Imperial College London
Department of Computing

MSc in Computing Science — Assessed C++ Exercise 3

Issued:	Monday 21st November 2011
Due:	**Friday 9th December 2011**

1 The Exercise

Sleepy Town is suffering from severe traffic congestion problems caused by malfunctioning traffic lights. To discourage vehicles from coming into the city, Sleepy Town City Council would like to charge vehicles as they enter the city centre, according to the type of vehicle and the time of day. Read the description of the vehicle charging rules devised by the Council (provided below) and then:

1. Draw the corresponding UML class diagram.
2. Write appropriate class definitions in C++ which, together with the main program given in the file `vehicleMain.cpp` (which you should not alter), generate a simulation of the specific scenario described below.

2 Details of the Vehicle Charging System

1. Entry charges are given in terms of *units*. Vehicles will be charged different numbers of units depending on their type, the time of day and other factors.
2. The monetary cost of one unit is the same across all vehicle types. It is determined by the Council, and is initially set to £1.00 per unit.
3. For billing purposes, the total amount of money owed by each vehicle is tracked.
4. An attempt must not be made to charge any vehicle more than once on any given day.
5. Buses are charged according to the number of passengers on board. Buses carrying 20 or more passengers go free. Buses with fewer passengers are charged 5 units. A certain number of passengers may get on or off a bus from time to time.
6. Lorries are charged according to the number of axles they have. A lorry with n axles is always charged n units.
7. Cars of all types go free if they enter the city centre before 09h00 or after 18h00[1].
8. Petrol cars are always charged 2 units.
9. Diesel cars are charged 1 unit, unless their pollution emissions exceed a certain threshold, in which case they pay 3 units.
10. Diesel car emissions are measured in particulates per cubic metre (ppcm) and the qualifying threshold for this concession (in ppcm) is decided by the Council. The limit is initially set at 5 ppcm.

[1] You may assume that the times when vehicles enter the city are always rounded to the nearest hour.

3 The Specific Scenario to Simulate

You are given a main program which describes the following events (a) – (q):

a. A 12-axle lorry (number plate “L10RRY”) and a bus (number plate “T3SSA”) are registered with the council.

b. 10 passengers board the bus (T3SSA).

c. The 12-axle lorry (L10RRY) enters the city centre on 02/10/2011 at 10h00 hours.

d. The council has a meeting and decides that the basic unit charge should be set to £2.00, and that diesel car drivers with emission less than 10 ppcm should pay a reduced rate.

e. The bus (T3SSA) enters the city centre on 02/10/2011 at 11h00 hours.

f. 5 passengers leave the bus (T3SSA).

g. The bus (T3SSA) enters the city centre on 02/10/2011 at 13h00 hours.

h. A petrol car (number plate “M1LLY”) and a diesel car with an emissions rating of 15 ppcm (number plate “V4NLI”) are registered.

i. The petrol car (M1LLY) enters the city centre on 02/10/2011 at 16h00 hours.

j. The diesel car (V4NLI) enters the city centre on 02/10/2011 at 17h00 hours.

k. The council has a meeting and sets the basic unit charge to 2.50.

l. The diesel car (V4NLI) enters the city centre on 03/10/2011 at 02h00 hours.

m. 20 passengers board the bus (T3SSA).

n. The bus (T3SSA) enters on 03/10/2011 at 14h00 hours.

o. The council has a meeting and decides that Diesel cars with emissions less than 20 ppcm should pay a reduced rate.

p. The diesel car (V4NLI) enters the city centre on 01/01/2012 at 10h00 hours.

q. Finally, the 12-axle lorry (L10RRY) enters the city centre on 01/01/2012 at 11h00 hours.

4 Approach

We suggest you approach this exercise as follows:

1. Draw the UML diagram. Decide very carefully the functionality you’d need, and where it should be placed.
2. Design and implement your Vehicle class hierarchy using your UML diagram. Work with the diagram and the code iteratively. You may find it very useful to look at `vehicleMain.cpp` during this process.
3. Implement and test your class using the main program in `vehicleMain.cpp`.

The main program hints on how the problem could be solved. For all the classes you define, you need to ensure that your code includes the appropriate declarations and definitions so that the constructors and member functions used in the main program `vehicleMain.cpp` would work.
Note that repetitions in the exercise description may be implemented in an abstract manner by means of appropriate function definitions, or by the use of inheritance. The shorter and leaner your program is the better (but not at the expense of blank spaces and comments).
The output produced by your program should look like the sample output shown in Appendix B.

