IT114105 ITP4507

190765242

Chan Tsz Ho

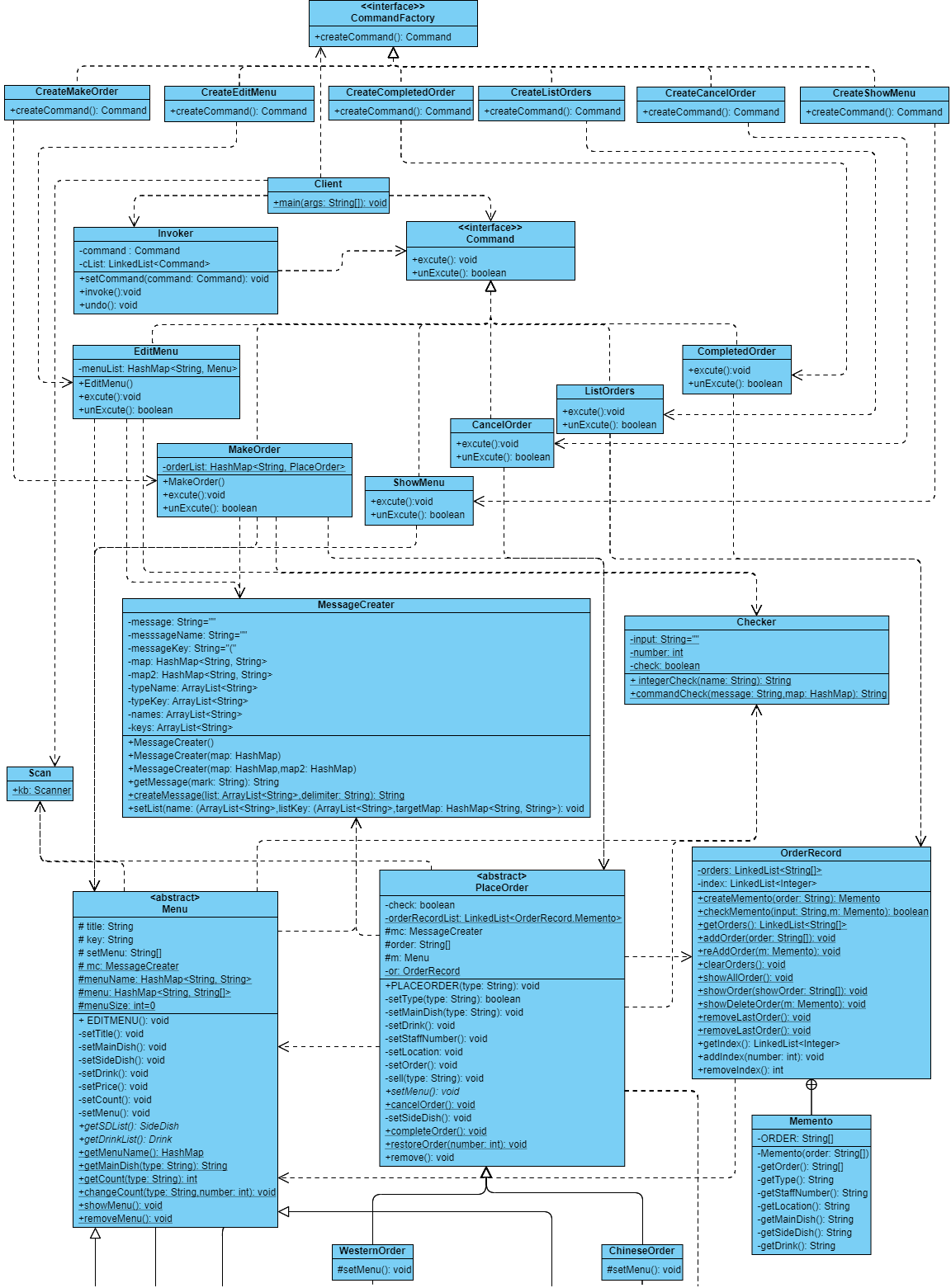
Assignment

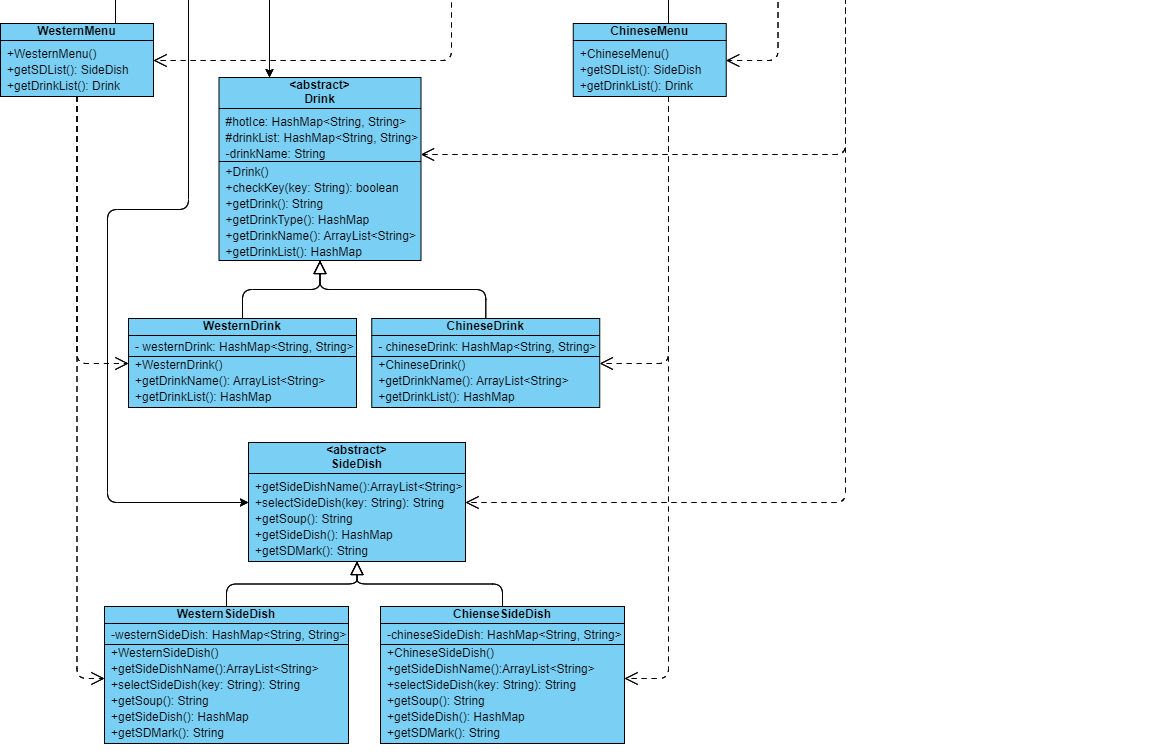
**Assumptions**

Regarding to the problem context, I have the following assumptions:

1. This system will not provide any eat-in service.
2. Each menu must include main dish, side dish, soup and drink one of each.
3. Staff Number and Office Location are composed of numbers
4. Each order sell one lunch set only.
5. Same staff do not allow buy more than one time.
