Xi'an Jiaotong-Liverpool University
西交利物浦大学

Department of Electrical and Electronic Engineering

EEE102 C++ Programming and Software Engineering

Group Project D Restaurant Management System [1]
SDP Report

Group ID: G19

Group Leader[2]: Zheng Sun

ID Number: 1507820

We certify that we have read and understood the University's *Policy for dealing with Plagiarism, Collusion and the Fabrication of Data* with reference to this policy, we certify that:

- Our work does not contain any instances of plagiarism and/or collusion.
- Our work does not contain any fabricated data.

By handing in our assignment for marking, WE formally declare that all the above information is true to the best of our knowledge and belief.

Signature: Zheng Sun, Zheng Huang, Liming Zong, Zecheng Li, Shengyi Yang, Yuze Hu

---

[1] Last updated on May, 29th, 2017
[2] Group Leader should submit the related material of the group project

Contribution form

–On the second page of your report

–Signed by all the group members

–Breakdown of the roles and responsibilities of each group member during the duration of the project.

–The item of "How did the student appreciate the work within the group" could be your reflection or comments about your work during the group (no more than 70 words).

Students are requested to rate sincerely their contributions (1=low,5=High) as well as the percentage of their work load (%) during the project development process. Note that summation of all the students' work load for each specific item, e.g., coding, should be 100%.							
		Spec	**Design**	**Coding**	**Testing**	**Doc.**	**How did the student appreciate the work within the group?**
Zheng Sun 1507820	**Rate**	**4**	**5**	**5**	**4**	**5**	**Honoured to work with you guys. Excellent team members.**
	WorkLoad	**20%**	**25%**	**41%**	**25%**	**25%**	
Zheng Huang 1508070	**Rate**	**4**	**5**	**4**	**5**	**5**	**Gain a lot from this project and have nice corporations with team mates.**
	WorkLoad	**20%**	**25%**	**18%**	**45%**	**25%**	
Liming Zong 1508092	**Rate**	**3**	**5**	**3**	**3**	**4**	**I have enjoyed collaborating with my group members and be happy to design this program.**
	WorkLoad	**10%**	**16%**	**13%**	**10%**	**16%**	
Zechen Li 1507645	**Rate**	**3**	**5**	**3**	**3**	**4**	**The process of design is wonderful, I really enjoy it.**
	WorkLoad	**10%**	**16%**	**9%**	**10%**	**12%**	
Shengyi Yang 1508085	**Rate**	**3**	**4**	**3**	**3**	**4**	**In this project, all of the team members work hard and have a cordial working relationship. Additionally, I also learn some useful skills of programming form my excellent teammates. After all, I am very grateful to do this project with these wonderful team members. I reckon this project is vitally beneficial for my future studying.**
	WorkLoad	**10%**	**12%**	**9%**	**5%**	**12%**	
Yunze Hu 1508069	**Rate**	**5**	**3**	**3**	**3**	**4**	**From the project, I have a good experience about c＋＋ and I understand the meaning of cooperation.I feel lucky to have such a good team and feel proud to complete the project perfectly.**
	WorkLoad	**30%**	**10%**	**6%**	**5%**	**10%**	

1. Problem Statement Specification

1.1. Overall Specification

Our team is employed by a restaurant to create a database management system (DBMS) to improve the management of a restaurant. This report is written to shoe the design analysis and test of implementation (coding) of the DBMS, as well as instruct user to manage the DBMS.

1.2. Customer Specification

The restaurant has an *inventory storing* ***raw materials*** and *a menu listing all the* ***dishes***. At the same time, the restaurant implement a membership system which keeps the *detailed information of the* ***customers*** and a ***consumption record list*** *which keeps every piece of dish consumption along with* ***customer info, dish info*** *and* ***date****.* Additionally, the system should be able to calculate the fee that the customer should pay according to his/her ordered dishes. The gross profit should also be calculated based on the total avenue from the dishes minus the total expenditure of raw materials.

1.3. System specification

The DBMS should provide following functions as shown below:

a) All the detail information of ***raw material in inventory***, ***dishes on menu*** and ***customer in file*** should be able to be added, browsed, modified and deleted.

b) Three ***different user interfaces (UI)*** should be presented with different authority, namely, manager, chef, customer. The manager user of the system can edit all the information of ***customer file, dish list and raw material inventory***. The chef users can browse and edit the raw material info in inventory and the dish in menu list but they cannot search or edit the customer information. The customer users can only browse the menu, place the dish orders and see his/ her bill.

c) When the customers add the dishes to the menu, the order of a same dish which is already order currently should be verified to place. When a new dish is added to the menu, the name of the dish cannot be the same as the old one.

2. Analysis

a) Inputs:

i. General

1. String types for login: account ID, PIN etc. →getline(cin, ID/PIN/etc);

2. Integer/string/char types for selecting options: temp → int intParser(),string strParser(), char intParser()

ii. Manger

1. String types for searching all the data: accountID/dishID/rawMaterialID → getline(cin,ID);

2. String types for revising all the data: temp → string strParser(),int intParser()

iii. Chef

1. String/integer types for revising the dishes and materials: temp → string strParser(),int intParser()

2. String types for searching the data of dishes and materials: dishID/rawMaterialID→ getline(cin,ID)

iv. Customer

1. string types for inputting the ID of a dish : temp → string strParser()

b) Output:

1. Interactive instructions prompt user to input ID, name etc.

2. Interfaces which vary from the identity of user

3. Necessary information display, e.g. when the user plans to order, the menu list should be automatically shown.

C) Extra and bonus characteristics

1. To store data, the vector has an overwhelming advantages over the structure arrays, because it has a dynamic growth length.

2. Managers have an ability of viewing all historical consumption records of a customer.

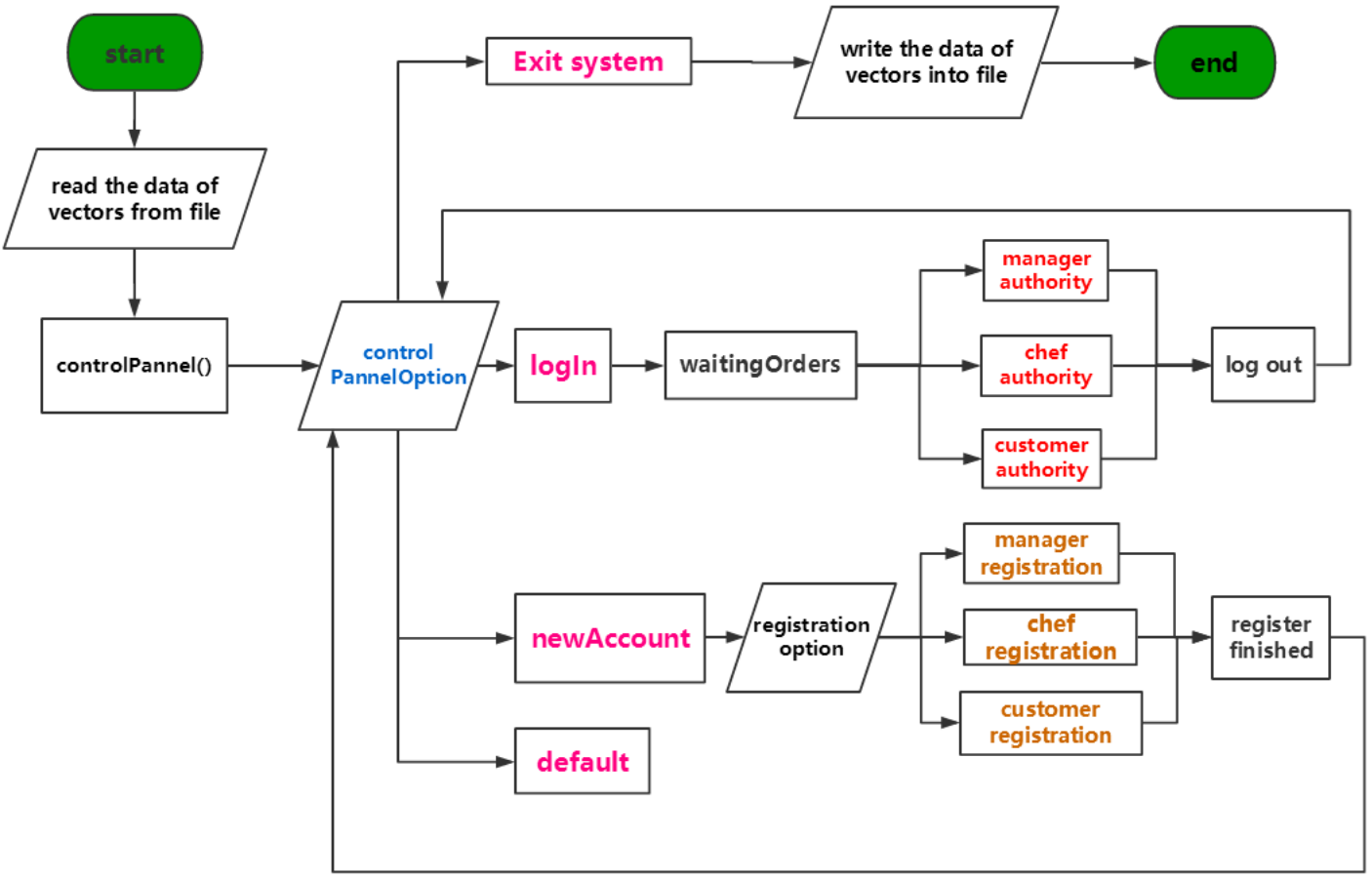
3. Managers are able to add dishes for every historical consumption.