5 Submission

1. In your `makefile` name your executable `vehicle`.
2. Your program files will be compiled with the standard main program `vehicleMain.cpp` and assessed firstly by the compilation results and program output as well as on program design, style, Object Oriented features e.g. modularity etc.
3. You need to draw the UML diagram (although you need not submit it) and we recommend that you tackle the problem by working iteratively with the diagram and the code.
4. In makefile please name the executable 'vehicle', and put all files i.e. `*.h`, `*.cpp`, `makefile` which are necessary to run the simulation **excluding vehicleMain.cpp**[2] into a directory `vehicle`.

   Compress the directory `vehicle` using the tar command:

   ```
   tar -cvf vehicle.tar -C vehicle .
   ```

   Please note the *space* and the full stop '.' at the end of the command.

   Please submit the file `vehicle.tar` using the CATE submission system.

   END

[2] It won't actually do any harm if you include vehicleMain.cpp, but it will be overridden by a standard one for testing purposes, so please remember not to store any of *your* code in that file!

A vehicleMain.cpp

This appendix contains the main program you should use to test your classes. This will also be made available electronically.

```
#include <iostream>
#include "date.h"
#include "vehicle.h"

using namespace std;

int main () {

  cout << "====================================" << endl;
  cout << "Vehicle charging system operational!" << endl;
  cout << "====================================" << endl;

  Lorry lorry ( "L10RRY", 12 );
  Bus tessa ( "T3SSA" );

  tessa.board ( 10 );

  lorry.enter ( Date(2,10,2011), 10 );

  Vehicle::set_rate ( 2.00 );
  DieselCar::set_limit ( 10 );

  tessa.enter ( Date(2,10,2011), 11 );
  tessa.leave ( 5 );
  tessa.enter ( Date(2,10,2011), 13 );

  PetrolCar milly ( "M1LLY" );
  DieselCar vanilli ( "V4NLI", 15 );

  milly.enter ( Date(2,10,2011), 16 );
  vanilli.enter ( Date(2,10,2011), 17 );

  Vehicle::set_rate ( 2.50 );

  vanilli.enter ( Date(3,10,2011), 2 );

  tessa.board ( 20 );
  tessa.enter ( Date(3,10,2011), 14 );

  DieselCar::set_limit ( 20 );
  vanilli.enter ( Date(01,01,2012), 10 );
  lorry.enter ( Date(01,01,2012), 11 );

  cout << "=====================================" << endl;
  cout << "Vehicle charging system powered down!" << endl;
  cout << "=====================================" << endl;

  return 0;
}
```

B The Output

The output from the completed program should look *exactly* as follows. Note that you should use hashes (‘#’) rather than pound signs (‘£’) in your output, as the latter can cause problems in scripting languages, document rendering, *etc*.

```
====================================
Vehicle charging system operational!
====================================
***  A 12-axle lorry (L10RRY) has been registered

***  A bus (T3SSA) has been registered

***  10 passengers board the bus (T3SSA), so 10 passengers are on board

***  The 12-axle lorry (L10RRY) enters on 02/10/2011 at 10h00 hours
     The vehicle is charged #12.00 (now owes #12.00)

***  The council sets the basic unit charge to #2.00

***  The council says diesel cars with emissions less than 10.00 ppcm
     should pay a reduced rate

***  The bus (T3SSA) enters on 02/10/2011 at 11h00 hours
     The vehicle is charged #10.00 (now owes #10.00)

***  5 passengers leave the bus (T3SSA), leaving 5 passengers on board

***  The bus (T3SSA) enters on 02/10/2011 at 13h00 hours
     The vehicle has already been charged today ; no action is taken

***  A petrol car (M1LLY) has been registered

***  A 15-ppcm diesel car (V4NLI) has been registered

***  The petrol car (M1LLY) enters on 02/10/2011 at 16h00 hours
     The vehicle is charged #4.00 (now owes #4.00)

***  The 15-ppcm diesel car (V4NLI) enters on 02/10/2011 at 17h00 hours
     The vehicle is charged #6.00 (now owes #6.00)

***  The council sets the basic unit charge to #2.50

***  The 15-ppcm diesel car (V4NLI) enters on 03/10/2011 at 02h00 hours
     The vehicle goes free (now owes #6.00)

***  20 passengers board the bus (T3SSA), so 25 passengers are on board

***  The bus (T3SSA) enters on 03/10/2011 at 14h00 hours
     The vehicle goes free (now owes #10.00)

***  The council says diesel cars with emissions less than 20.00 ppcm
     should pay a reduced rate
```

```
***  The 15-ppcm diesel car (V4NLI) enters on 01/01/2012 at 10h00 hours
     The vehicle is charged #2.50 (now owes #8.50)

***  The 12-axle lorry (L10RRY) enters on 01/01/2012 at 11h00 hours
     The vehicle is charged #30.00 (now owes #42.00)

=====================================
Vehicle charging system powered down!
=====================================
```