6. The order will complete in order.
7. More style of menu will be provided in the future
8. The system is staff use only, so that no need to provide different permission account
9. System will not shut down before service time over
10. Menu must edit before open.
11. Staff must edit menu first before they using other function on this program.

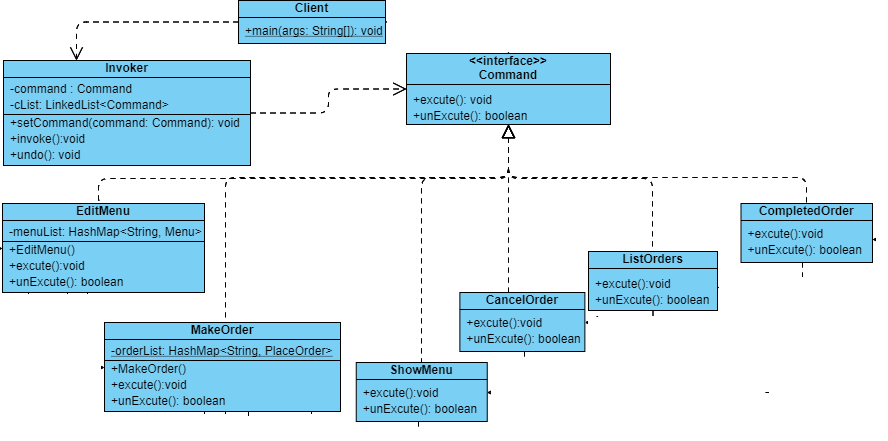
**class diagram**

****

****

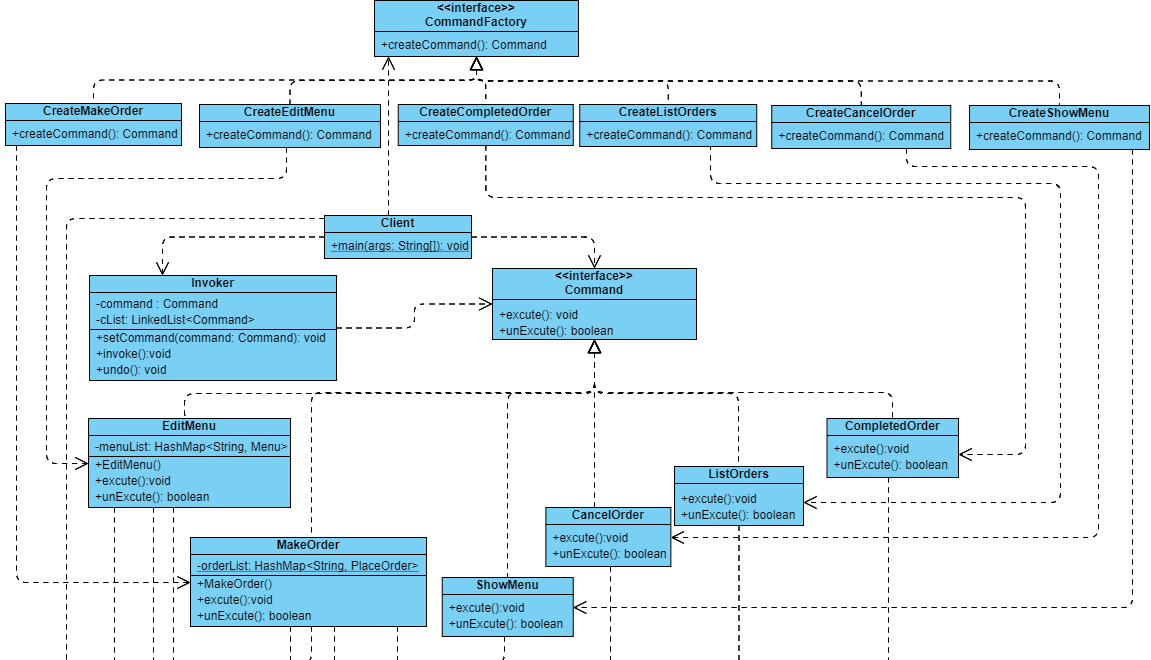
**design patterns applied to the application**

