**Hong Kong Institute of Vocational Education (Tsing Yi)**

**Department of Information Technology**

**HD in Software Engineering**

**Contemporary Topics in Software Engineering**

**Assignment Report**

**Date: 2015-11-22**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student No** | **Student Name** | **Contribution to the project (%)**  **(Total 100%)** | **Signature** |
| **140016008** | **Ting Yu Fai** | **50%** |  |
| **140197442** | **Sit On Wa** | **50%** |  |

We declare that this is a group project and that no part of this submission has been copied from any other student’s work or from any other source except where due acknowledgement is made explicitly in the text, nor has any part been written for us by another person.

Contents

[1. Assumptions regarding the problem context 2](#_Toc435992674)

[2. Application design with class diagram 3](#_Toc435992675)

[3. Discussion and explanation on each of the design patterns applied to the application 4](#_Toc435992676)

[4. User Guide 7](#_Toc435992677)

[5. Test Plan and Test Cases 11](#_Toc435992678)

[6. Well documented Source Code 22](#_Toc435992679)

# Assumptions regarding the problem context

*we would like to create the new system before, we have the problems:*

1. The company manages different subsidy companies’ membership programs. Therefore, the company has more than one system and database to handle the subsidy companies.

2. The user have to create, search, renew, and close accounts in different membership programs.

3.If the company have new subsidy company, the company need to create a new system for the new subsidy company.

4. The old system cannot to decide membership. Hence, the user need to decide for yourself to use the correct system.

5.All systems of the company are no connection between them. If they want to communicate with other system, the user need to do for yourself.

6.If the company want to update or create the new function for all systems, the programmer have to update more than one system.

s

*The new system module should provide the following functions:*

1. Create a membership in a company (Africa Java Coffee Series shop (AJCS), or Wo-long Tea Series shop (WLTS), or any new company in the coming future).

2. Show membership details, by given id. (input id = \*999 to show all records)

3. Update the address of the membership, if membership is valid.

4. Extend membership expiry date

5. Undo last command ss

6. Redo the last undone command

# Application design with class diagram



# Discussion and explanation on each of the design patterns applied to the application

Adapter Pattern



We use a Memberadapter to lets classes work together that couldn't otherwise because ofincompatible interfaces. Those adapter interact with different client object or member object. In the original situation, member and client didn’t know each other, so we create a adapter to let the target can associate with the adaptee.

Factory Pattern



We use the factory pattern to create the command object and the member, client object. It hide how objects are created and help make the overall system independent of how its objects are created and composed.

Command Pattern



We use command pattern to do the different type of the action such as the create member method. Command pattern obtain the execute method which contain the code with corresponding action. Encapsulate a request as an object, thereby letting you parameterize clients with different requests, queue or log requests, and support undoable operations.

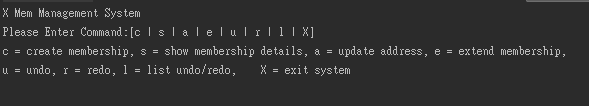
Memento Pattern



We use the memento pattern to do the redo/undo method. We have a caretaker object to store a list of the undo and redo. When we execute the Command.execute method we will create a new memento with corresponding action. After we create the memento object. We will add to the caretaker undo list. Those memento save an object in every different state. It means we make a “snapshot” when the command execute. So we can use the caretaker to control the undo or redo with the memento that can provide the state or date for us.

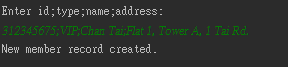
# User Guide

1.Enter the command including c, s, a, e, u, r, l, X.

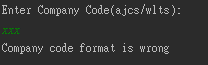


2.Command c is creating member. You have to enter the company code(ajcs or wlts) to choose your company. Then, you need to enter the member ID, type, name and address to store the member information.

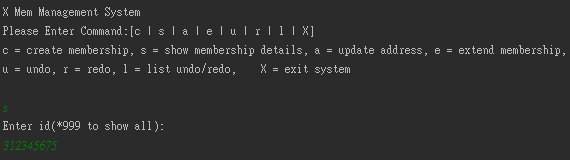
if you create member successfully.



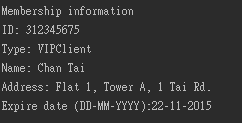
if you enter the company code being wrong.



3.Command s is showing the member details. You have to enter the member ID or \*999(show all member details) to choose your member details.



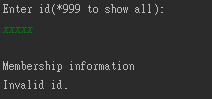
if you enter member ID to choose one member details.



