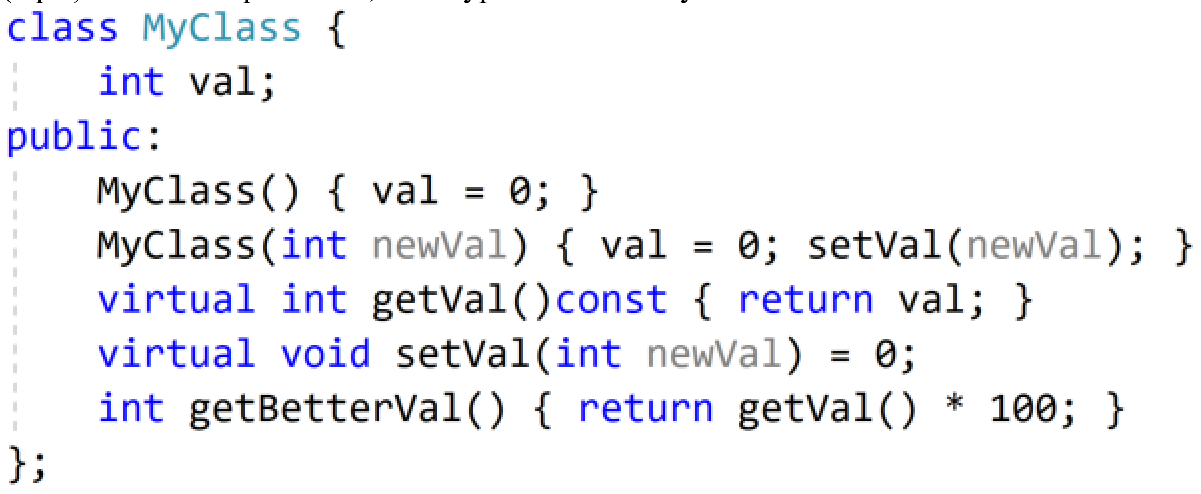
*****PLEASE REMEMBER TO SUBMIT ON:**
CLASSES,
GRADESCOPE
and EMAIL (dkatz@nyu.edu)*****

- 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 being asked to write a function which will return the first value stored in a doubly linked list. Your function will be a member of the class and will be called getFirstValue. Which of the following is the best function signature

a) int DLL::getFirstValue()	c) int DLL::getFirstValue() const
b) int& DLL::getFirstValue()const	d) const int DLL::getFistValue()

2. (3 pts) In the example below, what type of class is MyClass

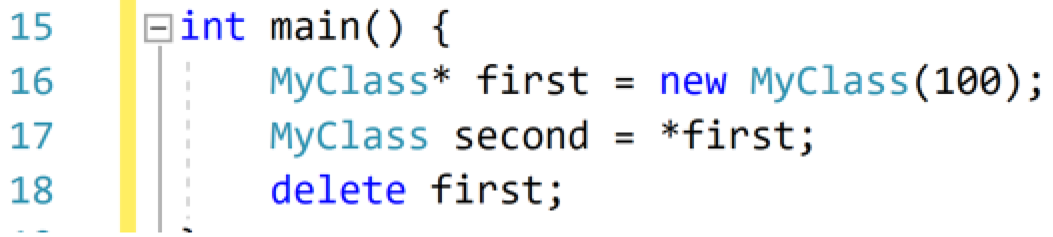


```
class MyClass {
    int val;
public:
    MyClass() { val = 0; }
    MyClass(int newVal) { val = 0; setVal(newVal); }
    virtual int getVal()const { return val; }
    virtual void setVal(int newVal) = 0;
    int getBetterVal() { return getVal() * 100; }
};
```

   a. A “normal” class (nothing special)
   b. An abstract class
   c. A derived class
   d. (there is a syntax error, this is not valid)

3. (3 pts) Please provide the equivalent post-fix form of the math expression 3*(4+5)/2

4. (6 pts, 2 each) In the following code, identify, by name, which functions from MyClass are being called?



```
15  int main() {
16      MyClass* first = new MyClass(100);
17      MyClass second = *first;
18      delete first;
```

   a. (2 Points) Identify, by name, which function is being called on line 16
   b. (2 Points) Identify, by name, which function is being called on line 17
   c. (2 Points) Identify, by name, which function is being called on line 18

5. (5 pts) Given the following insertions into an empty Binary Search Tree, what would be the output if the tree were processed in post-order? Insertions are: 7, 9, 3, 4, 5, 8, 2.

