Programming Assignment #3
CS 202 Programming Systems

*****This program is about operator overloading *****

Programs will only be graded if they support operator overloading and inheritance.

Primary Goal for program #3:

The primary goal for program #3 is to experience operator overloading in an object oriented environment. We want to correctly create classes that properly support the copy constructor, destructor, and now the assignment operator when dynamic memory is involved. Remember if we need to implement a copy constructor for a derived class, they it MUST have an initialization list to cause the base class copy constructor to be invoked. Every class that manages dynamic memory must have a copy constructor, destructor, and assignment operator.

Your primary goal with program #3 is to apply the functionality of the operators, the appropriate arguments (operand data types) and returned types (residual values for the operators) to the following problem. We want to make sure an apply operators in hierarchies since we are exploring OOP this term. Think about how the operators are used and try to mimic the behavior in your own classes. Consider each of the following:

1. What is the residual value of the operator?
2. How is the returned type used?
3. Who controls the memory of the residual value?
    a. The client program (lvalue) or
    b. The residual value is a temporary controlled by the operator (rvalue).
    c. Make sure to pass (and return) by reference whenever possible!
4. Do the operands get modified?
    a. If not, then they should be constant arguments
    b. If it is a member, should it be a constant member function?
5. Should the operator be overloaded as a member function or friend?
    a. Member functions are selected when the first operand is always an object of the class.

Remember OOP:

With OOP the key idea is to break down the problem outlined (below) into small pieces and assign those responsibilities to individual classes. For each assignment, your job will be to create an OO design and program that shows how Object Oriented constructs could be used to solve a real-world problem. You will want to focus on how to design classes that are well structured, efficient, that work together, and where each class has a specific **"job"**. This time you are adding operator overloading as the new syntax for your solution!

Before you design this program, take a look back at your first two programs. Did you tie the data structures into the midst of your OO designs? Did the designs really become all about the data structure and not so much about the problem at hand? If so, realize that the object orientation programming rarely is about the data structure. Most data structures will have a "has a" relationship with the OO part of the problem. Nodes are not object oriented! They are a contained used to assist your data structure.

Program #3 – The Problem

How do you like your cell phone? I have two phones in front of me. An older phone that I loved (but is irreparably cracked) and a newish iPhone which sadly doesn't have all of the features I have become accustomed to. So, I keep the old cracked phone around. Have you ever done anything like this? Kept something old around because it just did things better or had all of your photos? Each phone has unique features but some I like better than others (I really miss the finger print reader….). Then of course I have my "land line" (do you have one of those); there isn't actually a physical phone hooked up to it so at some point I will discontinue its use, but my security system is tied to it, so I keep it active. My daughter has a satellite phone so that she can be in contact while working as a white water kayaking guide. There are so many options available to us!

When starting this assignment, plan to do this incrementally. Start small and only implement the data structure after you have implemented this hierarchy.

1. Your derived and base class need to **manage dynamic memory** so that you can experience the full breadth of the operators
2. **Extra credit** will be provided if the code (.cpp) supporting this hierarchy is submitted by Feb 19th with your .h submission.

The key to OOP is to create the design based on the problem prior to applying the data structure.

START with Building the INNER CORE:
For this programming assignment, we will be creating a program that keeps track of **three different types of communication devices**. Your job is to find at least three classes relevant to this problem that make sense to you. Two of these require the customer to pay for them and at least one should have a text based form of communication. **Look for similarities and use inheritance and break down the common elements, pushing those up to the base class!** Then, have the derived classes handle the differences. Remember to avoid getters as much as possible with these classes – instead implement the functions that actually work ON the data IN the classes that own the data!! This is where you will get the most benefit of OOP. Implement this part of the program first, applying operator overloading.

Apply Operator Overloading to these classes: An important part of this assignment is to implement a complete set of operators. The operators to support must include: =, +, +=, [], ==, !=, the relational/equality operators, and the ability to input/output data. You may apply these to a single class or use them throughout your hierarchy. As you decide how to apply the operators, make sure to stay within the rules of how the operators are expected to behave. You should try to apply operator overloading in as many classes as possible to get comfortable with the concept.

In Program #3, you CAN now write your own STRING data type, but it can't be the only place you use operator overloading (since it is provided in topic #6!). Therefore, if you implement a string class, the operators for that class "do not count" for this assignment. I highly recommend implementing this class!!

NEXT, AFTER THIS IS DONE build the Data Structure:
Once you have completed the hierarchy for three different types of communication devices, then it is time to start working on a collection of devices that you and your friends can use to communicate with each other. Think of this as building a network of interactivity between you and a close group of friends.

The data structure for program #3 will be a tree organized of contacts where each contact has a LLL of devices they use. You have an option with this – it can be a binary search tree or it can be a 2-3 tree. If you don't implement a 2-3 tree in program #3, you can wait until program #5 (in Java). It is expected that you will support insert, retrieval, display, and removal all. Individual removal is NOT expected for a balanced tree.