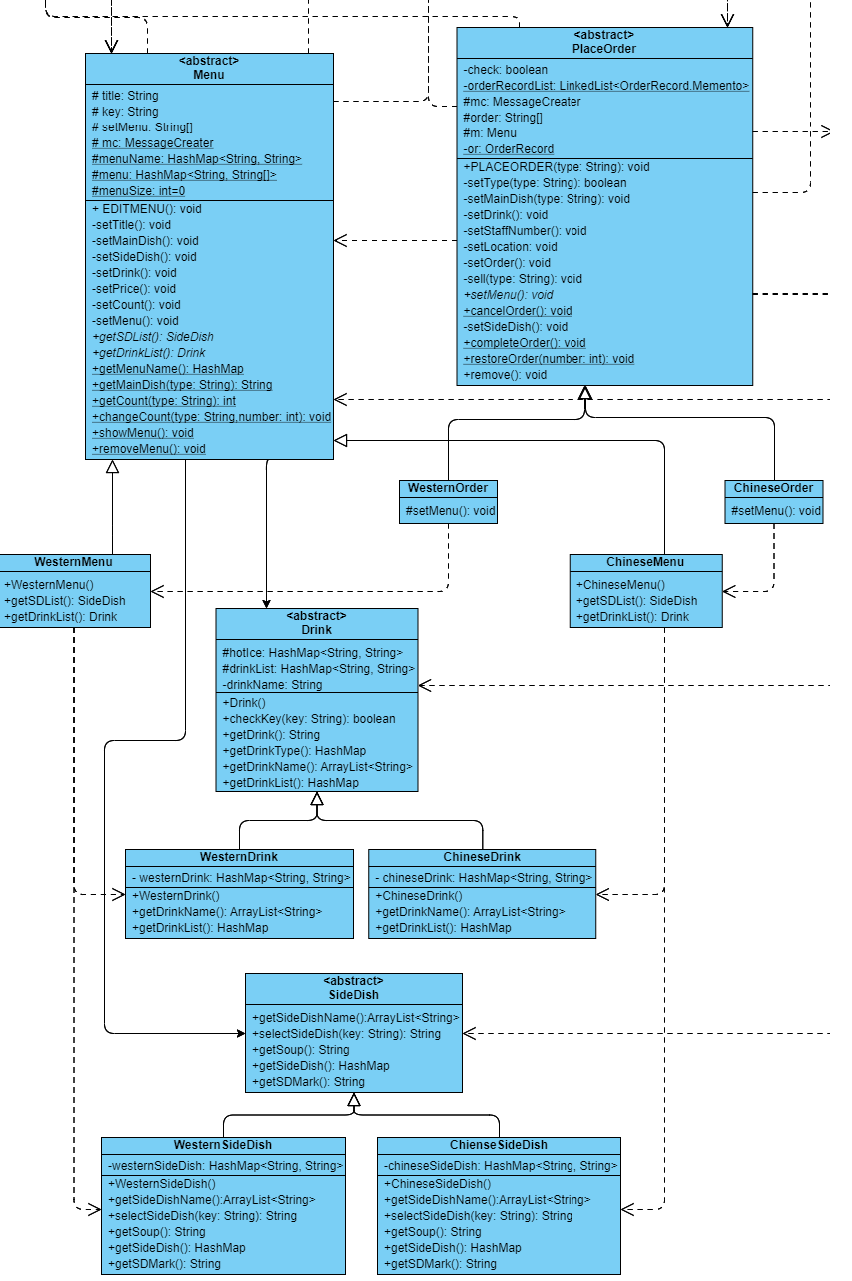
1. Command Pattern



During this assignment, this system provided six functions. I decide to use the command pattern to provide those functions. As illustrated, decouples the classes that invoke the operation from the object that knows how to execute the operation. invoker knows how to execute a Command but doesn’t know the concrete Command object and receiver class. Command provides an interface for executing an operation concrete Command call different receiver. Each character only has one reason to change. The benefit of this design pattern is to add extra command will be simple. For example, add a new function that shows how many orders were finished, create a class(concrete) implements Command interface, and set the relevant receiver after that add a new object that references the new concrete Command on client than finish. No need to change any existing class (without Client). Also, this design support undoes, all Command input from the user will be store at invoker. When the user input a command that calls invoker to undo method, invoker will find the last command input form user and use that object to call the specific concrete Command to unexecute method(Some command do not support unexecute such as show menu). For example, If user input show menu commands after edit menu, when user input a command to undo, System will check that the latest command input by the user is undoable or not(that the reason why unexecute method will return a boolean) if no system will keep trying next command object to unexecute automatically. So that in this case System will undo the edit menu.