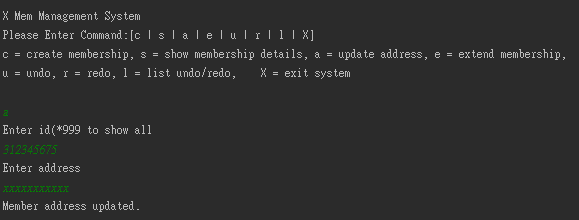
if you enter \*999 to show all member details.



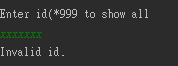
if the member ID of your entered is wrong.



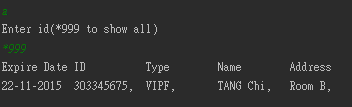
4.Command a is updating address. You have to enter the member ID to choose your member of you want to update. Then, you need to enter the new address on command line. The system will show the member address updated message to you.



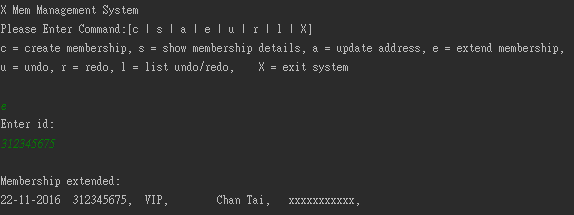
if the member ID of your entered is wrong.



if you enter the \*999 command, The system show all member details.



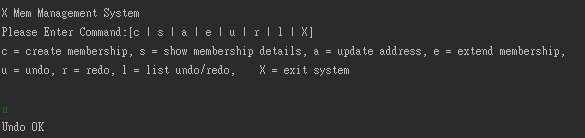
5.The command e is extending the member the Expire date. This function can increase one year to the member' expire date. Also, you need to enter the member id. The system will show the extended member details.



if the member ID of your entered is wrong.



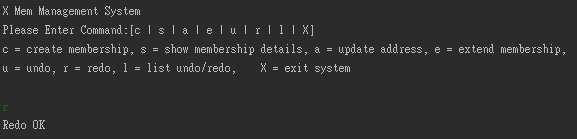
6.The command u can undo the system. The system will show the undo OK message and go to previous of the system when you enter the u command.



if you do not do anything such as creating member and updating address before on the system, y cannot undo anything. The system will show the empty of the undo list message.



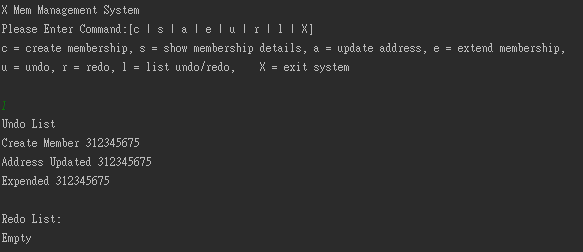
7.The command r can redo the system. The system will show the redo OK message and go to next of the system when you enter the r command.



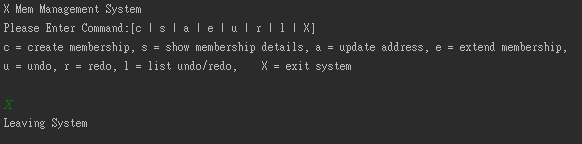
if you do not use the undo function before on the system. you cannot use the redo and the system will show the redo list empty message.



8. The command l is showing the undo and redo list. Also, if list is null, the system will show the empty.



9.Command X can exit the system.



