LAB #3 – Practice Classes

Each lab will begin with a brief demonstration by the TAs for the core concepts examined in this lab. As such, this document will not serve to tell you everything the TAs will in the demo. It is highly encouraged that you ask questions and take notes.

As long as you attended lab 1 or received prior permission for missing the lab, then you can receive any number of points back for finishing the lab prior to this lab #2. Make sure you get checked off at the beginning of lab #2 for credit!!!

Pair Programming
In this lab, you can choose a partner for pair programming. You must be checked off together. You only need one computer for pair programming. One person will be the driver, who controls the computer and codes, while the other is the navigator, who observes and advises. After 20 minutes, your TA will switch driver and navigator, continuing this pattern until the task is complete. Please read more about pair programming and the benefits: Pair Programming Student Handout

(2 pts) Design classes with a “Have a” and “Uses” relationship

Come up with a scenario when you would want to use multiple classes in a program. You must think of at least one “have a” relationship and one “uses” relationship. For example, our library class has a cart and patron, and the patron uses the cart in the library. You do not need to implement anything. You just need to brainstorm. 😊

- Describe each of the objects with the specific features and functionality.
- What is the “have a” relationship?
- What is the “uses” relationship?

(8 pts) Create .h and .cpp files with constructors, accessors, and mutators for Assignment 2 (Use Friday and Monday note slides to help with this!!!)

In Assignment 2, you need to create a patron, librarian, cart, and library classes. Create the .h and .cpp files for each of the classes with the appropriate members (all being private), mutator functions, accessor functions, and constructors.

- For example, here are some prototypes for the default constructor, mutators, accessors for the cart class to get you started. After you complete these, make sure it you can set and get cards.
  
cart(); //set books to NULL and num books to zero
book * get_books() const; //return address of array of books
void set_num_books(const int); //set num_books and make book array
int get_num_books() const; //return number of books in cart
Make sure you are using const in the appropriate places for all classes, and create a Makefile that compiles all of your .cpp files and makes an executable!!!

Extended Learning
Another concept we will learn about Wednesday and Friday of week 3 is the Big Three for classes that contain dynamic memory. You can look up the concept of the “Big Three” to help you with this (Google C++ Big Three)!!

First, work on the cart class, which has a dynamic array of books.

- Now, write the “Big Three” for the cart class. Here are the prototypes to get you started.
  cart(const cart &); //copy constructor to do deep copy
  ~cart(); //destructor to delete book array
  //assignment operator overload, same contents as copy constructor
  void operator=(const cart &);

Convince your TA that the constructors, mutators, accessors, and “Big Three” are working correctly for the cart class!!

If you have enough time, implement the “Big Three” for the library class!!

Remember, you and your partner will not receive lab credit if you do not get checked off before leaving each lab. Once you have a zero on a lab, then it cannot be changed because we have no way of knowing if you were there or not!!!