CSIS 211 – Data Structures
Spring 2017
Lab #2 – Due Date: 03/28/17 at 11:55 pm
Remember to submit ALL files (.cpp,.h etc.) on Moodle – NO EMAILS – no exceptions
You may work in a group of 1-3 for this assignment
EACH group member should submit just to make sure I don't miss any while grading
IMPORTANT! – put EACH group member's name at the TOP of the CODE in comments
If you are a group of 1 put your name only at the top
Make sure and use good design techniques
This will be worth 50 points

For all Programs:
Do NOT forget to put ALL your groups names at the TOP of EACH FILE in comments.
Example:
//LAB 1
//Group: Kristina Shroyer, John Smith, Jane Doe
Do this EVEN if you have a group of 1
Since this is very important

Lab #2 – Classes, Inheritance, initializers, base class method reuse and overloaded methods

This is an object oriented program – it's BASED ON #16 on page 116 of your book but use these instructions because I have altered the book instructions a bit. NOTE: There is more than one right way to do this lab – but make sure you use inheritance and overload and reuse base class methods where indicated.

For this assignment you are going to create two class data types that could represent characters in a game. Your base class will be a class called **`Character`** and your derived class will be a class called CombatCharacter.

First create the `Character` data type (your base class):

- Design and implement an ADT that represents a character in a game. As you read the instructions remember "client" refers to the client program using the ADT.

- This ADT should have member variables that hold the following data: the character's name, the character's height, the character's weight, the character's tendency and the character's health. The tendency of a character is the goodness of a character and ranges from -1.0 (very bad) to 1.0 (very good) with 0.0 being neutral. The health of the character is represented as a percentage between 0.0 and 1.0 with 1.0 being 100% healthy.

- Your class should have one constructor – determine the parameters based on the information given next. Characters should begin with a neutral tendency and 100% health. The name, height and weight of the character should be initialized by the client program.

- Your class should have the following member methods:

a. **heal:** increases a character's health by a client-supplied percentage.
b. **injure:** decreases a character's health by a client-supplied percentage.
c. **takeAction:** changes a character's tendency. The member function should take one parameter between 0.1 and 0.9 (positive or negative) to increase or decrease a character's tendency. Do not let the tendency be less than -1.0 or greater than 1.0
d. **get and set methods for:** weight and height
e. **a get method for name** (no set they can't change their name)
f. you should only have get methods for: health and tendency (these are changed only via other methods like heal etc)
g. **displayAttributes:** displays all information about the character: name, height, weight, tendency and health.

Second create the `CombatCharacter` data type (your derived class):

- A **`CombatCharacter`** is a **`Character`** capable of fighting in combat and will carry a weapon and a protection. In addition, a **`CombatCharacter`** can die, once dead they can't be rejuvenated. A protection could be a shield or a magic barrier or anything that projects the character in combat. The **`CombatCharacter`** ADT should be everything a Character is plus it should have the following additional member variables: a weapon name, a weapon strength (between 1 and 10 with 10 being the strongest), a shield name and a shield strength (between 1 and 10 with 10 being the strongest).

- Design a constructor for the **`CombatCharacter`** class. Don't forget the base class must call the derived class constructor – see C++ interlude #1 if you forgot how. In addition to the client program passing in the name, weight and height of this character they should also pass in the weapon name. Don't forget to use intitializers!

- Design the following member methods for the **`CombatCharacter`** class:

a. **useWeapon:** this method allows the character to use their weapon but decreases their health by 5 percent, decreases their weapon strength by 1, and decreases their tendency by 0.2. Make sure and reuse base class methods where appropriate. Do not let the character use their weapon if their weapon strength is 0 or their health is 0.
b. **getAttacked:** this method is called when the character is attacked. This method will decrease the character's shield strength by 1. If the character's shield strength is at 0 this method should decrease the character's health by 15%. Make sure and reuse base class methods where appropriate.
c. **saveAnotherCharacter:** this method increases the character's tendency by 0.2 and increases their health by 5%. This method also increases the character's shield strength by 1. Do not let the character saveAnother if their health is zero.
d. **reloadWeapon:** this method increases the character's weapon strength by 1 but decreases their tendency by 0.1. Do not let the character reload their weapon if their health is zero.
e. **noHealth:** checks to see if a character has any health.
f. **Overload the display attributes method of the base Character class:** this method should do everything the base class method does (reuse) PLUS it should display the character's: weapon name, weapon strength, shield name, shield string and if the character has no health it should display the character is dead (unfortunately).
g. **You will need set methods and get methods for weapon and shield name**
h. **You should only have get methods for weapon strength and shield strength (no sets)**

Finally test your ADTs with a client program

- You may design main any way you want but make sure you create at least one CombatCharacter and one Character and test all the methods
- I recommend not making this too complicated just use it to test.