University of Nottingham
UK | CHINA | MALAYSIA

EEEE 2067 - Design and Implementation of Engineering Software
Department of Electrical and Electronic Engineering

Coursework Autumn 2024/25

"Intelligent Recycling Machine"

This coursework constitutes **40%** of your total assessment in this module.

LO Addressed

- LO1 Demonstrate critical judgment in decomposing large tasks into collections of small objects and functions.
- LO2 Design scalable object-oriented software with an appreciation of a larger environment encompassing code recycling, maintenance, expansion and issues of robustness.

Main Objective

Your main objective is to develop a smart recycling machine software according to the given requirements below. A typical recycle machine is shown in Figure 1.

Figure 1: A Typical Recycle Machine

Tasks or Requirements (What should this machine do?)

1. On the starting menu:
   a. Menu for **User Registration** and **login**.
   b. Menu for **Maintenance** login.

2. After login:
   a. Display **name**, **phone number** and **current points**.
   b. Display **item name** and **points per kg** as shown in Table 1.
   c. Display **maximum capacity** and **current load** as shown in Table 2.

   You can be creative on how to display the menu above.

3. When the machine is first powered on (when you run the program), it will read a text file (**item_point.txt**), which contains the information listed in Table 1.

Table 1: Recycle Item Initial Data

Item Name	**Points Per Kg**
Paper	10
Plastic	20
Glass	12
Metal	25
Cardboard	15
Hazardous Waste	30
Organic Waste	8
General Waste	2

4. The machine will then read another text file (**capacity.txt**) which contains the info listed in Table 2.

Table 2: Machine Capacity Initial Data

Item Name	**Maximum Capacity (kg)**	**Current Load (kg)**
Paper	50	0
Plastic	50	0
Cardboard	50	0
Glass	80	0
Metal	100	0
Hazardous Waste	60	0
Organic Waste	100	0
General Waste	90	0

If a particular item has reached its **maximum capacity**, it will not accept anymore and the deposit door will refuse to open.

5. **Point Collection System**: The system should provide a menu for user to **register** or **login**. For the first time, you need to register. It should use your **name** and **phone number.** Upon successful registration, you can login and the machine will display your name and **total point** collected so far. The system should also provide login for other users. Therefore, a **logout** function is also required. The system should also **retain** the account info after you program restarts.

6. **Maintenance mode**: Implement a maintenance mode for the machine operator. The operator should be able to **reset** a particular load to zero, or all those items which have **reached the maximum load**, or **all loads to zero**. It should be able to create a report of all past transactions. A menu option for selecting the maintenance mode should be provided in the **startup menu**.

Software Design

1. Design **at least 4 classes:**
   a. `RecycleItem` class to represent the object to be deposited by the user.
   b. `RecycleItemStatus` class to represent the status of a recycle item contained in the machine.
   c. `RecyclingMachine` class to represent the recycle machine.
   d. `User` class to represent the end user or person using the machine.

2. **Provide a UML class diagram** to illustrate the relationships between these classes. The basic members of each class are listed in Table 3, 4, 5 and 6 respectively. **Note: Your classes can have more variables and functions than what are listed in these tables.**

Table 3: Members of `RecycleItem` class

No.	Variable	Description	Type
1	`itemName`	Name of the item to be deposited e.g. paper, plastic, glass, etc.	`string`
2	`weight`	Weight of the item to be deposited in kg.	`int`
No.	**Function**	**Description**	**Type**
1	Member functions	Relevant assessors and mutators functions. Don't forget constructors and destructor.	function

Table 4: Members of `RecycleItemStatus` class

No.	Variable	Description	Type
1	`itemName`	Name of the item in the machine e.g. paper, plastic, glass, etc.	`string`
2	`maxCapacity`	Maximum capacity.	`int`
3	`currentLoad`	Current load of item in the machine. E.g. how many kg of paper in total in the machine.	`int`
4	`pointsPerKg`	Number of points per kg of item deposited.	`int`
No.	**Function**	**Description**	**Type**
1	`addToCurrentLoad()`	Function to add load to current item.	function
2	Member functions	Relevant assessors and mutators functions. Don't forget constructors and destructor.	function