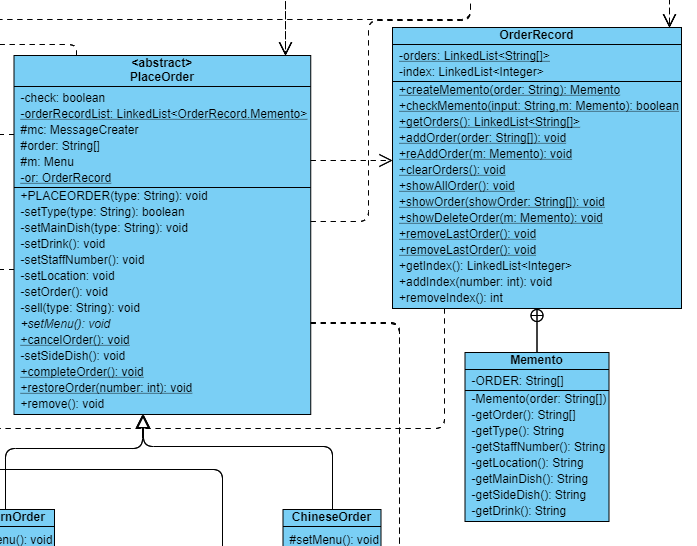
1. Abstract Factory Pattern



First I using Factory Pattern is used to create different command objects. provides an interface for creating families of related or dependent objects and without specifying their concrete classes. That avoids the Client reaches the concreate command class directly. In this picture, the Client knows the concreate factory object but they do not know what command will be returned and what will happen.

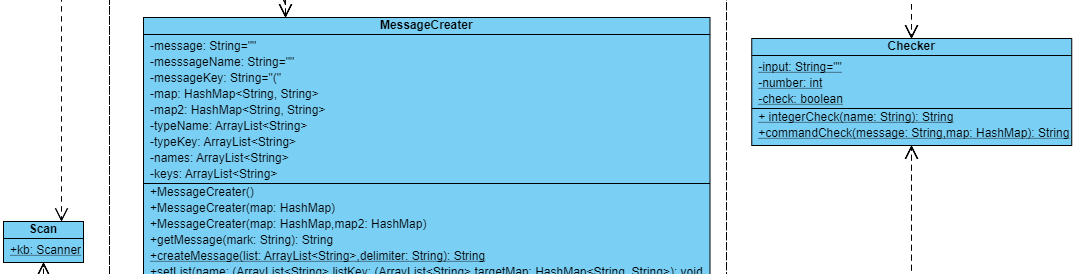
The second is using create different menu. The Abstract Menu defines the template process to Create a menu. The concrete Menu suppose implement all method define an Abstract Menu but some method process is the same on this assignment so I implement some of the method only(same as Order). As we see, there are two types of menu. I create a different type of material group. Each type of material has an abstract material class, the concrete material class complements the method defined by the abstract material class. The benefit of this design pattern is Client will understand easily. when the client wants to add an extra menu, create a concrete class one of each abstract class than finish. Moreover, assume that users do not know how to code those methods, they can look at other concrete classes for reference cause all concrete classes under the same abstract class must similar. Reduce time on thinking how to do.

1. Memento Pattern



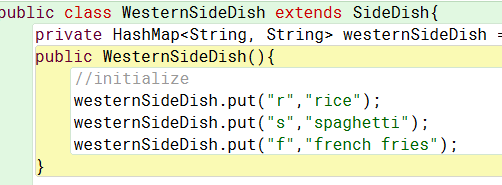
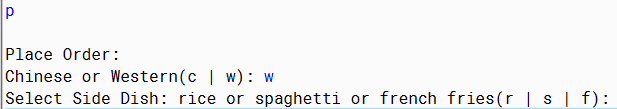
Using Memento Pattern to support the “Cancel order” function. Class Memento is an inner class inside Class OrderRecord. OrderRecord(Originator) has an internal object order. PlaceOrder(Caretaker) can request a memento from OrderRecord to save the order. Memento is opaque to the PlaceOrder. When method cancelOrder is called, first will clear the variable orders by calling the method inside OrderRecord. Then pass the memento and the specific staff number back to OrderRecord to restore the order one by one. When the staff number as same as the staff number inside the memento, OrderRecord will not restore this record and add this memento index in a linkedlist to remember that this memento is unavailable and ready for undo. The entire process PlaceOrder can not get anything from memento directly.

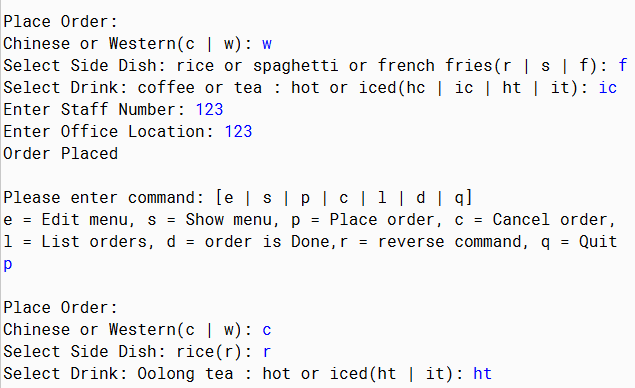
1. Other System Design



The Class Checker is for checking the input is an integer or not and check did the input has relevant command. This Class reduce most of the exception and make sure the system will not crash during the input phase

Class MessageCreater will generate the Message for request user input relevant input.