4. The information is quite real-life, e.g. it concludes the date of an order, the order of a historical consumption etc.

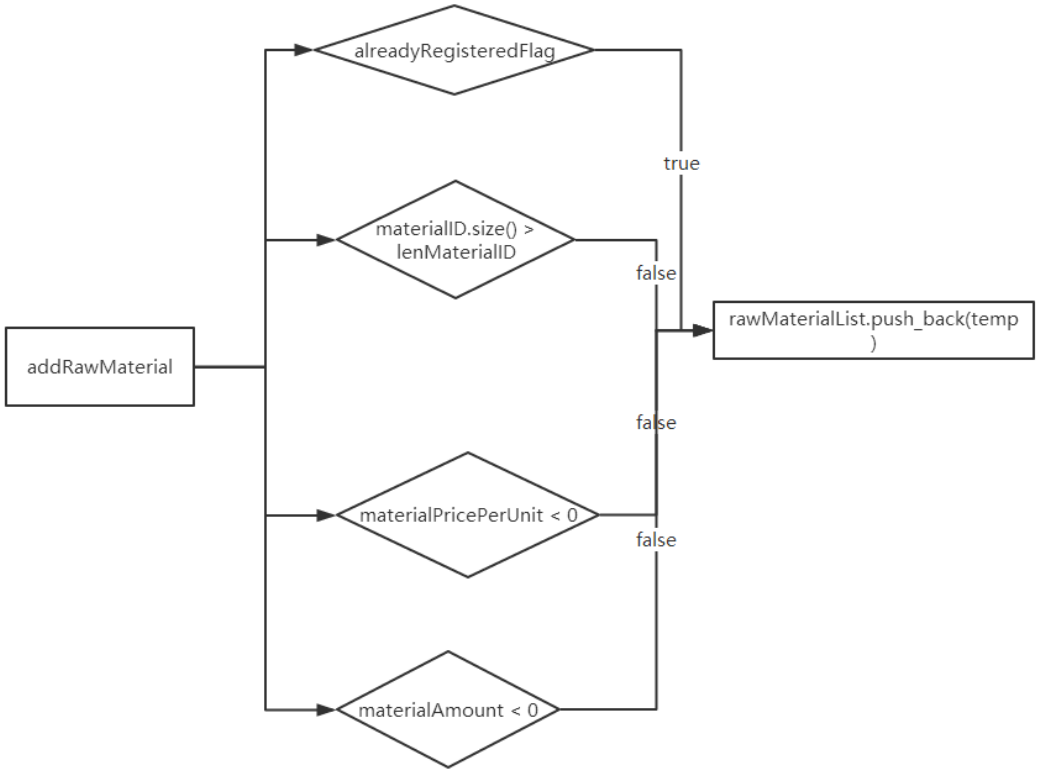
3. Design

3.1. Interface and main() function design

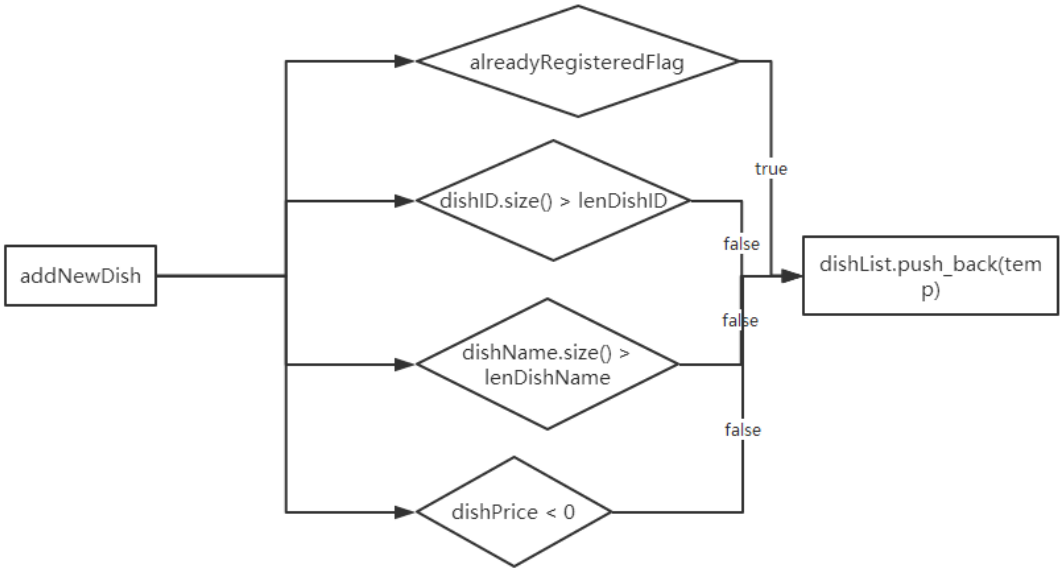
3.1.1 main():



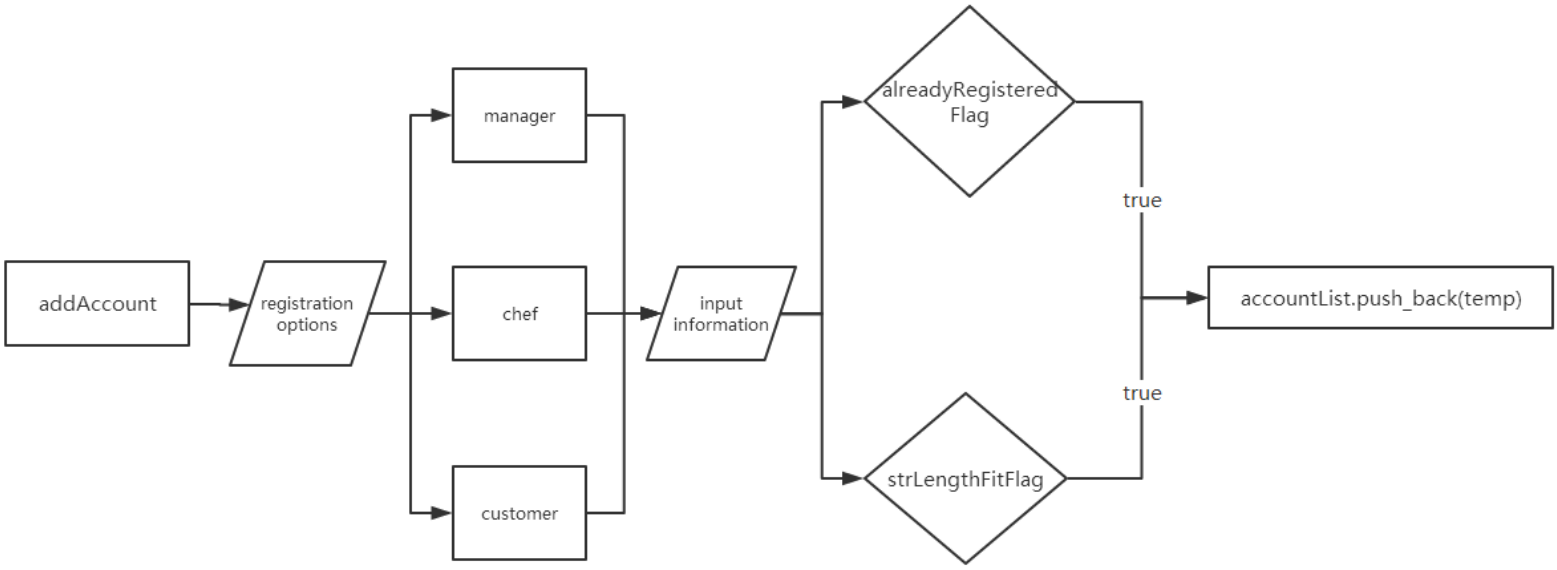
3.1.2. Add Raw Material:



