//File for Vehicle.h

#pragma once

#ifndef Vehicle

#include <string>

#include "Person.h"

using namespace std;

class Vehicle

{

public:

Person Person;

string manufacter;

int cylinders;

};

#endif // !Vehicle

//File for Truck.h

#include <string>

#include <iostream>

#include "Vehicle.h"

using namespace std;

class Truck : public Vehicle

{

public:

double load;

int towing\_cap;

double Truck::GetLoad();

double Truck::GetTowing\_Cap();

int Truck::GetCylinders();

string Truck::GetManufacter();

Truck();

Truck(string manuf);

};

//File for Person.h

#pragma once

#include <string>

using namespace std;

class Person {

public:

Person();

Person(string theName);

Person(Person& theObject);

string getName();

Person& operator =(Person& rtSide);

friend istream& operator >> (istream& inStream, Person& personObject);

friend ostream& operator <<(ostream& outStream, Person& personObject);

~Person();

private:

string name;

};

//File for Person.cpp

#include "Person.h"

Person::Person()

{

name = "";

}

Person::~Person()

{

}

Person::Person(string theName)

{

name = theName;

}

Person::Person(Person& theObject)

{

name = theObject.getName();

}

string Person::getName()

{

return name;

}

Person& Person::operator =(Person& rtSide)

{

Person newPerson(rtSide.getName());

return newPerson;

}

istream& operator >> (istream& inStream, Person& personObject)

{

inStream >> personObject.name;

return inStream;

}

ostream& operator <<(ostream& outStream, Person& personObject) {

outStream << personObject.getName();

return outStream;

}

//File for Truck.cpp

#include "Truck.h"

//default constructor that sets values = 0

Truck::Truck()

{

load = 0;

towing\_cap = 0;

}

Truck::Truck (string manuf)

{

manufacter= manuf;

}

//Accessor Method for load

double Truck::GetLoad()

{

return load;

}

//Accessor Method for Towing\_cap

double Truck::GetTowing\_Cap()

{

return towing\_cap;

}

//Accessor Method for Cylinders

int Truck::GetCylinders()

{

return cylinders;

}

//Accessor Method for Manufacturer

string Truck::GetManufacter()

{

return manufacter;

}

Truck::Truck (Truck& Object)

{

Truck newTruck = (Object.GetCylinders);

}

Truck& Truck:: operator = (Truck& Object)

{

return Object;

}

istream& operator >> (istream& inStream, Truck& TruckObject)

{

inStream >> TruckObject.manufacter;

inStream >> TruckObject.cylinders;

inStream >> TruckObject.load;

inStream >> TruckObject.towing\_cap;

return inStream;

}

ostream& operator <<(ostream& outStream, Truck& TruckObject)

{

outStream << TruckObject.manufacter;

outStream << TruckObject.cylinders;

outStream << TruckObject.load;

outStream << TruckObject.towing\_cap;

return outStream;

}