For example, when system administrators add a new side dish, they are just required to define the name and the short key. But System will show a complete sentence to the user to confirm that the user must understand what they should input. The advantage of this design is not only to avoid system administrators define similar things much time but also to confirm the system unity that will not have many types of message. Avoid User confusion.

One more special design I did is change the input request. The assignment shows that there are no necessary to require user select side dish and drink type when they order Chinese order. But for the system unity and expansibility, I decide to keep request that information. Because this design can be extended more easily. For example, if I following the way on assignment, I must change the code on the Chinese menu. (not request 🡪 request).

**User Guide**

\*All input should be the key of command (small letter),

not the name of command (Except specified request).

System Command: Step 1

Edit Menu: Step 2 – Step 6

Show Menu: Step 7

Place Order: Step 8 – Step 10

List Orders: Step 11

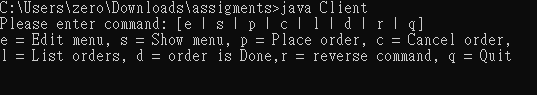
Order is Done: Step 12

Reverse command: Step 13

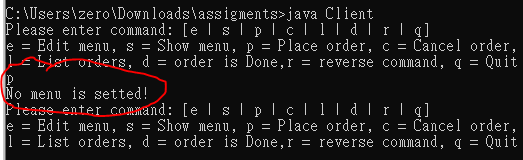
Cancel Order: Step 14 – Step 15

Quit: Step 16

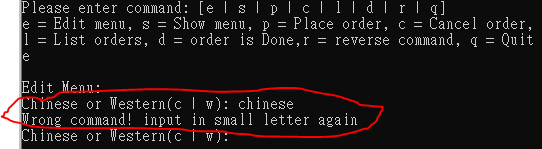
1. When system start, there are 8 command that can choose by user.



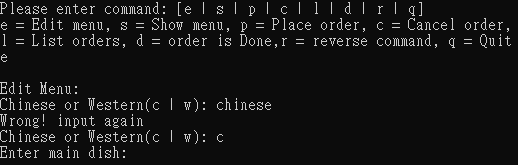
1. User must Edit menu at the begin, otherwise other command can’t work normally. For example, input “p” to try place order before edit menu will show the error message.



1. User input ”e” from their keyboard, edit menu function will start operation, first, it will show all type of menu on this system, then request choose one of it by input the relevant key. Input the name of menu is not allows.

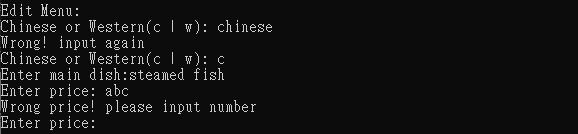


1. After input correctly, system will request the name of main dish of relevant menu

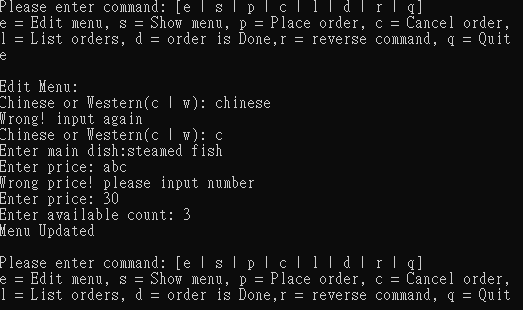


\* step 4 to step 6 is same whether user input “c” or “w’ on select type of menu on step 3

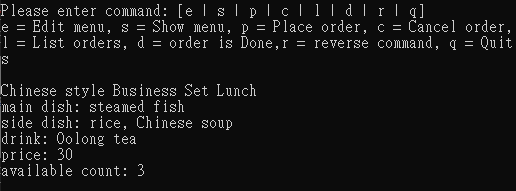
1. System will request the price of menu after input correctly. User can’t input non-numeric value, error message will be showed.



1. Similarly, can’t input non-numeric value when system request user input the available count. After available count input correctly, system will show the menu was updated.

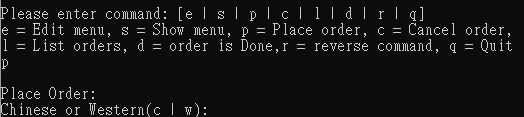


1. When User edit menu already, input ”s” to show the menu detail to check the information is correctly or not. Here will show the menu name, main dish, side dish, drink, price and available count.



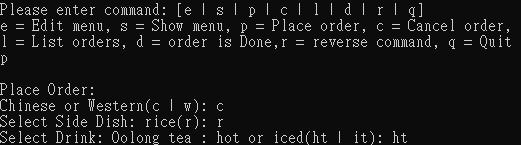
\*User can edit menu again if there are something incorrect(back to step 3)

1. User can start place order if menu information is correct. Input ”p” to operate place order function. As same as edit menu, User need to select the type of menu first.