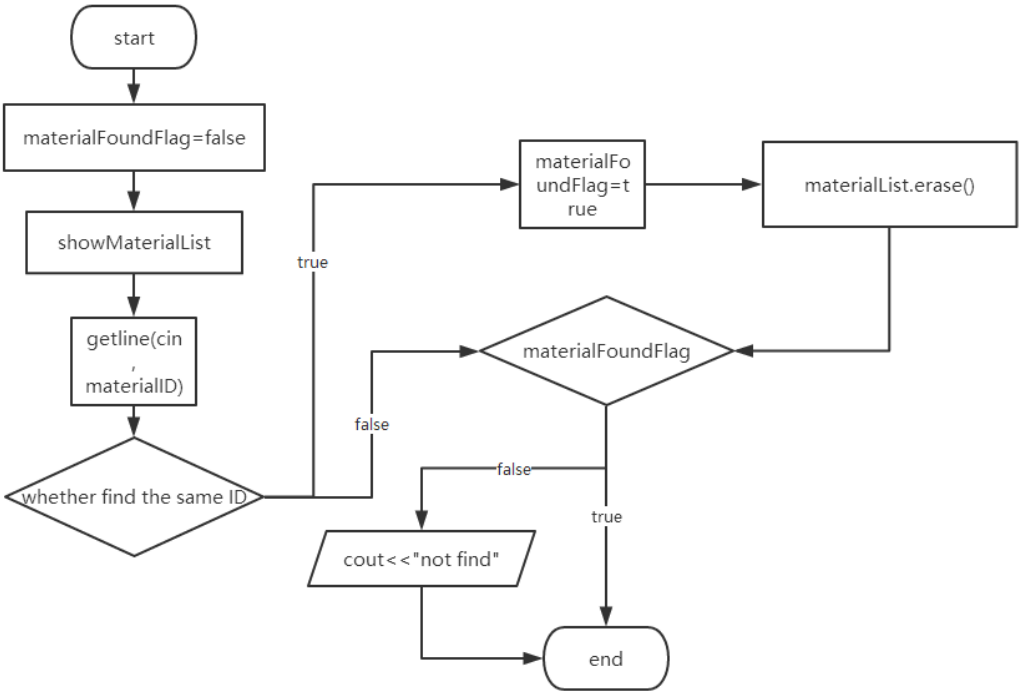
3.1.3. Add dish:



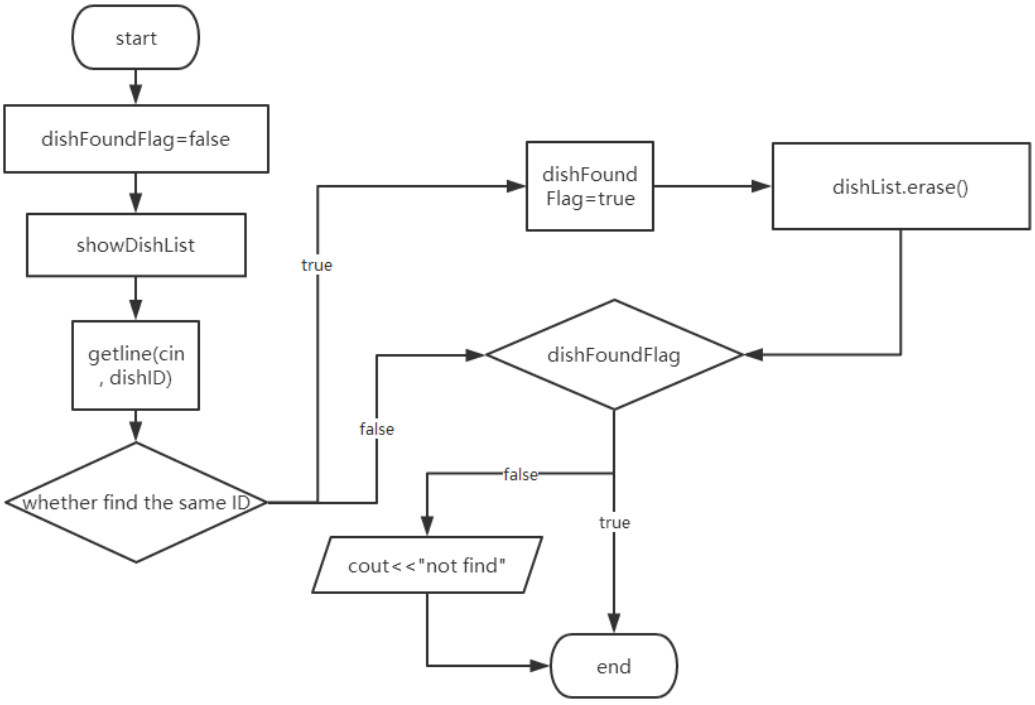
3.1.4. Add account:



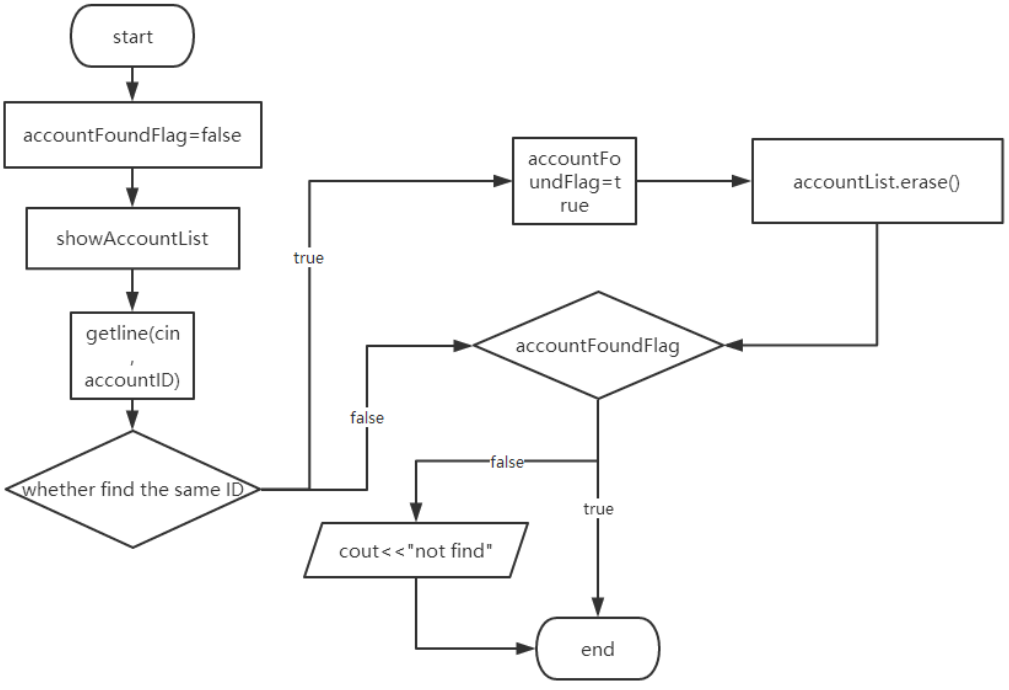
3.1.5. Delete material:



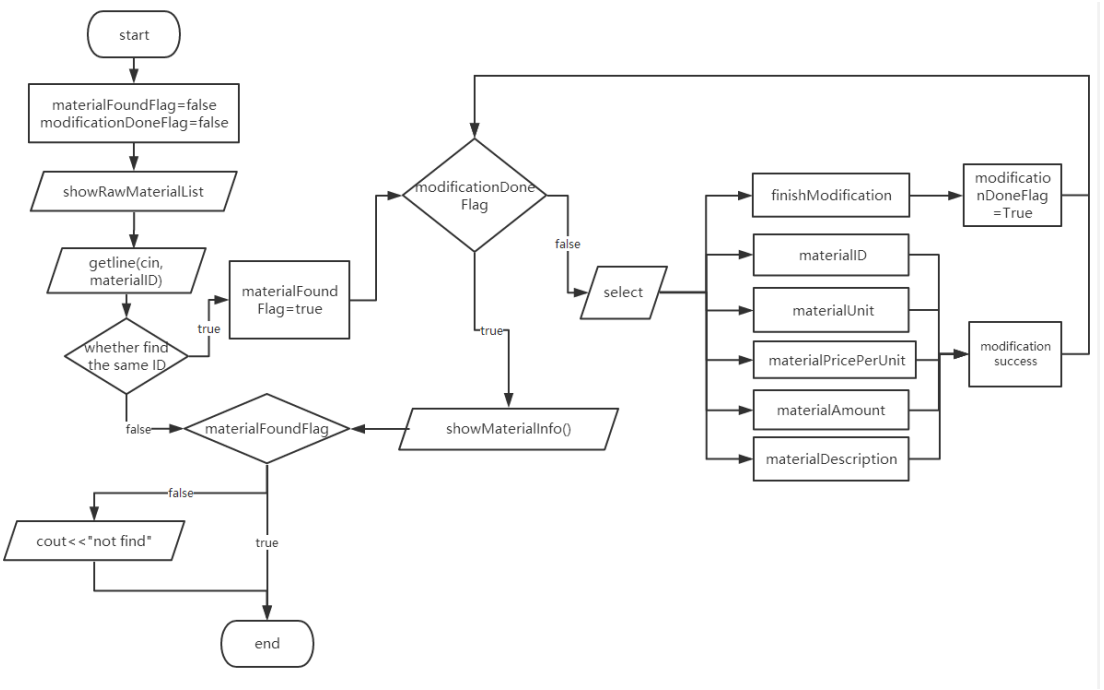
3.1.6. Delete dish:



