CS-224 Object Oriented Programming and Design Methodologies

Assignment 03

Feb 09, 2020

1 Guidelines

You need to submit this assignment on 23rd of February at 8 pm as the next assignment will be given on the same day. Some important guidelines about the assignment are as following:

- You need to do this assignment alone.
- You will submit your assignment to LMS.
- You need to follow the best programming practices
- Submit assignment on time; late submissions will not be accepted.
- Some assignments will require you to submit multiple files. Always Zip and send them.
- It is better to submit incomplete assignment than none at all.
- It is better to submit the work that you have done yourself than what you have plagiarized.
- It is strongly advised that you start working on the assignment the day you get it. Assignments WILL take time.
- Every assignment you submit should be a single zipped file containing all the other files. Suppose your name is John Doe and your id is 0022 so the name of the submitted file should be JohnDoe0022.zip
- DO NOT send your assignment to your instructor, if you do, your assignment will get ZERO for not following clear instructions.
- You can be called in for Viva for any assignment that you submit

2 Package Delivery System

For this assignment you will be creating a package delivery system. You need to think in terms of objects. The first object is the delivery truck `(Truck)`that can store 50 liters of petrol. The cost per liter of petrol is 2.73$. You will be using the sample file, `Drivers.txt` for this assignment. Your code should however take into account that if an entry is increased or reduced (5 lines per entry) it reads all the entries in the file (You are going to assume that there are no errors in the file). For example, if there is just one entry:

Elton John
34
218
9
7

Based on this entry, the driver's name is Elton John, his truck has 34 liters already, his total funds are 218$. His truck covers 9 km per liter if empty and 7 km per liter when loaded.

The trucks can carry 12 to 20 packages/boxes (which is the second object) with random dimensions. The length, width and height of every package can range from 5 to 30 inches. This means that you will need to declare a dynamic array of boxes for each truck and every box will have a different dimension. You can set the number of boxes and their dimensions randomly for every truck.

The function `loadTrucks()` reads the file `Drivers.txt`, and populates a liked list with all the truck information given in the file. Each Trucks' `load()` is called in this function.

The function `calculateCost()` calculates the total cost it will take the loaded truck to travel 60 km, drop the cargo and return empty based on the fuel consumption when the tank was full. The drivers need to fill the tank first before making the journey, the tank can be filled only once. Based on the amount of money they have, calculate if everyone can do the journey. Those trucks who are unable to make the journey will call their `unload` and be removed from the linked list.

The function `makeJourney()` updates all the trucks after making all the journey.

The function `unloadTrucks()` shows all the trucks information, once the journey is complete. It calls the `unload()` of every truck. A new file `Trip.txt` should be generated that will show the current state of all the Trucks that made the journey.

When unloading the boxes, show the volume of all the boxes and then deallocate the array of boxes. Once the trucks return, deallocate the linked list of all trucks after calculating the cost for the trip, how much money is left, how many litres of petrol are left.

The truck needs to have a `load()` and an `unload()` Function. When the trucks have been generated, the `load()` function should be called that will generate the boxes and put them inside the truck (It should show the dimensions of all the boxes). Once the journey is over, it should call the `unload()` function and unload all the boxes (It should show the dimensions of all the boxes).

3 Pigeons at HU

A sample code is given in Pigeons folder, if you run it you can see some pigeons are drawn at some random location. Go to `game.cpp` ⇒ `Game::run()`, you can see an array of 10 pigeons is created and being drawn in a loop. You should understand how this function is working. Mouse coordinates are also printed, you should learn it as well.

Your task is to remove this array implementation, and rather implement a linked list of pigeons. Initially no pigeon is drawn, then on every click on the screen, it should (create a new pigeon and) draw a pigeon at that location. On the right click, the last pigeon should be removed from the linked list (also from the drawing). Pigeons should not go beyond the boundaries of screen.

You should implement the linked list in a separate `(.cpp and .hpp)` file.

4 Some important points:

- Sample code is there for your benefit. If you are going to use it, understand how it works.
- In the sample code we are using structs. You need to change it appropriately to use classes. A general rule of thumb is that all attributes in a class are private.
- Implement Q1 prior to implementing Q2, it will help you to implement linked list.
- Where necessary, declare your own functions inside classes. Make sure why you would keep a function as private or public.
- You do not need to follow the code given exactly. You can make changes where you see fit provided that it makes sense.
- You need to define separate `*.hpp` and `*.cpp` files for all the classes.
- A tutorial for file I/O is given `http://www.cplusplus.com/doc/tutorial/files/`.
- You should take `www.cplusplus.com` and `www.cppreference.com` as primary web source to search about C++

5 Credits

Some questions in this assignment are derived from the work of Dr. Umair Azfar Khan.

– The End –