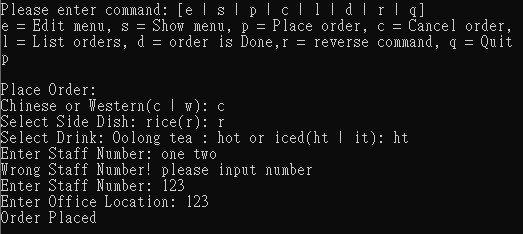


\* step 9 to step 10 is similar whether user input “c” or “w’ on select type of menu on step 8

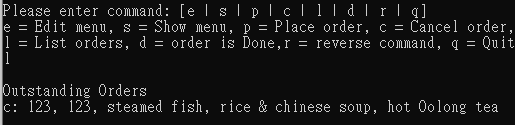
1. Then System will request user select main dish and drink sequentially. Each drink has two type which is ice and hot, so each drink has two selection, user required to input the key of type before the key of drink. (at same input)



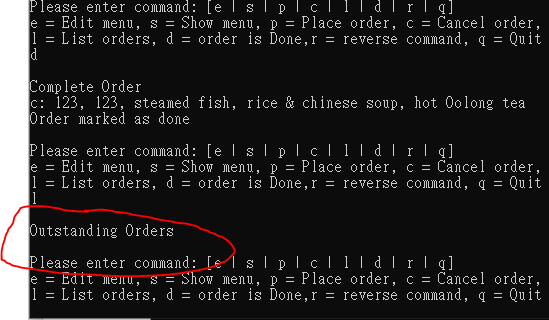
1. User requested to input the staff number and office location after they decided their order. Both staff number and office location such input numeric value only. System will show “order placed” when user complete all the place order function procedure.



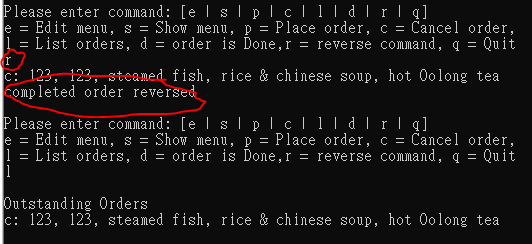
1. Now user can input ”l” to view what order was placed.



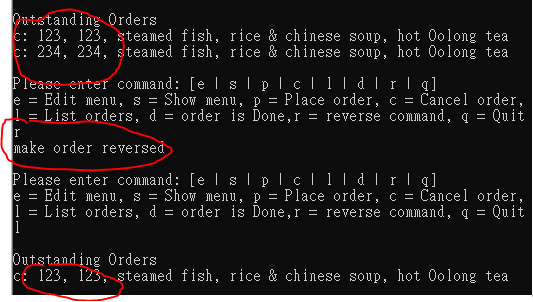
1. When lunch set was delivered to customer, user can input “d” to mark that order was completed. System will not show this order on placed order list anymore.



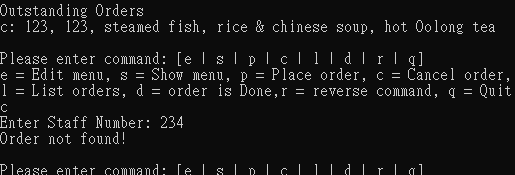
1. In case of user wrong input, system provide undo function to handle this situation. Input “r” to start this function. On this example, undo function will recover the order to placed order list.



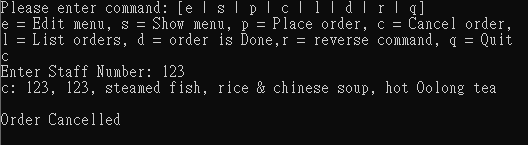
undo function supports all command, Doesn't matter what the command is.



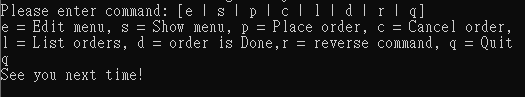
1. User can cancel order by input the staff number. Input “c’ to start operation. Then system will request a staff number, If staff number do not exist on placed order list, system will show error message.



1. If user input correctly, system will show ”order canceled”.



1. Input “q” to exit system when business hour is over.



**Test Plan and Test Cases**

Scenario:

staff canteen will sell two type of menu today.

Chinese menu main dish is steamed fish, price is 30 and available count is 3. Western menu main dish is steak, price is 40 and available count is 3.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test cases no. | Goal | Input | Excepted result | True result |
| 1 | Edit Chinese Menu | **e** | Request type of menu | Consistent |
|  |  | **c** | Request main dish | Consistent |
|  |  | **Steamed fish** | Request price | Consistent |
|  |  | **thirty** | Request available count | Error message show, require input number. |
|  |  | **30** | Request available count | Consistent |
|  |  | **3** | Show message  ”Menu updated” | Show message  ”Menu updated” |
| 2 | Edit Western Menu | **e** | Request type of menu | Consistent |
|  |  | **w** | Request main dish | Consistent |
|  |  | **Steak** | Request price | Consistent |
|  |  | **40** | Request available count | Consistent |
|  |  | **3** | Show message  ”Menu updated” | Consistent |
| 3 | View menu detail | **s** | Show two menu detail | Consistent |

In morning, There are two customer order two Chinese lunch set, one order western menu. After that, two more customer order Chinese lunch set.

