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 that copy constructors in a hierarchy require the use of initialization lists 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. Think about how the operators are used and try to mimic the behavior in your own classes. How is the returned type used? Who controls the memory of the residual value? The client program (lvalue) or the operator (rvalue). Make sure to pass (and return) by reference whenever possible!

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

Program #3 – The Problem

Have you ever thought about the apps that you use on your phone? How many of them do you actually use? Do you ever delete them when you find they aren't useful? Do you find some are annoying while others give some great information (or are fun to use)? Some apps provide basic features in the "free" mode but give you the option to pay to avoid advertisements or to get those extra special features that you always wanted. Are you tempted to pay? Or do you just keep thinking about it each time messages pop up? After awhile it can get confusing. So, we need some software to help out!

The Data: The data for this programming assignment will be about the various apps you use. You will want to keep track of things like the name of the app, what it does, what you like about it, whether or not you paid for it, if you would recommend it to others, and so on. You should make this useful so think about how you use apps (or your friends use apps) and gather information that would make the most sense. You will also want to keep the most recent history of what you have done with this app (like for a game what your high score is and for a weather app maybe a picture of when Portland was completely in the middle of a snow storm). **All of this data should be stored in an external data file. Although you should provide some user interactivity to add new apps and remove apps.**

Making it OO: Your job is to pick three different types of apps to support (e.g., games, business, weather, etc.). You will want to develop at least five classes, as normal. Make sure to use single inheritance in your design. To break down the problem, think about what is similar or different about different kinds of apps. Using inheritance, break down the common elements in the types of apps and push 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.

Searching: The primary reason to collect all of this data is to allow the user to search on various parts of the data (searching by kind of app, the subject, historical information, etc.). Any field should be able to be searched. For a description, be able to search for keywords!

Data Structure: The data structure for program #3 will be a tree. You have an option with this – the BST can be a balanced tree. This term the balanced tree that we will be implementing is a red-black tree. Or, you can wait until program #5 to implement a balanced tree (in Java). It is expected that you will support insert,

retrieval, display, and removal all. Individual removal is expected for a BST but not for a balanced tree.

Operator Overloading: The key part of this assignment is to implement a class with a complete set of operators. The operators to support must include: =, +, +=, ==, !=, the relational/equality operators, and the ability to input/output data. I imagine the [] would be useful as well. As you decide how to apply the operators, make sure to stay within the rules of how the operators are expected to behave. You may find that some operators don't apply at all. And, don't forget your copy constructor!

You should try to apply operator overloading in as many classes as possible to get comfortable with the concept. Although you CAN now write your own STRING data type, but it can't be the only place you use operator overloading (since we did that in topic #6!). Therefore, if you implement a string class, the operators for that class "do not count" for this assignment.

Questions to ask…about operator overloading

When using operator overloading, remember to ask yourself the following questions:

a) What should be the residual value (and what data type is this)?
   - *Never have a void returning operator!*

b) Should the result be a modifiable lvalue or an rvalue?
   - Lvalues are returned by reference, most rvalues are returned by value.

c) What are the data types of the operator's operands?
   - If the operand isn't changed, then make it a const
   - Remember to pass as a constant reference whenever possible.

d) Is the first operand always an object of class?
   - If so, then it could be a member function.

e) Can the operator be used with constant operands?
   - If the first operand can be a constant, and IF it is a member function, then it should be a constant member function.