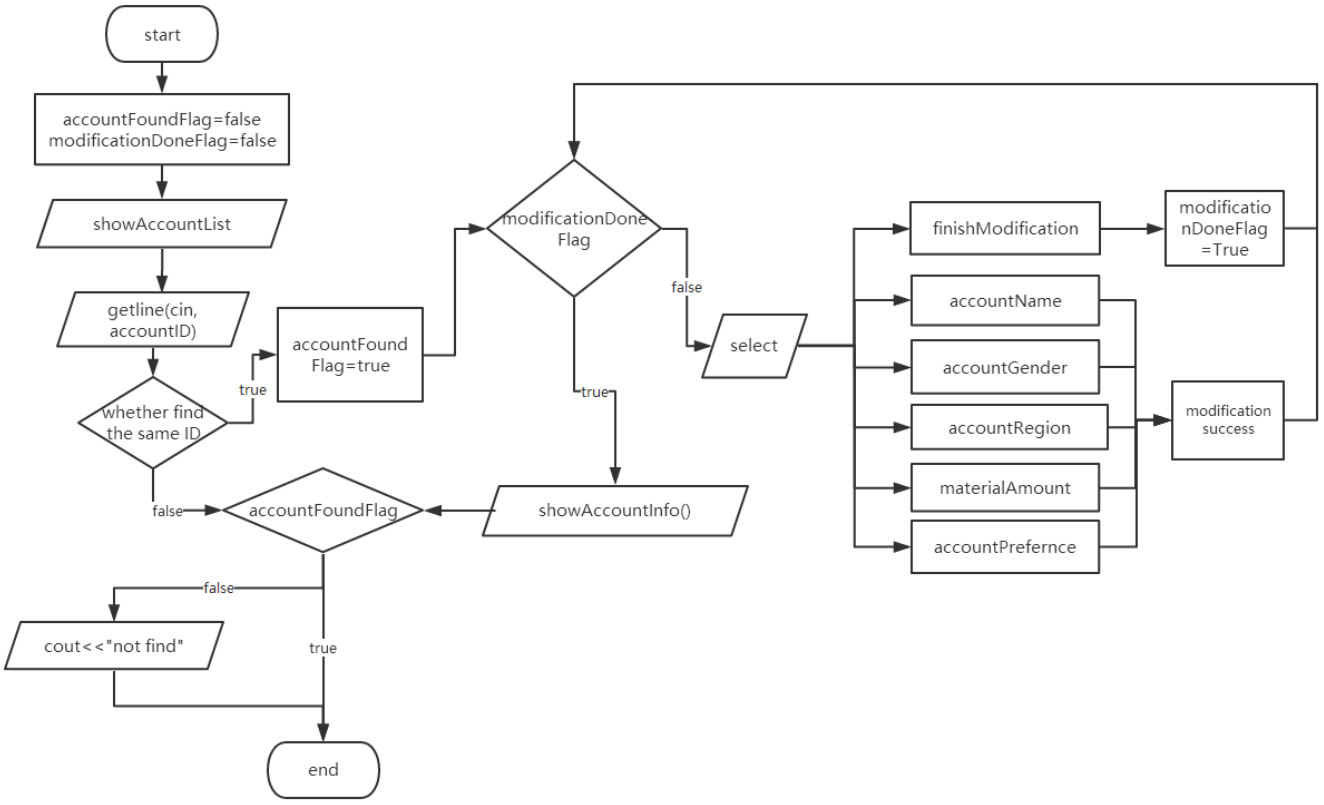
3.1.7. Delete account:



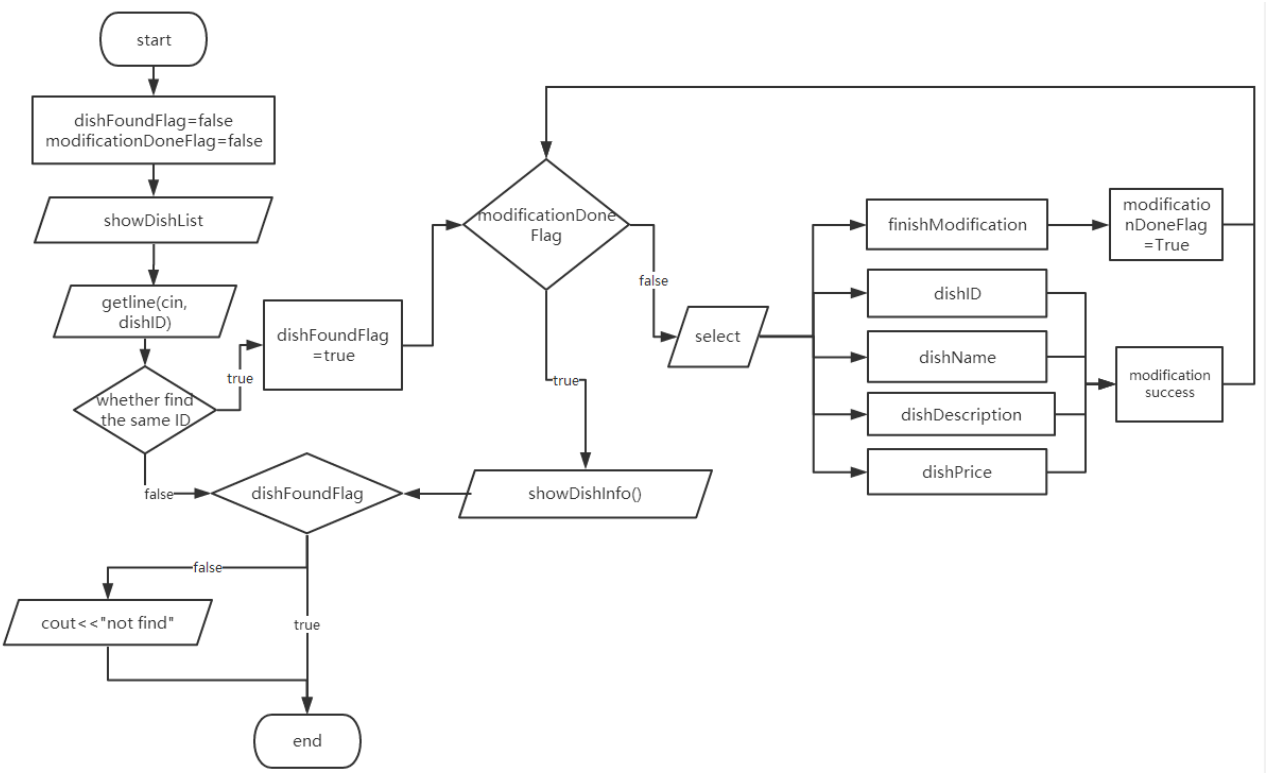
3.1.8. Modify material:



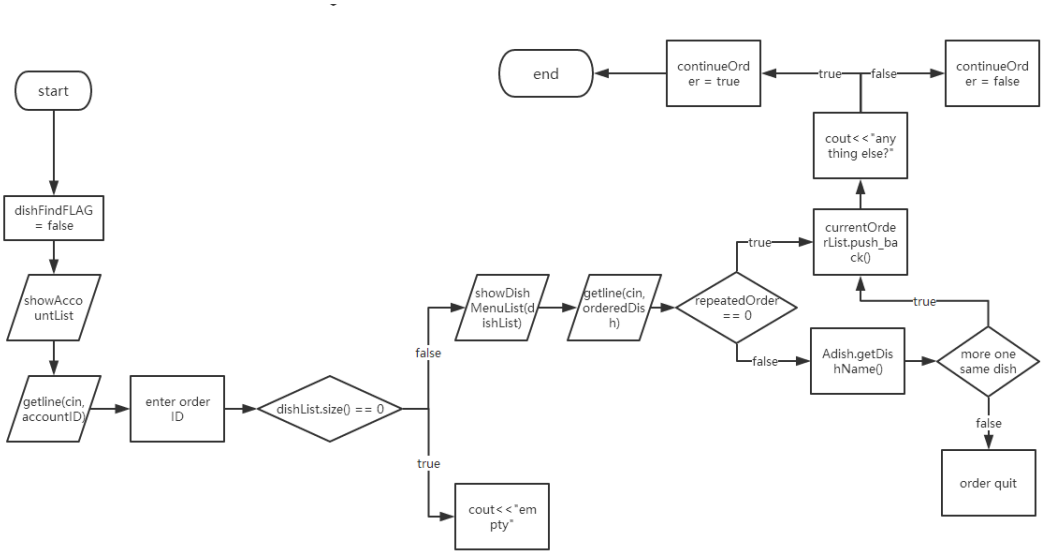
3.1.9. Modify account:



3.1.10. Modify dish:



3.1.11. Place Order



b) Class Design: 1.CRC cards and 2.hierarchical UML class graph

1. CRC cards of classes

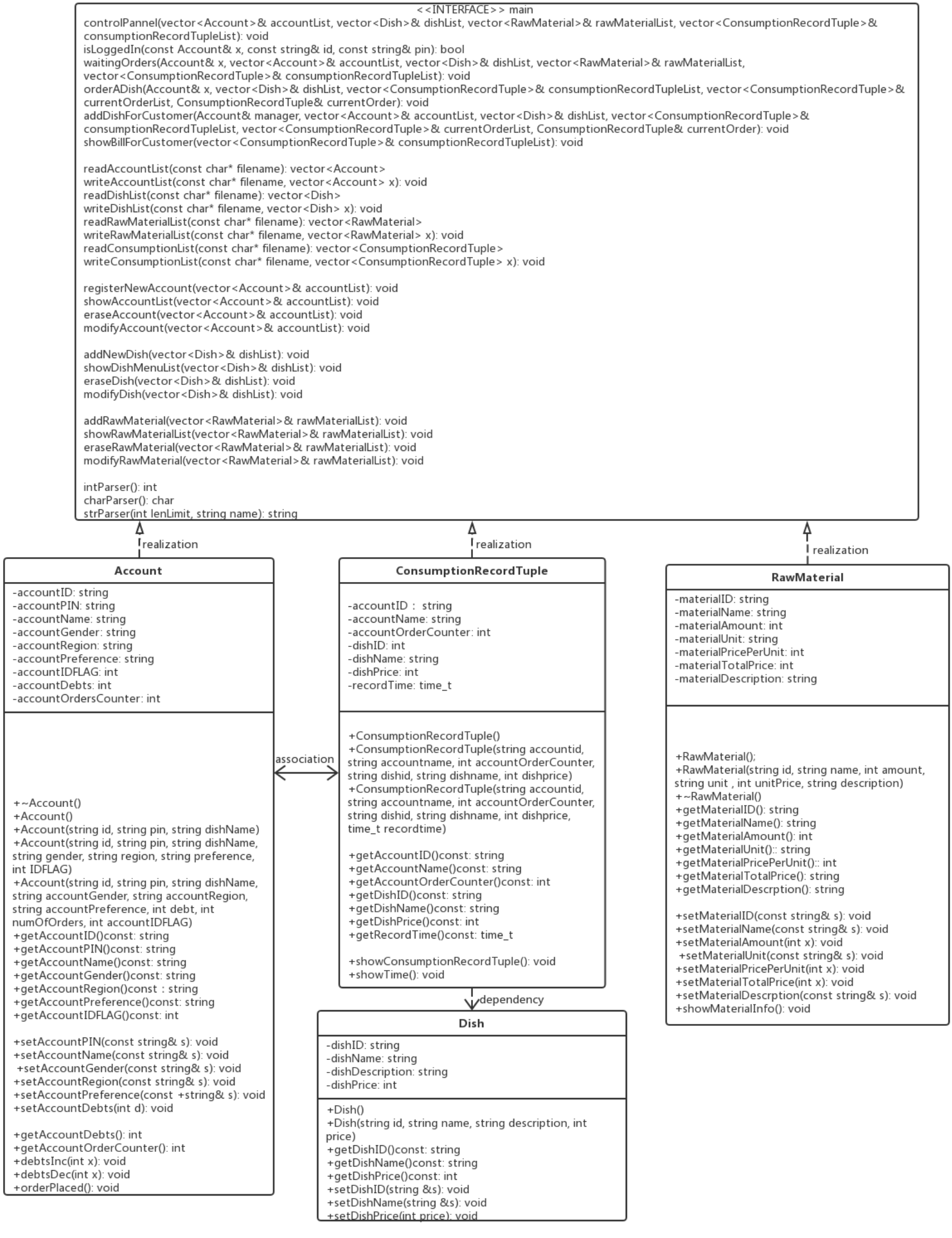
Class: Account	
Responsibility: Storing the information of all users. Record the ID, user's name, password(PIN), gender, religion and preference taste. Collaborate with Consumption Record Tuple to create a record of an order by providing account info.	**Collaborators:** ConsumptionRecordTuple

Class: Dish	
Responsibility: Holding the information of dishes. Serve as a basic element to be read from and write into. Provide functions to set and change the dish data. Collaborate with Consumption Record Tuple to create a record of an order by providing dish info.	**Collaborators:** ConsumptionRecordTuple

Class: RawMaterial	
Responsibility: Basic element in this program, holding the amount, type and price information of raw materials. The restaurant should be functioning with raw materials and be edited except for customer account. Collaborate with main interface to calculate the total expenditure and gross profit.	**Collaborators:** Main() <<interface>>

Class: ConsumptionRecordTuple	
Responsibility: Hold information for every customers' every dish order which contains the account info, dish info and time of order. Collaborate with main interface to calculate the total expenditure and gross profit, with Account and Dish to create order record.	**Collaborators:** Main() <<interface>> Account Dish

2.Hierarchical UML class graph


<<INTERFACE>> main
controlPannel(vector<Account>& accountList, vector<Dish>& dishList, vector<RawMaterial>& rawMaterialList, vector<ConsumptionRecordTuple>& consumptionRecordTupleList): void
isLoggedIn(const Account& x, const string& id, const string& pin): bool
waitingOrders(Account& x, vector<Account>& accountList, vector<Dish>& dishList, vector<RawMaterial>& rawMaterialList, vector<ConsumptionRecordTuple>& consumptionRecordTupleList): void
orderADish(Account& x, vector<Dish>& dishList, vector<ConsumptionRecordTuple>& consumptionRecordTupleList, vector<ConsumptionRecordTuple>& currentOrderList, ConsumptionRecordTuple& currentOrder): void
addDishForCustomer(Account& manager, vector<Account>& accountList, vector<Dish>& dishList, vector<ConsumptionRecordTuple>& consumptionRecordTupleList, vector<ConsumptionRecordTuple>& currentOrderList, ConsumptionRecordTuple& currentOrder): void
showBillForCustomer(vector<ConsumptionRecordTuple>& consumptionRecordTupleList): void
readAccountList(const char* filename): vector<Account>
writeAccountList(const char* filename, vector<Account> x): void
readDishList(const char* filename): vector<Dish>
writeDishList(const char* filename, vector<Dish> x): void
readRawMaterialList(const char* filename): vector<RawMaterial>
writeRawMaterialList(const char* filename, vector<RawMaterial> x): void
readConsumptionList(const char* filename): vector<ConsumptionRecordTuple>
writeConsumptionList(const char* filename, vector<ConsumptionRecordTuple> x): void
registerNewAccount(vector<Account>& accountList): void
showAccountList(vector<Account>& accountList): void
eraseAccount(vector<Account>& accountList): void
modifyAccount(vector<Account>& accountList): void
addNewDish(vector<Dish>& dishList): void
showDishMenuList(vector<Dish>& dishList): void
eraseDish(vector<Dish>& dishList): void
modifyDish(vector<Dish>& dishList): void
addRawMaterial(vector<RawMaterial>& rawMaterialList): void
showRawMaterialList(vector<RawMaterial>& rawMaterialList): void
eraseRawMaterial(vector<RawMaterial>& rawMaterialList): void
modifyRawMaterial(vector<RawMaterial>& rawMaterialList): void
intParser(): int
charParser(): char
strParser(int lenLimit, string name): string
realization
realization
realization
Account
-accountID: string
-accountPIN: string
-accountName: string
-accountGender: string
-accountRegion: string
-accountPreference: string
-accountIDFLAG: int
-accountDebts: int
-accountOrdersCounter: int
+~Account()
+Account()
+Account(string id, string pin, string dishName)
+Account(string id, string pin, string dishName, string gender, string region, string preference, int IDFLAG)
+Account(string id, string pin, string dishName, string accountGender, string accountRegion, string accountPreference, int debt, int numOfOrders, int accountIDFLAG)
+getAccountID()const: string
+getAccountPIN()const: string
+getAccountName()const: string
+getAccountGender()const: string
+getAccountRegion()const : string
+getAccountPreference()const: string
+getAccountIDFLAG()const: int
+setAccountPIN(const string& s): void
+setAccountName(const string& s): void
+setAccountGender(const string& s): void
+setAccountRegion(const string& s): void
+setAccountPreference(const +string& s): void
+setAccountDebts(int d): void
+getAccountDebts(): int
+getAccountOrderCounter(): int
+debtsInc(int x): void
+debtsDec(int x): void
+orderPlaced(): void
association
ConsumptionRecordTuple
-accountID : string
-accountName: string
-accountOrderCounter: int
-dishID: int
-dishName: string
-dishPrice: int
-recordTime: time_t
+ConsumptionRecordTuple()
+ConsumptionRecordTuple(string accountid, string accountname, int accountOrderCounter, string dishid, string dishname, int dishprice)
+ConsumptionRecordTuple(string accountid, string accountname, int accountOrderCounter, string dishid, string dishname, int dishprice, time_t recordtime)
+getAccountID()const: string
+getAccountName()const: string
+getAccountOrderCounter()const: int
+getDishID()const: string
+getDishName()const: string
+getDishPrice()const: int
+getRecordTime()const: time_t
+showConsumptionRecordTuple(): void
+showTime(): void
dependency
Dish
-dishID: string
-dishName: string
-dishDescription: string
-dishPrice: int
+Dish()
+Dish(string id, string name, string description, int price)
+getDishID()const: string
+getDishName()const: string
+getDishPrice()const: int
+setDishID(string &s): void
+setDishName(string &s): void
+setDishPrice(int price): void
RawMaterial
-materialID: string
-materialName: string
-materialAmount: int
-materialUnit: string
-materialPricePerUnit: int
-materialTotalPrice: int
-materialDescription: string
+RawMaterial();
+RawMaterial(string id, string name, int amount, string unit , int unitPrice, string description)
+~RawMaterial()
+getMaterialID(): string
+getMaterialName(): string
+getMaterialAmount(): int
+getMaterialUnit():: string
+getMaterialPricePerUnit():: int
+getMaterialTotalPrice(): string
+getMaterialDescrption(): string
+setMaterialID(const string& s): void
+setMaterialName(const string& s): void
+setMaterialAmount(int x): void
+setMaterialUnit(const string& s): void
+setMaterialPricePerUnit(int x): void
+setMaterialTotalPrice(int x): void
+setMaterialDescrption(const string& s): void
+showMaterialInfo(): void

