LAB #5
Practice Inheritance and Polymorphism

Remember, if you need to get Lab #4 graded, you need to show your lab to the TAs within 10 minutes of getting to lab, and 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 know if you were there or not!!!

Reminder: All of our labs involve paired programming. You do not have to keep the same partner for each lab, but you MUST work with someone in each lab!!! First, find a partner for this lab. It can be the same partner from the previous lab or a different partner.

(5 Pts) Finish Implementing our Vehicles paying tolls example.

So far, we have a vehicle.h, vehicle.cpp, bike.h, bike.cpp, and main.cpp files that we created in class. I want you to re-create this on your own, implementing the get_toll_amount() that we did as a class. It helps to type things on your own versus me giving you the code to download☺ Look on the slides to see the bike and vehicle classes we developed in class:

http://classes.engr.oregonstate.edu/eecs/spring2015/cs162-001/slides/slide11-notes.pdf

You should be able to create these two objects in the main.cpp file, and set and print the seats, along with getting the toll amount, for both objects. In other words, you need to have a set_seats() and get_seats() functions in the parent class, and get_toll_amt()in both classes for displaying the appropriate information with the amount of money you pay going over a toll, but you will only have one accessor and mutator for the seats to be inherited by all vehicles.

First, add a default constructor to the bike and vehicle classes!!!

Now, create a car, motorcycle, and skateboard classes, and set up the correct relationship with the vehicle and bike classes.

In Oregon, we value clean air!!! Therefore, vehicles without engines do not have to pay a toll on this bridge. However, vehicles that have an engine have to pay 20 per seat they have. Add the **get_toll_amount()** to all classes, and those with engines pay **seats*$20!!**

Now, you will make an array of vehicles in your main/application file based on how many vehicles you have at your house, i.e. vehicle v_array[2]={bike(1), car(4)}. **Observe what happens when you put different vehicles into the array**. **Does your bike pay a toll, i.e. v_array[0].get_toll_amount();?**

This is because it has already statically bound the object to the vehicle class. Let's dynamically bind these vehicles at runtime, i.e. vehicle *v_array[2]={new bike(1), new car(4)};

What do you notice when these vehicles pay a toll, i.e. **v_array[0]->get_toll_amount();**

(5 Pts) Add a Date to your main file.

Download these two files to the directory with the vehicle, bike, skateboard, motorcycle, and car classes. I have written the Date class for you to use. Include the Date.h to your main file, and compile the Date.o with the other .cpp/.o files.

http://classes.engr.oregonstate.edu/eecs/spring2015/cs162-001/labs/Date.o
http://classes.engr.oregonstate.edu/eecs/spring2015/cs162-001/labs/Date.h

Now, add a Date object to your vehicle class. This will represent the date you plan to drive the specific vehicle. **On Friday, the cars get a 50% discount, and Sundays, the motorcycles get a 50% discount.**

You need to modify the get_toll_amount() to reflect this change. You can set the date by using the **set_day(), set_month(), and set_year()** functions. **You can get the day of the week from the get_day_of_week() function, but this returns 0 – 6 for Sunday – Saturday.**

Describe the relationship between these 6 different classes!!!

Extended Learning: Abstract Base Classes
At this point, you may be thinking that you never make a vehicle, but rather use this for inheritance an polymorphism. Therefore, the get_toll_amount() function in this class might be setup as follows: **virtual int get_toll_amount()=0;**

This makes the vehicle class a pure virtual, abstract base class. Therefore, no objects can be directly made of the vehicle type.