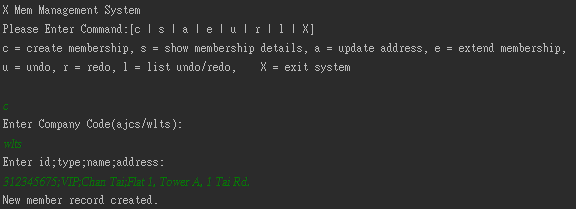
# Test Plan and Test Cases

Test Plan

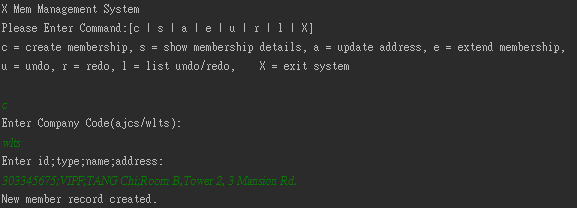
|  |  |  |
| --- | --- | --- |
| Command | Expected Result | Tested Result |
| CreateCommand(wlts),( 312345675;VIP;Chan Tai;Flat 1, Tower A, 1 Tai Rd.) | Create Successful | Create Successful |
| CreateCommand(ajcs),(21234561;Pri;Chan Siu;Flat 3, Tower C, 21 Tai Rd.) | Create Successful | Create Successful |
| CreateCommand(error) | Display error message | Display error message |
| CreateCommand(ajcs),(error) | Display error message | Display error message |
| CreateCommand(wlts),(error) | Display error message | Display error message |
| CreateCommand(error) | Display error message | Display error message |
| DrawCommand  (\*999) | Display all member | Display all member |
| DrawCommand  (312345675) | Display Chan Tai | Display Chan Tai |
| DrawCommand  (12345678) | Display error message | Display error message |
| UpdateCommand  (312345675),(12345) | Update  Chan Tai Address  (Flat 1, Tower A, 1 Tai Rd. to 12345) | Update  Chan Tai Address  (Flat 1, Tower A, 1 Tai Rd. to 12345) |
| UpdateCommand  (error) | Display error message | Display error message |
| ExtendCommand  (21234561) | Extend 1 year to Chan Siu  (22-11-2016 to 22-11-2017) | Extend 1 year to Chan Siu  (22-11-2016 to 22-11-2017) |
| ExtendCommand  (error) | Display error message | Display error message |
| ListCommand | List all Undo List and Redo List | List all Undo List and Redo List |
| UndoCommand | Undo List the last one will be move to Redo List | Undo List the last one will be move to Redo List |
| UndoCommand(empty) | Display error message | Display error message |
| RedoCommand | Redo List the last one will be move to Undo List | Redo List the last one will be move to Undo List |
| RedoCommand(empty) | Display error message | Display error message |
| ExitCommand | Exit the system | Exit the system |

Test Cases

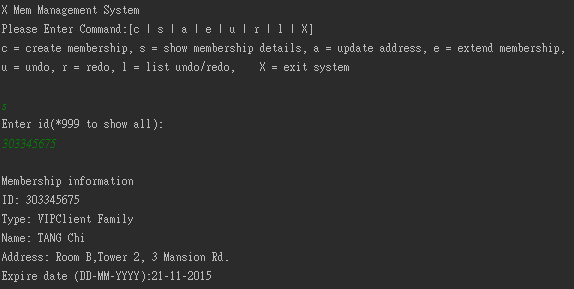
1. Create Member Information (c)



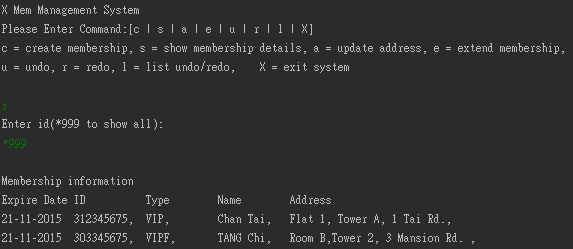
Create family member



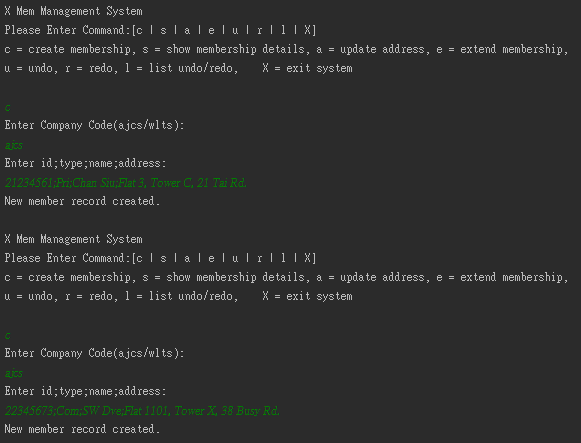
2. Show one member information (s)