Table 5: Members of `RecyclingMachine` class

No.	Variable	Description	Type
1	`paperStat, glassStat, …`	Machine will contain `RecycleItemStatus` for the recycled items.	`RecycleItemStatus`
2	`location`	Location of the machine. E.g. UNNC, Jintianfu, Manor, …	`string`
3	`usr1, usr2, …`	Machine will contain `User` objects to represent users who are registered.	`User`

No.	Function/Variable	Description	Type
1	`RecyclingMachine`	Constructor which initializes the recycling machine based on the data provided in Table 1 and 2 by reading **item_point.txt** and **capacity.txt**.	function
2	`startupMenu`	A function that allows the machine to display the startup menu as stated in Task1 .	function
3	`depositMenu`	A function that allows the machine to display the deposit menu as stated in Task2.	
3	`registerUser`	A function that allows machine to register user. The new user should be added to the file **account.txt**.	function
4	`userLogin`	A function that allows user to login into the machine. It should match the login with the existing account in **account.txt**.	function
5	`userLogout`	A function that allows the user to log out of the machine. Upon logout, the system should return to the login menu, allowing another user to log in.	function
6	`depositItem`	A function that allows user to deposit items to the machine after login. The user is allowed to deposit more than one type of item per time. Provide the right condition to handle the case when reaching the maximum capacity. The user is also allowed to cancel at any point and go back to the starting menu. Upon successful deposit operation, the function will update the current load in **capacity.txt** and don't forget to add points to the user.	function
7	`emptyLoad`	A function in the **maintenance mode** that allows operator to empty a particular type of item or the whole machine. Take note of the maximum capacity. This menu is only accessible by the machine operator.	function
8	`printTransaction`	A function in the **maintenance mode** that allows the machine to display the past transactions on screen also save the record with the filename "**Transaction-YYYY-MM-DD.txt**". This menu is only accessible by the machine operator.	function
9	Member functions	Relevant assessors and mutators functions. Don't forget destructor.	functions
10	Advanced function(s)	Create **at least one special function** which is not stated in the requirements. Suggestion: increase the number of locations for the recycling machine. Even if the resident's information is linked, if you are not in the apartment where the current recycling machine is located, they can only dispose of items as a	function

		guest, meaning they are still limited to five disposal attempts. You should also demonstrate this feature according to demo step in the next section. Note: match the "location" variable in RecyclingMachine class with "residence" variable in User class as follows.	

Table 6: Members of User class

No.	Variable	Description	Type
1	name	The account holder's name.	string
2	number	The account holder phone number.	string
3	points	The accumulated reward points of the account holder.	double
4	residence	The residence name of the user e.g. "Block215". If the user is unlinked to this residence, this variable should be "None".	string
No.	**Function**	**Description**	**Type**
1	Member functions	Relevant assessors and mutators functions such as addPoints, setResidence, etc. Don't forget constructors and destructor.	functions

Demo Step

Create a video with voice narrative to demonstrate the following tasks:

Initial conditions:

- The load of recycle items in the machine should be zero, meaning the machine is empty.
- No user has been registered.

