1811ICT/2807ICT/7001ICT Programming Principles Workshop 10

School of Information and Communication Technology

Griffith University

|  |  |
| --- | --- |
| Goals | This workshop focusses on everything in the course up to object-oriented programming. |
| When | Week 11 |

# Before your workshop class:

* Read this whole document.
* Review the lecture notes sections 1 to 25.

# Workshop activities

## Problem 1a: Class Design

*Case study:* A GoCard account maintains a balance that may be spent on public transport. Users may request a statement that shows all transactions. The only transactions are:

* topping up the account with some positive number of dollars, and
* taking a ride costing some dollars and cents.

The goal for this exercise is to develop a class for a GoCard Account. The class will be tested by a program that simulates transactions, like this:

|  |
| --- |
| Creating account. Input initial balance: 100  ? r 3.50  ? r 10.90  ? b  Balance = $85.60  ? t 20  ? x gghhg  Bad command.  ? t  Bad command.  ? q  Statement:  event amount ($) balance ($)  Initial balance 100.00  Ride 3.50 96.50  Ride 10.90 85.60  Top up 20.00 105.60  Final balance 105.60 |

where:

* r *number* indicates a ride costing *number* dollars;
* t *number* indicates a top up of *number* dollars;
* b requests the current balance; and
* q ends input and prints a statement.

Bad/Incorrect inputs are to be reported and ignored.

Let us consider the design for a class that represents a GoCard account. To design a class, we consider what services the object(s) must provide (its methods), and what data needs to be stored in the object(s) to support those services. Questions:

* What is a good name for a class that represents a GoCard account?
  + Be descriptive of what the class represents. Don’t include the word “class” in the name.
* What services should be provided?
  + A constructor (\_\_init\_\_) is required to set up the account with an initial balance.
  + It needs to record the amount each ride costs. A method that accepts the amount as a parameter is required.
  + It needs to record the amount for each top-up. A method that accepts the amount as a parameter is required.
  + It needs to be able to report the current balance at any time. A method that returns this is required.
  + A method is required to print out a statement of all the transactions.

We can see from the output of the proposed program that the class needs to store the details of every transaction in order.

* What data is required to be stored in the object to enable those services?
  + So that a method can return the current balance at any time, it would be useful to have a field (attribute) for the current balance.
  + So that the full statement can be printed, the object must store the amount of each transaction, in order. Which data type can grow and keep multiple values in the order they are added?

## Problem 1b: Class Implementation

*Problem:* Implement the program described above.

*Testing:* Test your code with the example input and output provided in the simulation above (Problem 1a).



How would you change the code to read in the transactions from a file instead of user input?

## Problem 1c: File Input

*Problem:* Implement the program described above but extending it to use either user input or reading from a file.

*Testing:* Test your code with the example input and output provided in the simulation above (Problem 1a). Copy and past part of the text into a file and test whether you get the same result as with the user input.



How would you change the code to read transactions for more than 1 customer?

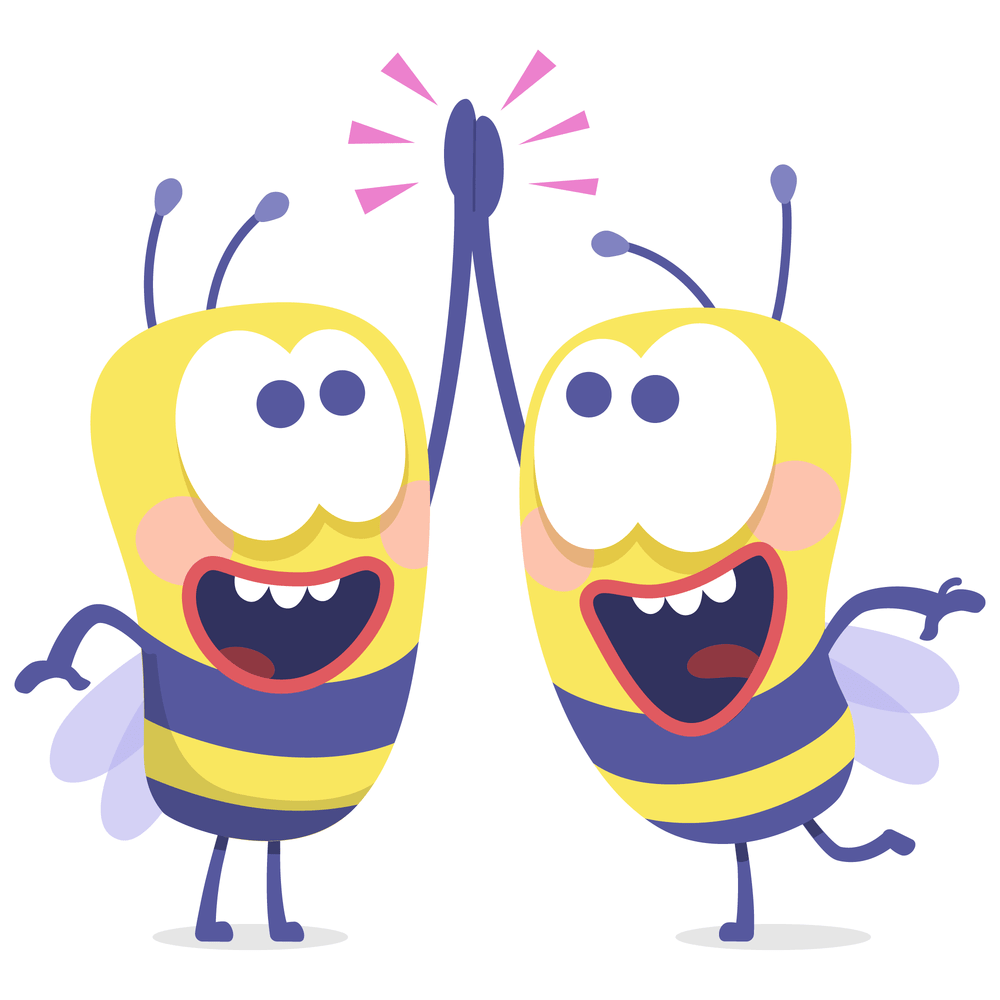
## Problem 1d: Inheritance

*Problem:* The GoCard system is being extended to have 2 different types of cards, each with their own concessions, as follows:

* Regular GoCard receiving 5% discount for the next 5 rides if a user used the GoCard 10 times (for 10 rides). Those transactions that receive the discount are not counted towards the 10 rides that result in the next 5% discount.
* Pensioner GoCard receiving 5% discount for all rides.

Extend your code where the GoCard class you defined and implemented above is your parent class (super class) and these 2 classes are your sub-classes inheriting from the GoCard class.

*Testing:* Test your code with the file you created for Problem 1c.



How would you change the code to have a person associated with a specific GoCard?

Well done for finishing these activities!