Georgi Yazovaliyski

COS221

**Course Project Specification**

My project is going to be a pre-race management system which manages vehicles. Imagine a universe where cars race with trains and trucks, etc. So for the race the vehicles should be sorted properly. A user will be able to add / delete vehicles in a collection. The vehicles will be stored in a file every time an update in the collection is done. Records on this file will be sorted by HorsePower(HP). Every piece of vehicle has a HP, so it’s a reasonable property of the base class.

For sorting the vehicles (read from a file) in the first place I will use an array and a MergeSort algorithm and then pass them to the main Queue.

The collections for managing the vehicles will be Queue<Vehicle>. Whenever we want to add a vehicle in the sorted Queue, I will be using other 2 queues. I will split the main queue two. I will use Breadth-First Search to find the proper place where I can add the new vehicle so that after merging the two parts back, the main queue will still be sorted. The new car will be added on the front of the left queue.

The same process, but reverse will be done when we want to remove a car from the queue.

After we finish with the desired operations we quit the program and it saves the vehicle order back to the file.

Hierarchy:

Vehicle -> WheelFull / WheelLess -> Car/Truck/Train

Vehicle{

String BrandName;

Double GasConsumption;

virtual Double GetHP();

}

WheelFull: public Vehicle{

virtual Double GetHP();

String TyresBrandName;

}

WheelLess: public Vehicle{

virtual Double GetHP();

string Movement; // WATER, AIR, RAILS

}

Car: public WheelFull{

String WheelBrandName;

Double HorsePower;

Double GetHP();

}

Truck: public WheelFull {

Double ShipmentWeight;

Double GetHP();

}

Train: public WheelLess{

Integer AmountOfPassangersPossible;

String HorsePowerCode;

Double getHP(); // Will calculate HP by the HorsePowerCode

}

The getHP() function will do calculations for the different vehicles. For example the WheelLess vehicles, such as Trains, have HorsePowerCodes which represent HP in a special way. That’s why getHP will be a virtual function.