Table 7: Task List

Task list	Task Description
1	Demonstrate initialization of the recycling machine with items listed in **item_point.txt**. Proof that you can change the category' name, points per kg of an item during the demo by changing the **item_point.txt**.
2	Startup Menu -> Create an account with your name and phone number. Store the account info in **account.txt** file and read the file every time someone is trying to register to make sure the person has not registered. When you login, you need to match the account with your phone number. After login, it will go into Deposit Menu which displays how many points you already have which is zero when first registered.
3	Deposit Menu -> select an item to deposit -> enter weight -> machine opens compartment door of that item -> user deposit -> success -> display points you have earned and also total point accumulated. After 5 seconds, it will go back to the deposit menu automatically.
4	Deposit Menu -> select any item -> cancel -> select another item -> enter weight -> machine opens compartment doors of that item -> user deposit -> success -> display points you have earned and also total point accumulated. After 5 seconds, it will go back to the deposit menu automatically.
5	Deposit Menu -> select item1 -> enter weight -> select item2 -> enter weight -> machine open compartment door of the 2 items -> user deposit -> success -> display points you have earned and also total point accumulated. After 5 seconds, it will go back to the deposit menu automatically.
6	Deposit Menu -> select an item -> enter weight more than capacity -> display "Exceeded capacity, door will not open". After 5 seconds, it will go back to the deposit menu automatically.
7	The points earned can be withdrawn once the total number points reach 100. Every 100 points is equivalent to one yuan, and the corresponding points will be deducted from the user's account. Create a menu to perform this transaction. Assume money has been received by Alipay or Weixin.
8	In Maintenance Mode -> show current capacity and load -> empty a particular item -> show current capacity and load.
9	In Maintenance Mode -> show current capacity and load -> empty the item that have reached maximum capacity -> show current capacity and load.
10	In Maintenance Mode -> show current capacity and load -> empty all item -> show current capacity and load.
11	Print all past transactions on the screen.
12	These transactions should have already been saved in a text file with "**Transaction-YYYYMMDD.txt**" format, E.g. Transaction-20241010, which records all transactions

	performed on that day. **Do not clear the records when the program is restarted. In your video demo, just show the transactions for that day.**
13	User needs to link his or her resident's details. If the resident is linked, there will be no limit on the number of times he or she can deposit. Otherwise, he or she will be limited to five times. If the user is not linked to the resident number and has already deposited five times -> a "deposit limit is reached" message will be displayed. To test this feature, one user (account) should not be linked to the residence number.
14++	Show your special feature(s).

IMPORTANT NOTE: Each run result should be saved in the corresponding file, and restarting the program should not overwrite the historical records.

Reminders:

(1) Provide adequate comments to enhance the readability of your codes.

(2) The Recycling machine program should comprise of multiple files e.g. headers, implementation and main driver (e.g. *.h, *.cpp and main.cpp).

(3) Submission deadline is **5PM, 19 December 2024**. 5% (out of 100% of this CW) will be deducted per day of late submission.

(4) The following items are to be submitted to Moodle

a. **Code package**. A zip file EEEE2067CW-Name **(E.g. EEEE2067CW-DavidChieng.zip)** containing your code, compiled binaries and project file in one folder. Please check if you .exe file can run in a different machine.

b. **UML design**. A report containing your design and brief description (max 2 pages, one page for diagram and another page for description. Remember to describe your advance features here).

c. **Video demo**. Share it using OneDrive and put the link together with the UML design.

Assessment Rubrics

This project comprises of 4 parts and each part can be an independent assignment according to a typical software engineering process.

Category	Marks (%)	Description (What is excellent?)	What to submit?
1. Design	20	Design a UML class diagram which captures all the main requirements of the project specification including special features. Also clearly illustrates the relationship between the classes.	3 page-PDF file
2. Coding	30	Correct implementation of classes and main driver. Usage of pointers, arrays, DMA, constructor and destructor. Clear and concise remarks which explain the code. Separation of header, implementation and driver files in support of code recycling, and maintenance.	Zip file
3. Video Demo	20	Clearly demonstrate the basic features as required by the task list. Also demonstrate advanced features.	OneDrive link containing the video demo. Put the link in the 3-page PDF file in 1.
4. Testing	30	• **Functionality Test** – demonstrate the required and advanced features. • **User Interface Test** - Nice and intuitive user interface, easy to operate. • **Robustness Test** - Able to handle wrong inputs with input verifications. The program doesn't crash. Using exception handing is highly recommended.	Nothing to submit here.
Total	**100**		