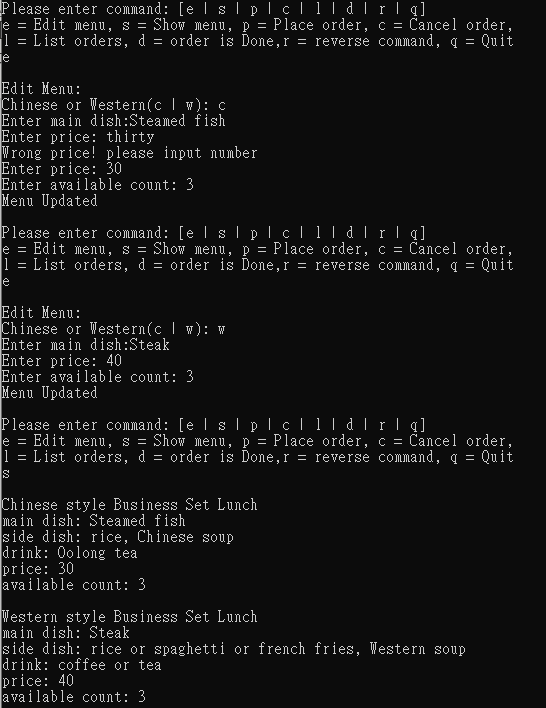
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test cases no. | Goal | Input | Excepted result | True result |
| 4 | Order Chinese lunch set | p | Request type of menu | Consistent |
|  |  | c | Request select a side dish | Consistent |
|  |  | R | Request drink | Error message show, require input small letter. |
|  |  | r | Request drink | Consistent |
|  |  | t | Request staff number | Error message show, require input again. |
|  |  | ht | Request staff number | Consistent |
|  |  | asd | Request location | Error message show, require input number. |
|  |  | 123 | Request location | Consistent |
|  |  | 123 | Show message  ”order placed” | Show message  ”order placed” |
| 5 | Order Chinese lunch set | p | Request type of menu | Consistent |
|  |  | c | Request select a side dish | Consistent |
|  |  | r | Request drink | Consistent |
|  |  | it | Request staff number | Consistent |
|  |  | 234 | Request location | Consistent |
|  |  | 234 | Show message  ”order placed” | Consistent |
| 6 | Order Western lunch set | p | Request type of menu | Consistent |
|  |  | w | Request select a side dish | Consistent |
|  |  | s | Request drink | Consistent |
|  |  | hc | Request staff number | Consistent |
|  |  | 987 | Request location | Consistent |
|  |  | 987 | Show message  ”order placed” | Consistent |
| 7 | View placed order list | l | Show all placed order with detail | Consistent |
| 8 | View available count | s | Chinese menu count become 1 and western menu count become 2 | Consistent |
| 9 | Order Chinese lunch set | p | Request type of menu | Consistent |
|  |  | c | Request select a side dish | Consistent |
|  |  | r | Request drink | Consistent |
|  |  | ht | Request staff number | Consistent |
|  |  | 345 | Request location | Consistent |
|  |  | 345 | Show message  ”order placed” | Consistent |
| 10 | View available count | s | Chinese menu count become 0 | Consistent |
| 11 | Order Chinese lunch set | p | Request type of menu | Consistent |
|  |  | c | Request select a side dish | Show message ”sold out” and back to main command selection |

