RMIT UNIVERSITY

# Software Engineering Design – Advanced Programming Techniques, Semester 2022C

# LAB ASSESSMENT 1 (25%) – TEST QUESTIONS

Test Duration: 120 mins (+ 15 mins for submission)

NOTE: only submit **one .cpp file for each question** (two files for two questions), and DON'T zip them together.

**Question 1 (10 pts)**

Write a program for a retail shop as below:

a) Prompt the user to enter the *total number of selling items*, then input **name** and **price** for each item. After that, save all information into a file namely "sellingItems.dat" in which each item is written in one line with the format **name : price.**

   Note that the names of items may consist of space characters.

b) Modify the program so that it will take **one argument from the command line**
   - If the argument is **-w**, it works as in question a
   - if the argument is **-r**, it read data from the file and print out all information

c) Continue to modify the program, so that:
   - If the argument is **-s**, it read data from the file and print out the information of the item with smallest price
   - Print out an error message, if there the argument does not match with any option (-w, -r, -s), or there is more than one argument inputted from the command line.

Sample Run:

```
g++ .\q1_retailShop.cpp; ./a.exe -w
Enter total number of items: 2
Enter name of item: coffee
 > price: 100
Enter name of item: bread
 > price: 200

g++ .\q1_retailShop.cpp; ./a.exe -r
Selling item list read from file:
coffee : 100
bread : 200

g++ .\q1_retailShop.cpp; ./a.exe -s
coffee : 100
```

## Question 2 (15 pts)

a) Define a class namely **EWallet** for online shopping with attributes **name** (*string*), **balance** (*double*). Provie a constructor for it to initialize the attributes.

- Write a method *bool **pay**(double amount, string coupon)*: to pay and deduct from the current **balance**.

  *The payment is discounted by 30% (only deduct 70% from balance) if the coupon is matched with the code "BLACKFRIDAY"**,** and the payment is false if does not have enough money*.

  *Note: You should print out appropriate message when the method is called.*

- Overload the >> operator we can do **double n >> EWallet obj**: increase the balance in obj by n.

b) Define another class namely **User** with two attributes are **name** (string), **pwd** (string), **acc** (EWallet object). Provide constructor for it, and two methods as below:

- bool **verifyPass**(): ask the user to input a password, and compare it with attribute **pwd**. Return true if match, and false if does not match with appropriate messages on screen.
- bool **doPayment**(): call the method **verifyPass**() first to verify password. If succeeded, allow user to input amount of money and coupon code for payment, then subtract it from current balance of the customer's ew object (hint: must call **pay**() method of ew object).

Overload the >> operator so that we can do **UserA >> UserB**: all money of UserA will be transferred to UserB

c) Create an array of **four customers** initialize values for them (*you can do it in your code, don't need to ask user to input*). Find and print out the customer with **second smallest balance** without doing sorting.