Olliver Aikenhead

Lab Section: 1108

**Terminal Output**

////////////////////////////////

///// Constructor Tests /////

////////////////////////////////

Testing Derived Default ctor

Vehicle: Default-ctor

Car: Default-ctor

Testing Derived Parametrized ctor

Vehicle: Parametrized-ctor

Car: Parametrized-ctor

Testing Derived Copy ctor

Vehicle: Copy-ctor

Car: Copy-ctor

Testing Derived Assignment operator

Car: Assignment

/////////////////////////////////

///// Polymorphism Tests /////

/////////////////////////////////

Testing VIRTUAL Move Function for DERIVED Class Objects

Car: DRIVE to destination, with throttle @ 75

Testing Insertion operator<< Overload for BASE Class Objects

Car: Throttle: 0 @ [39.54, 119.82, 4500]

///////////////////////////////////////////////////

///// Polymorphic Base Class Pointer Tests /////

///////////////////////////////////////////////////

Testing VIRTUAL Move Function on Base Class Pointers

Car: DRIVE to destination, with throttle @ 75

Car: DRIVE to destination, with throttle @ 75

Car: DRIVE to destination, with throttle @ 75

Testing Insertion operator<< Overload for Base Class Pointers

Car: Throttle: 75 @ [37.77, 122.42, 52]

Car: Throttle: 75 @ [37.77, 122.42, 52]

Car: Throttle: 75 @ [37.77, 122.42, 52]

////////////////////////////

///// Tests Done /////

////////////////////////////

Car: Dtor

Vehicle: Dtor

Car: Dtor

Vehicle: Dtor

Car: Dtor

Vehicle: Dtor

**Code Explanation**

At the start of the program it shows what tests will be done first. The first tests done are the Derived Constructor Tests. First the derived default constructor is tested for Vehicle and Car. These are tested by calling the default constructors in the main function for Vehicle and Car, and in the function definition of these constructors the debug output prints to the screen. The next constructor set to be tested is the parametrized constructors for both Vehicle and Car. The main program just calls both and prints the debug output to the screen. After that the copy constructors for both classes are tested. The main function calls both copy constructors and they properly print the debug output to the terminal. The final constructor test is for the derived assignment operator. The main program calls both assignment operators for their debug output to be printed to the terminal.

After the constructor tests, the next tests to be done are the polymorphism tests. The first function tested is the virtual move function for derived class objects. The serialize function prints to the screen its debug output, and then the insertion operator overload for base class objects is tested. It overloads the values and replaces them and prints to the screen. After these tests are done, the next set of tests to be done are polymorphic base class pointer tests. The virtual move function is tested first. Three different arrays are created, and the values moved between all of them. The same thing happens with the insertion operator overload test for base class pointers. Three different arrays are created and all three are printed to the screen. The final portion of the tests just prints to the screen that each constructor has been deconstructed. The main function does not need to call the dtor’s, they do the work themselves and print the debug output to the screen for each dtor.

The hardest part about this project was trying to determine if I had written all of the functions correctly. At one point I forgot to write a function definition for serialize in Car.cpp, and it would not compile at all. It took about an hour worth of googling to see what the issue was. Once I realized I had to write a serialize function in Car.cpp as well the program was done. I need to get better at writing the operators and serialize functions, those give me the most trouble out of all the project requirements. To compile this project I used “cmake ..” in the build directory. After calling that command I use “make all” to create the files. Once the files have been compiled, I call “make install” and then the project executable is created after calling these commands and its name is “proj6”. To run my program type “./proj6” in the build directory.