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. 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 been watching Game of Thrones? Do you wonder who will ultimately sit on the Iron Throne? For a few seasons you may have thought that Daenerys Targaryen with her dragons would take over. Certainly no one thinks Cersei Lannister will prevail. But as each episode takes place, there are twists and turns. Unexpected events take place and now we question once again who will it be? Will it really be Jon Snow? In my opinion, the best shows are those where the end is not obvious. Maybe even better would be where the viewer has a hand in helping guide the outcomes of events.

For This Program: For this programming assignment, we will be creating the "Game of Crowns". You will get to write the script (read from an external data file), and at each decision point (e.g., an event, battle, birth, etc.) the user should be made to be part of the decision making as to their options (e.g., who is the victor, etc.). What if Jon Snow never found out that he is in fact a Targaryen and the rightful heir to the Iron Thrown?

You may base this game off of the Game of Thrones, but you do not need to. You may create your own script or base it off of a TV series, mini-series, or other favorite trilogy (oh, like Highlander? Who will be the only one left at the end?!!).

Making it OO: Your job is to find at least three classes relevant to this problem. For example, there is script (e.g., dialogue) leading to an event. Then there is an event which then leads to different outcomes for the characters. 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.

Data Structure: The data structure for program #3 will be a tree. You have an option with this – it can be a binary tree or it can be a balanced tree. This term the balanced tree that we will be implementing is a 2-3 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 binary tree but not for a balanced tree.

Important: Every node should represent an event that takes place needs a linear linked list of all of the characters affected by this event.

Operator Overloading: 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.

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.