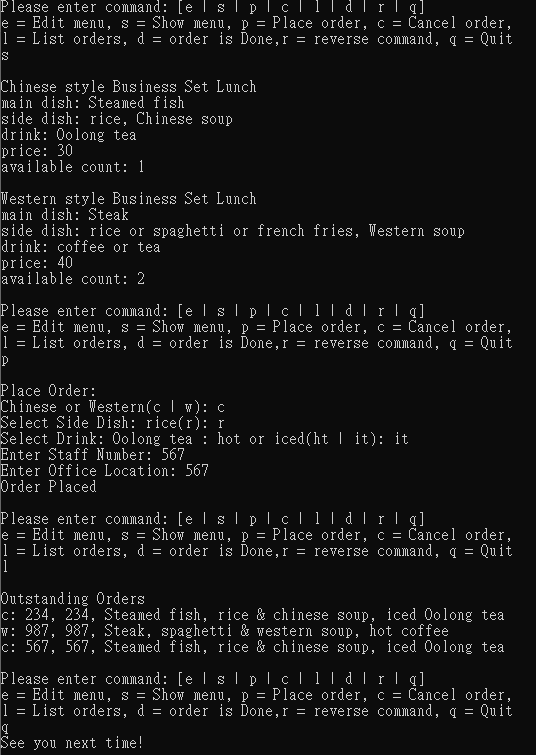
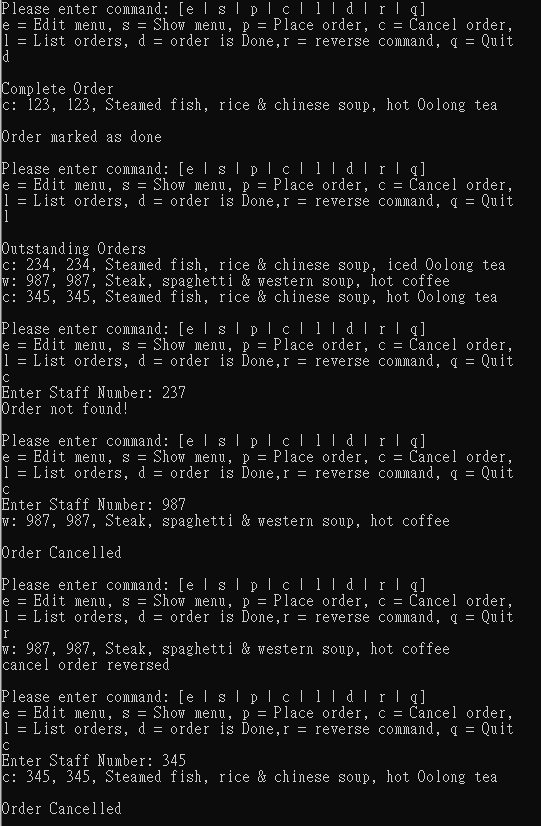
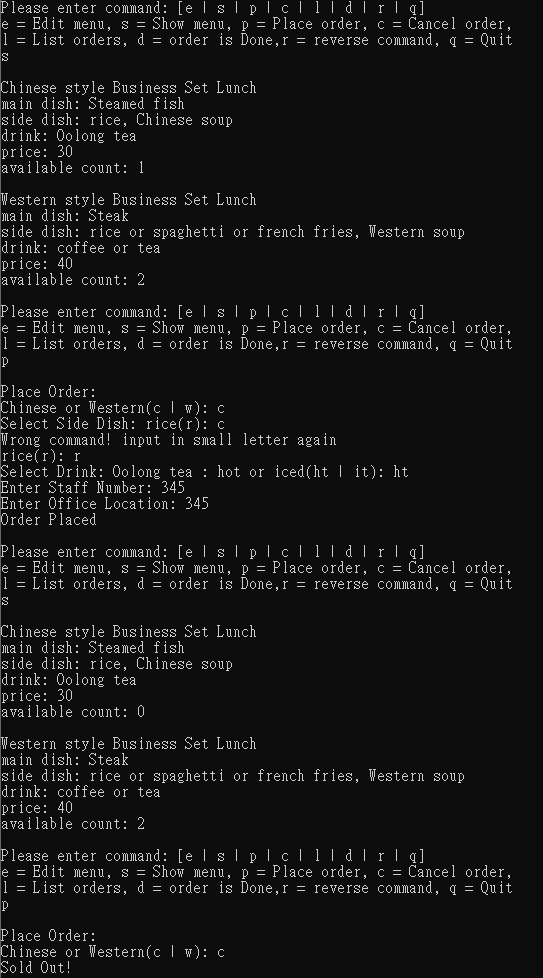
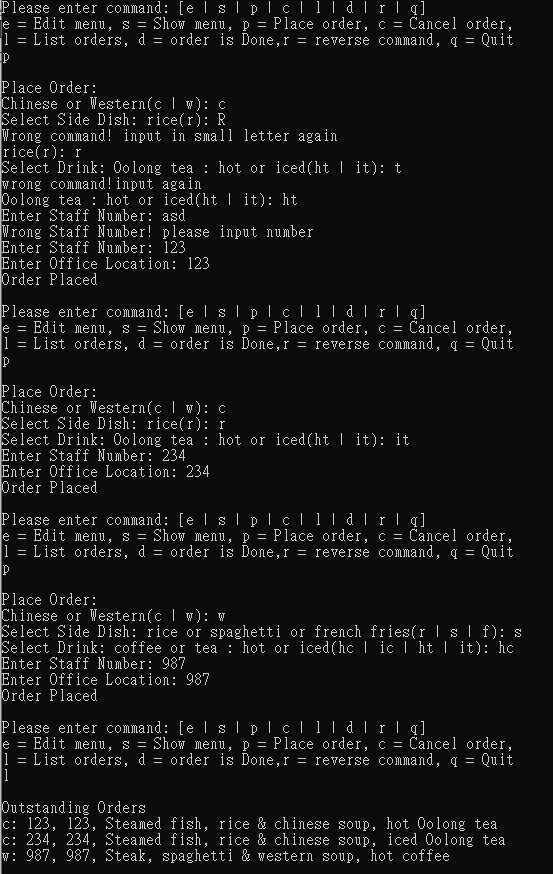
First order of the list was finished, and cancel the staff number 345 order. Then place Chinese lunch set to staff 567.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test cases no. | Goal | Input | Excepted result | True result |
| 12 | Mark first order as finish | d | Show which order was completed | Consistent |
| 13 | View placed order list | l | The order show above was disappeared | Consistent |
| 14 | Cancel order with staff number 345 | c | Request staff number | Consistent |
|  |  | 237 | Show which order was canceled | Show message  ”order not found” |
| 15 | Cancel order with staff number 345 | c | Request staff number | Consistent |
|  |  | 987 | Show which order was canceled | Consistent |
| 16 | Reverse the cancel order action | r | Show which order restored with message  ”cancel order reversed” | Consistent |
| 17 | Cancel order with staff number 345 | c | Request staff number | Consistent |
|  |  | 345 | Show which order was canceled | Consistent |
| 18 | View available count | s | Chinese menu count become 1 | Consistent |
| 19 | Order Chinese lunch set | p | Request type of menu | Consistent |
|  |  | c | Request select a side dish | Consistent |
|  |  | r | Request drink | Consistent |
|  |  | it | Request staff number | Consistent |
|  |  | 567 | Request location | Consistent |
|  |  | 567 | Show message  ”order placed” | Consistent |
| 20 | View Placed order list | l | Staff number should be 234,987,567 Sequentially | Consistent |

Close System when business time is over.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test cases no. | Goal | Input | Excepted result | True result |
| 21 | Exit system | q | Exit system with message  ”see you next time” | Consistent |





**Well documented Source Code**

Please refer to file in folder "source code".