4. Implementation:

G19_Project_D_Resturant_Management_System\Account.h

G19_Project_D_Resturant_Management_System\Account.cpp

G19_Project_D_Resturant_Management_System\account.dat

G19_Project_D_Resturant_Management_System\ConsumptionRecordTuple.h

G19_Project_D_Resturant_Management_System\ConsumptionRecordTuple.cpp

G19_Project_D_Resturant_Management_System\consumptionRecord.dat

G19_Project_D_Resturant_Management_System\Dish.h

G19_Project_D_Resturant_Management_System\Dish.cpp

G19_Project_D_Resturant_Management_System\dish.dat

G19_Project_D_Resturant_Management_System\RawMaterial.h

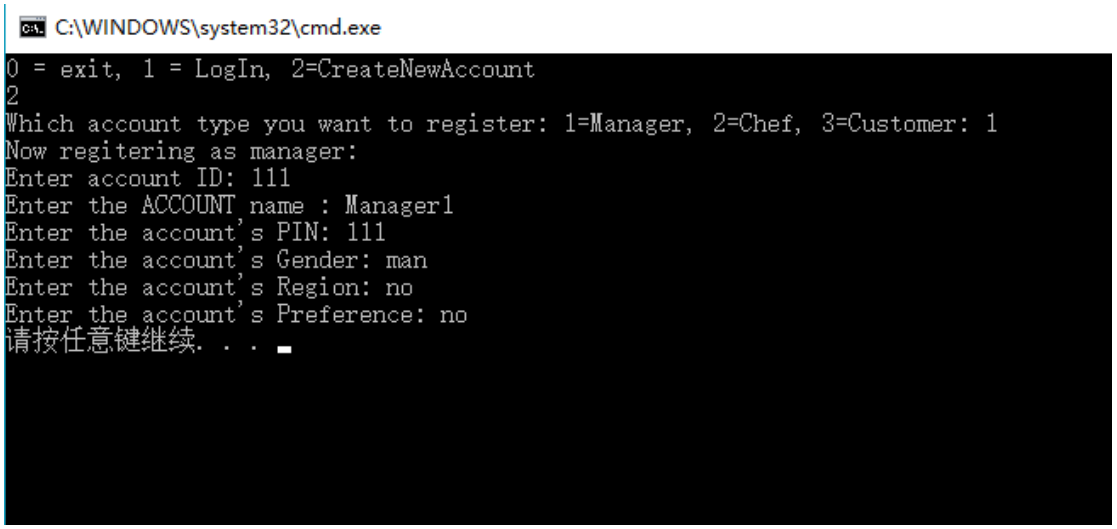
G19_Project_D_Resturant_Management_System\RawMaterial.cpp

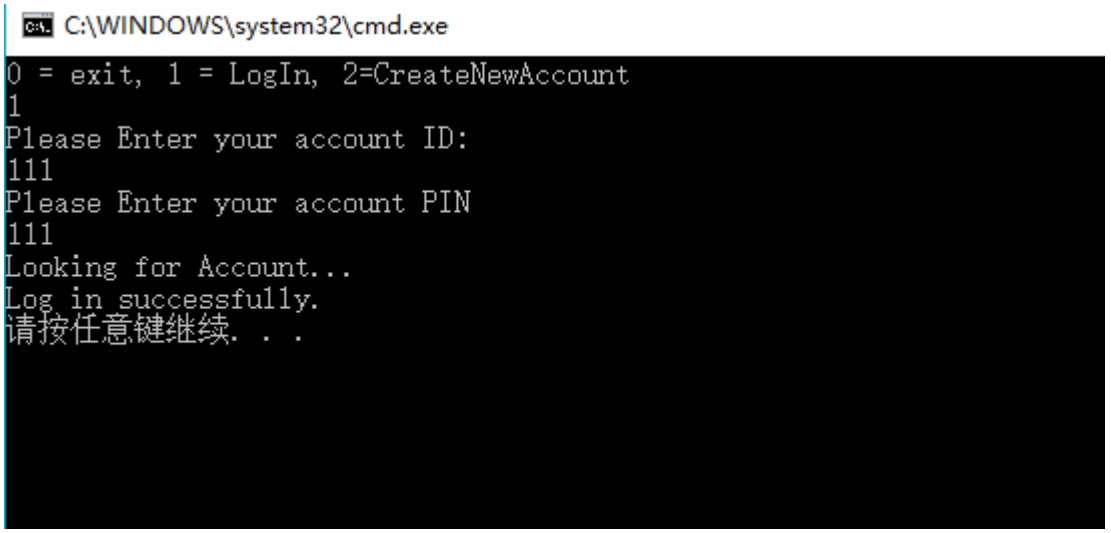
G19_Project_D_Resturant_Management_System\rawMaterial.dat

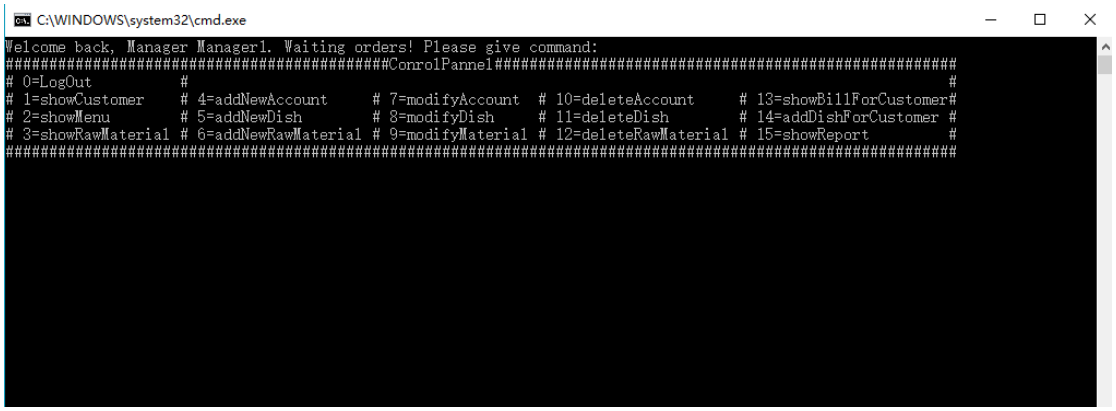
5. Testing & Bugs Report

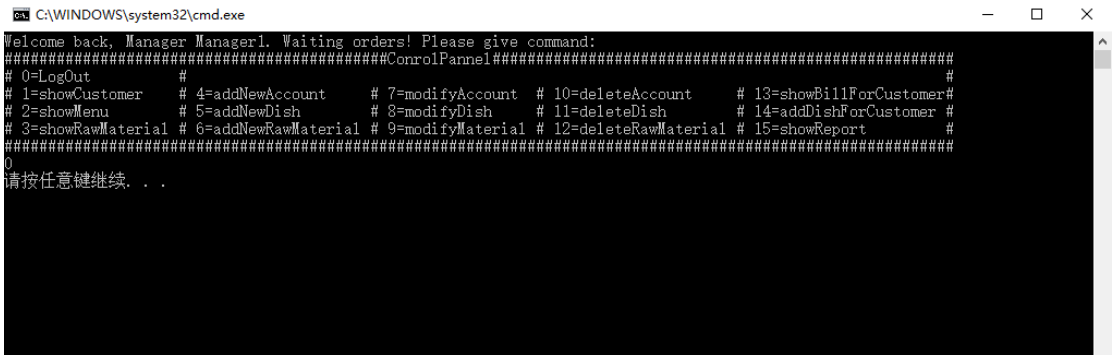
5.1. Account registration & Log in and log out & Viewing authority