6. (25 pts) New York City's MTA provides the ability to get the location of all of the busses on a particular line. The function, "vector<double> getBusLineInfo(string busLine)" is provided, you do not need to define it. This function, when passed a line number like B57a (the B57 bus line towards downtown Brooklyn) returns a vector of the distances of each bus from its departing station.

   For example, the function might return a vector with the following values:

   [14 0 5.2 3.1 7]

   This means that there is a bus at the departure station, a bus 3.1 miles from the station, a bus 5.2 miles from the station, a bus 7 miles from the station and a bus 14 miles from the station.

   You are being asked to determine if there are any TWO busses that are "too close." We define too close as having a distance of less than 1 mile. Please define a function "bool bussesTooClose(string busLine)" and return true if there are two busses which are too close. Your code will be evaluated for accuracy AND efficiency.

7. (25 pts) A doubly linked list has been corrupt in such a way that every second "real" node has it's pointers reversed, dummy nodes are not impacted by this corruption. In other words, in the first node, the next pointer points to the second node, and the previous pointer points to the node before the first (the dummy node). However, in the second node, the next pointer points to the first node and the previous pointer points to the third node!

   Given an LListNode<T> pointer to the first real node (not the dummy), please write a function to "fix" the list. You do not need to fix any of the dummy nodes and may consider a list structure similar to the one we described in class except that the LListNode class is all public so you can access all of the pointers and data.

8. (30 pts) A retail seller (think Amazon) has a number of types of "deals" that can be applied to a purchase, these are discounts that can lower the price of merchandise sometimes to very low levels. A deal can either be a single value reduction (i.e. $5.50 off the purchase price), a single percent reduction (i.e. 10.2% off the purchase price) or it can be a multi-part discount ($6 off AND 10% discount where the 10% is applied AFTER the $6 off). Only one of the above three discounts can be applied to an item in an order.

   Your job is to write the above three classes (SinglePercentDiscount, SingleValueDiscount and MultiDiscount) and an "Order" class. The order will keep track of the purchase price of multiple items and will need to keep track of the discount applied to each item. You will, need to create the Order class, the three discount classes and any other classes you may need.

   Each of the three discount classes will need a constructor that takes the amount and/or the percent of the discount and a function called "double calculateFinalDiscount(double originalPrice) which returns the value of the discount. For example, if the discount is 10% a call such as "singlePercentDiscount::calculateFinalDiscount(10)" would return 1.0 (indicating a $1 discount should be applied to this item).

   The Order class will contain two vectors. The first vector will store doubles, which are the prices of the items in the order. The second vector will store discounts related to the items. The discounts will be associated with the items in the same index (i.e. price[0] is adjusted by discount[0]; price[1] is adjusted by discount[1]). Obviously, that means the price and discount vectors must be the same length at all times and not all items will have discounts! In the Order class, design a "totalCost" function which returns the total cost of the order (sum of all prices after discounts are applied). Also design a function "addItem" which accepts a price and its discount (passed as a pointer to an object) for storage in the vectors. If addItem is passed a "nullptr," that indicates no discount should be applied.

   An example of how these classes would be used is below:

```
int main() {
    Order firstOrder;
    SingleValueDiscount svd10(10); //$10 off
    SinglePercentDiscount spd5(.5); // 50% off
    MultiDiscount md(.102, 5.5); //10.2% and $5.50 off!
    firstOrder.addItem(20, &svd10); //Cost is $10 after discount
    firstOrder.addItem(100, &spd5); // Cost is $50 after discount
    firstOrder.addItem(20, &md); //Cost is $13.021
                                 //($20-5.5=14.5)
                                 //(14.5*.102 = 1.479)
                               //(14.5-1.479=13.021)
    firstOrder.addItem(20, nullptr); //Cost is $20, no discount
    cout << firstOrder.totalCost() << endl; //prints: 93.021
                                            //(10+50+13.021+20)
}
```