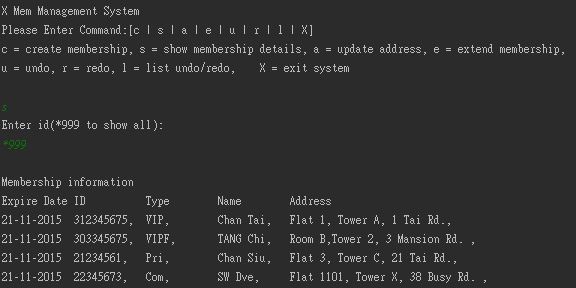
Show all records



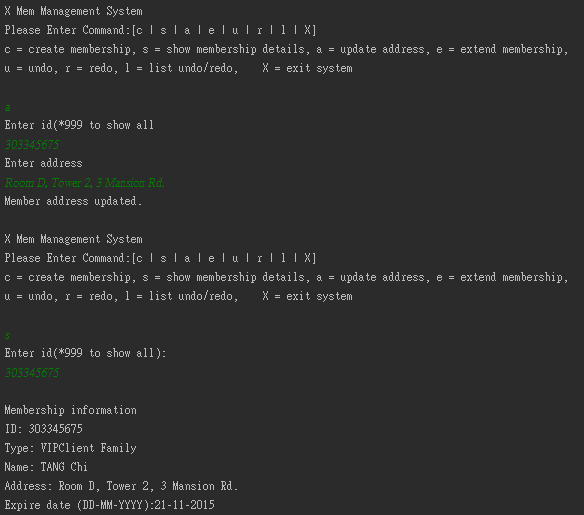
Create 2 members (1 primary member, and 1company member)



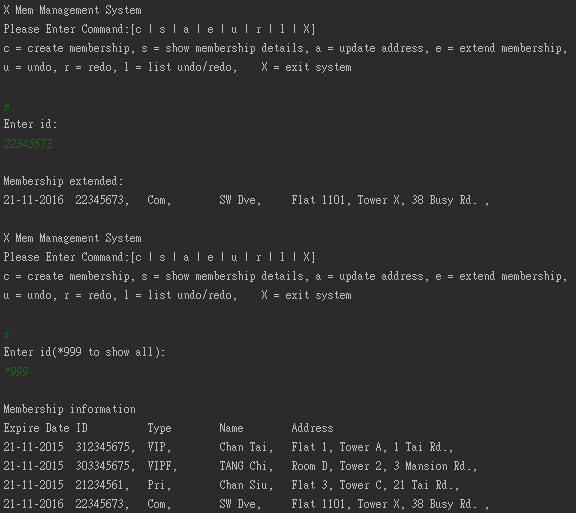
Show all records



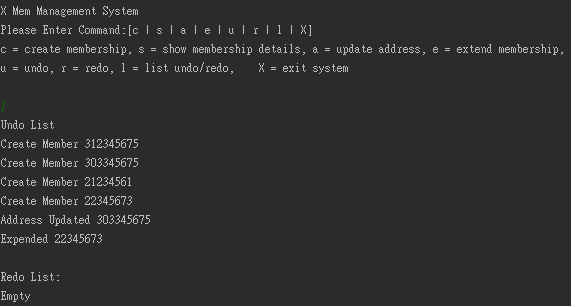
3. Update address (a)



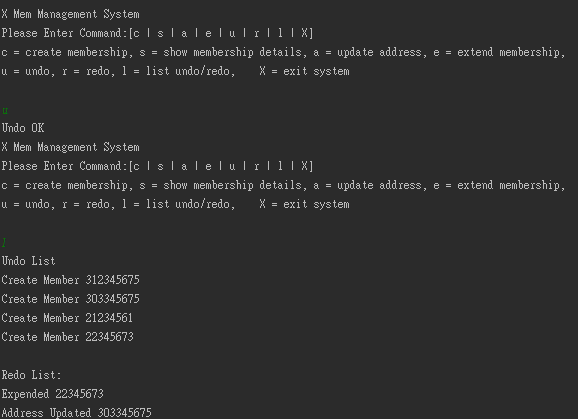
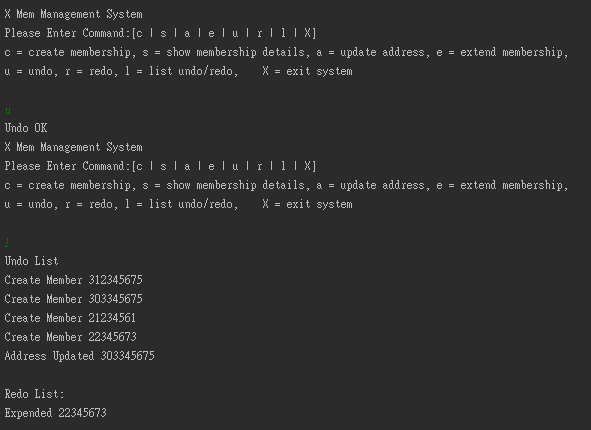
4. Extend membership (e)



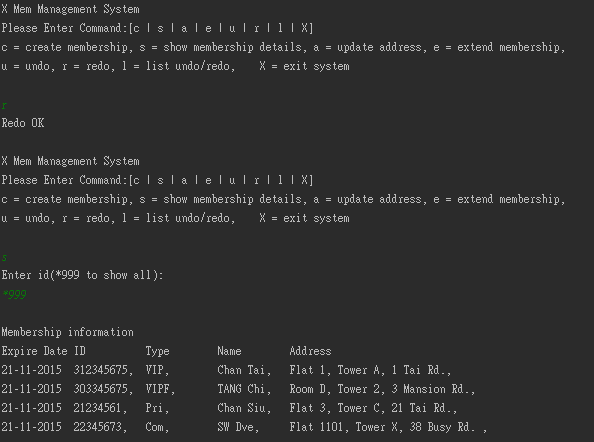
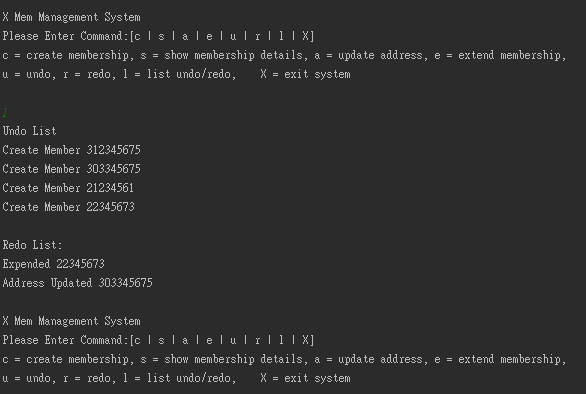
5. Display the Undo/Redo List (l)



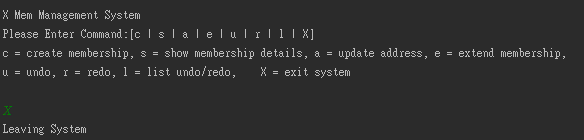
6. Undo Last Command in the Undo List (u)



7. Redo the last undo command (r)

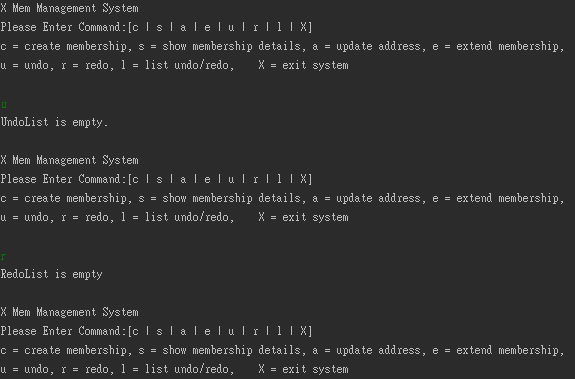


8. Exit the System (X)

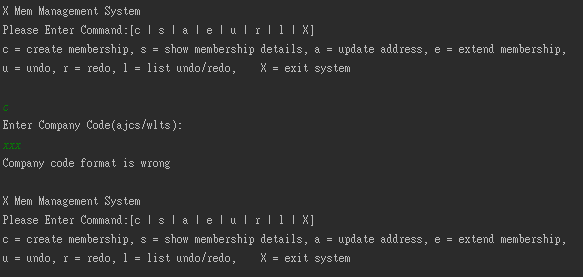


9. All Exception

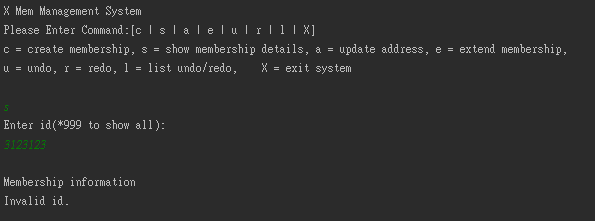
Undo and redo exception



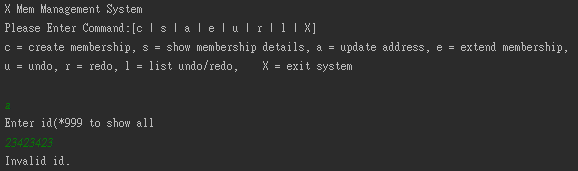
Create member exception



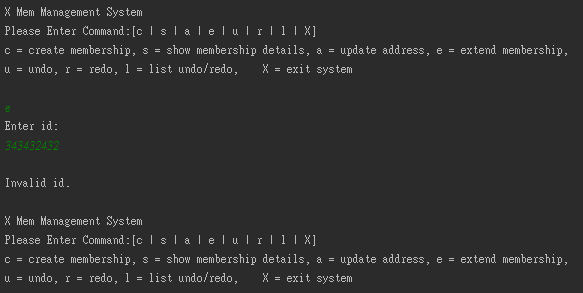
Show membership details exception



Update address exception



Extend membership exception



# Well documented Source Code