Register a manager account and view his authority, finally log out:

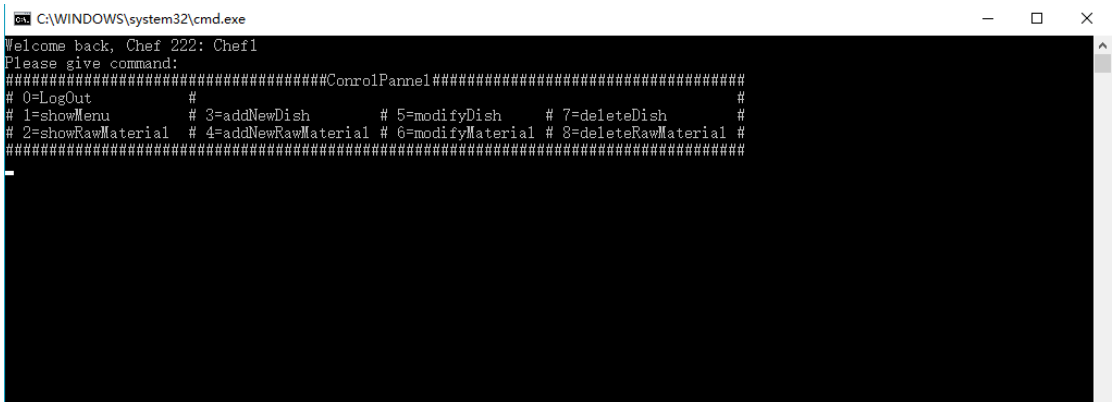




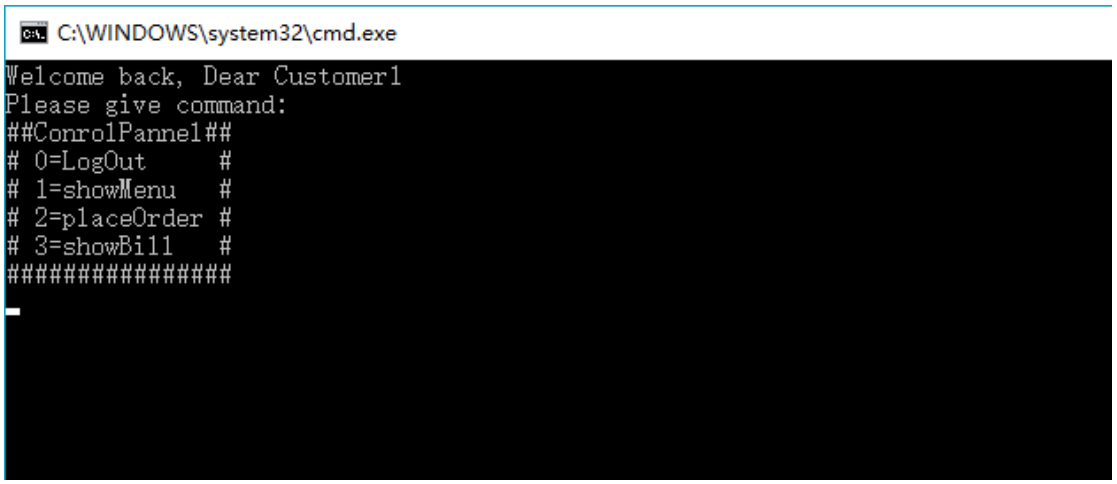




Authority of chefs:

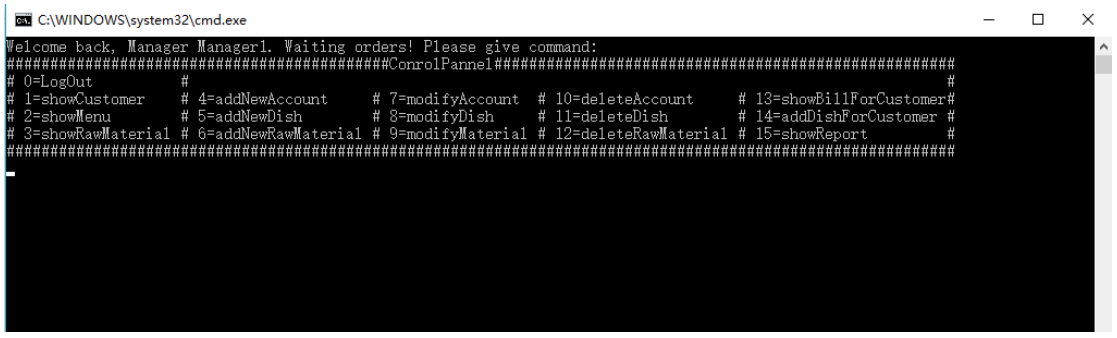


Authority of customers:

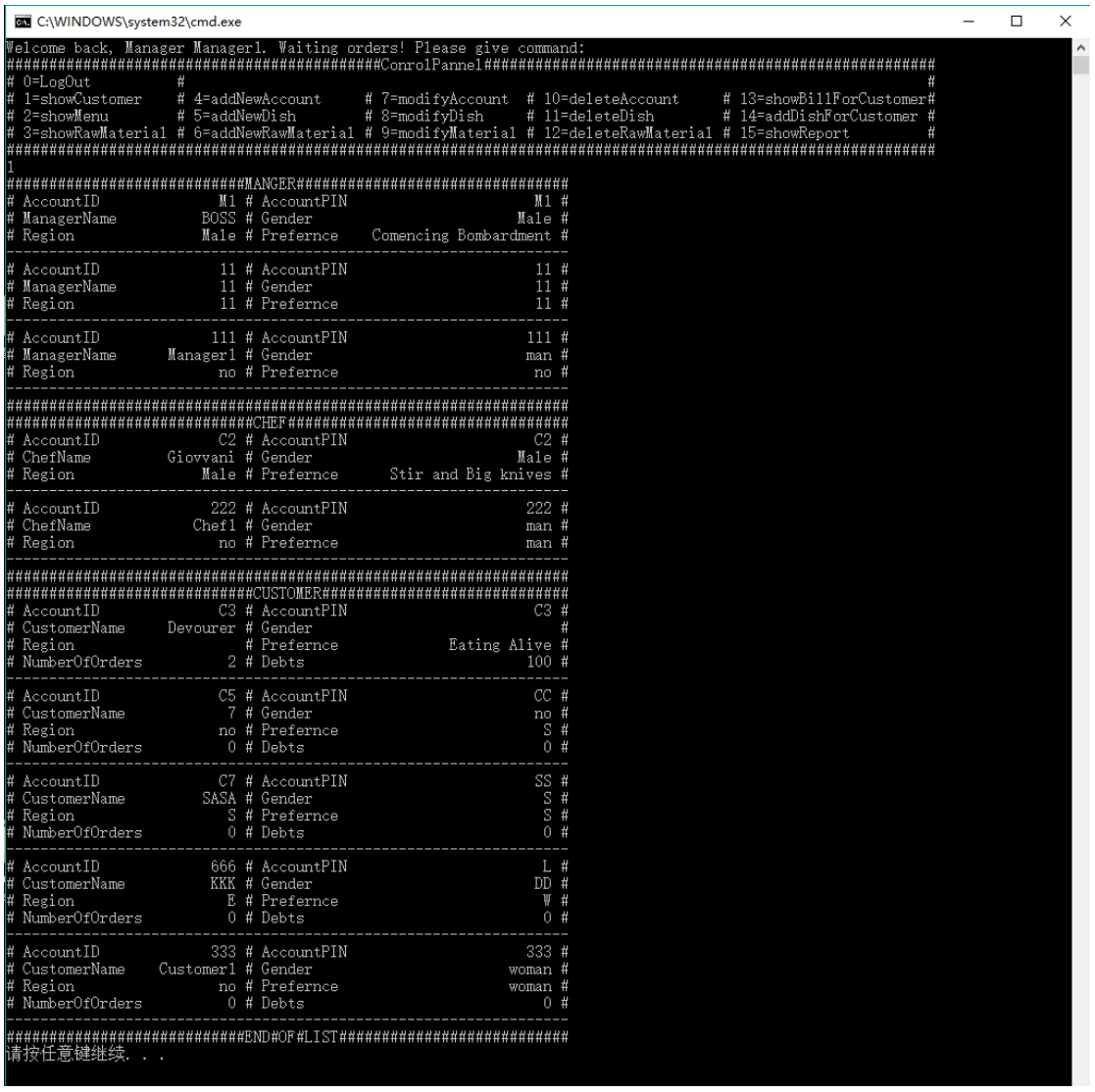


5.2. Test for authority of the manager

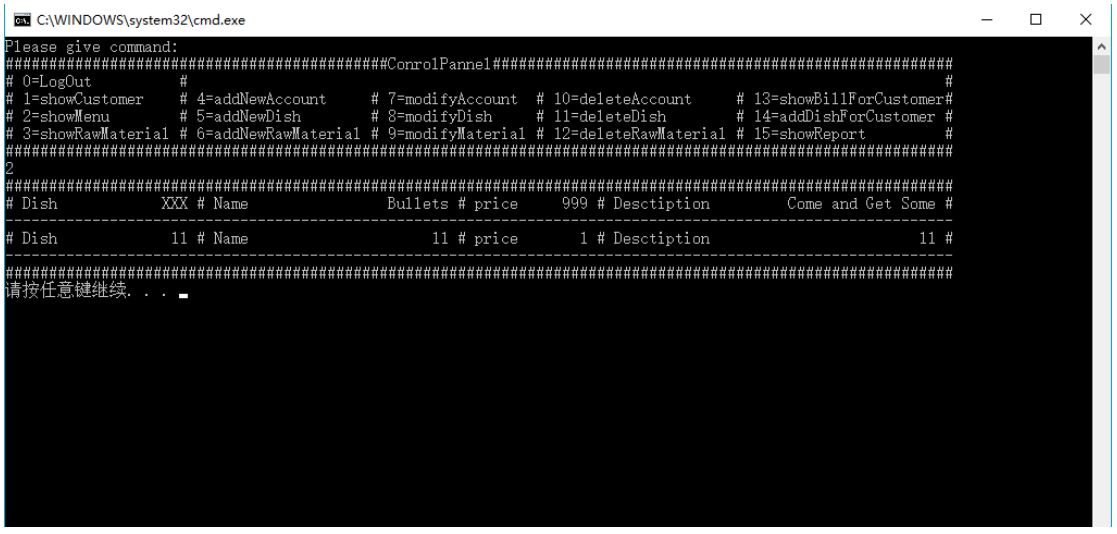
Interface:



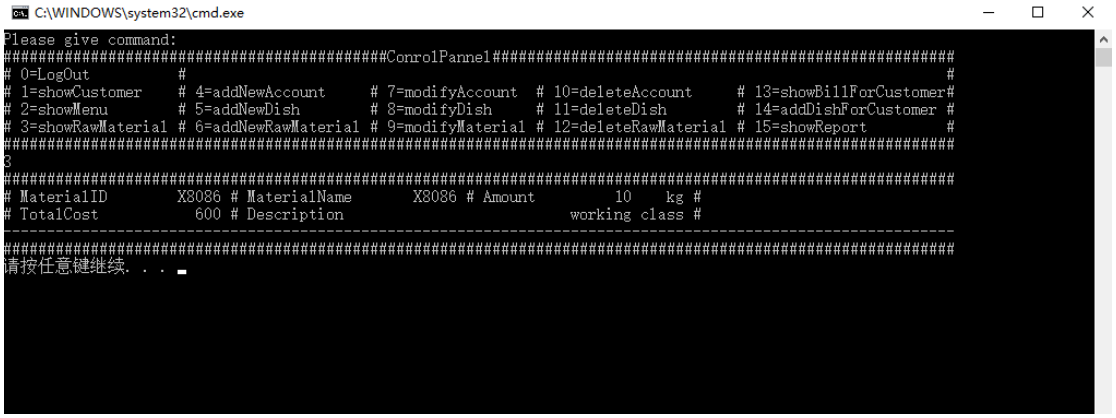
2.1 show customer



2.2 show menu

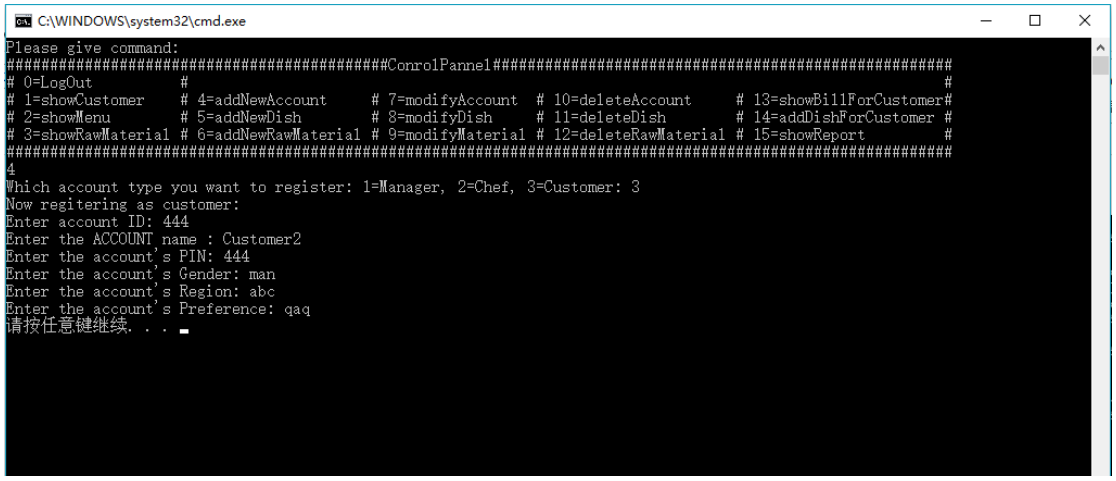


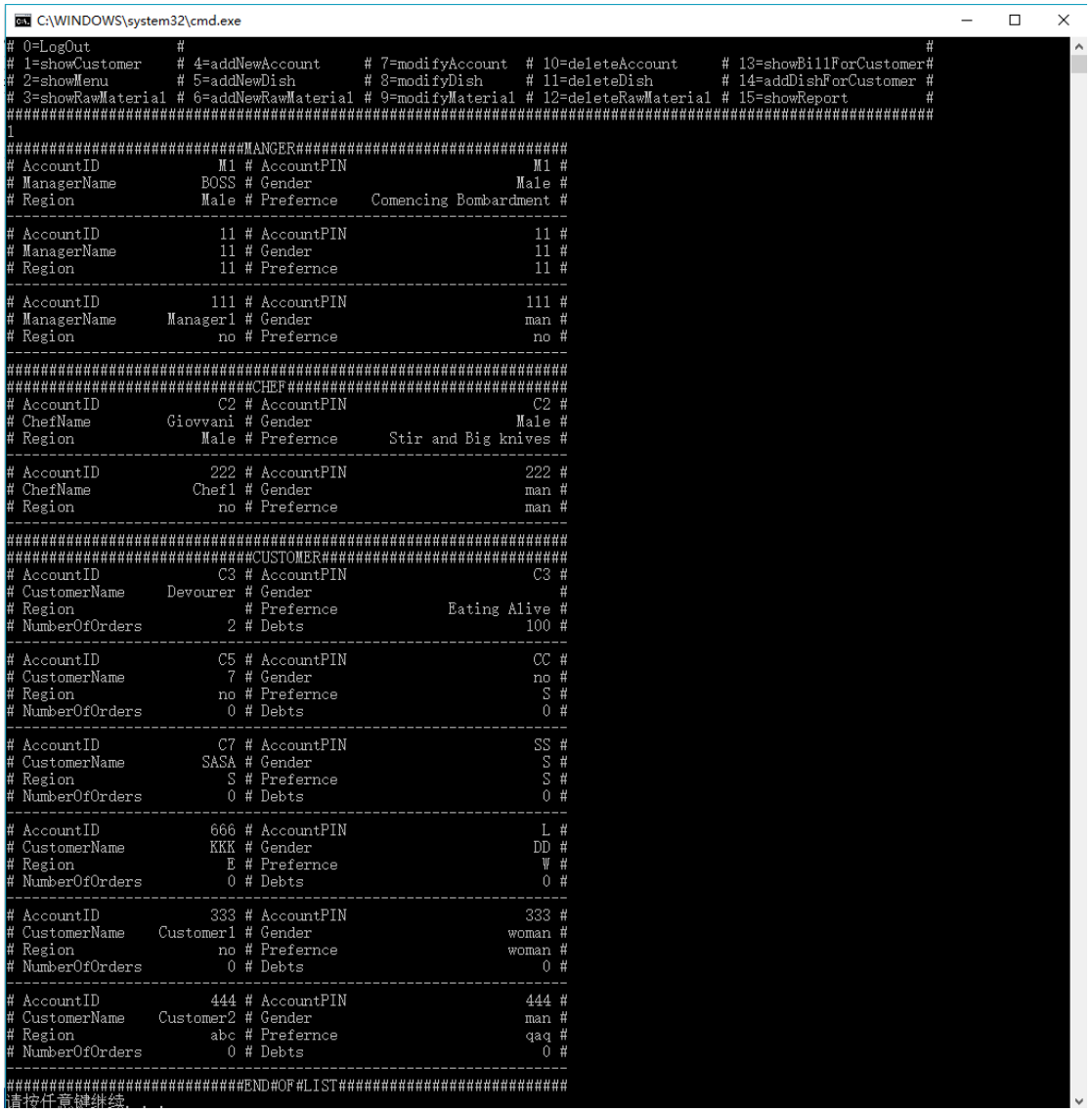
2.3 show raw material



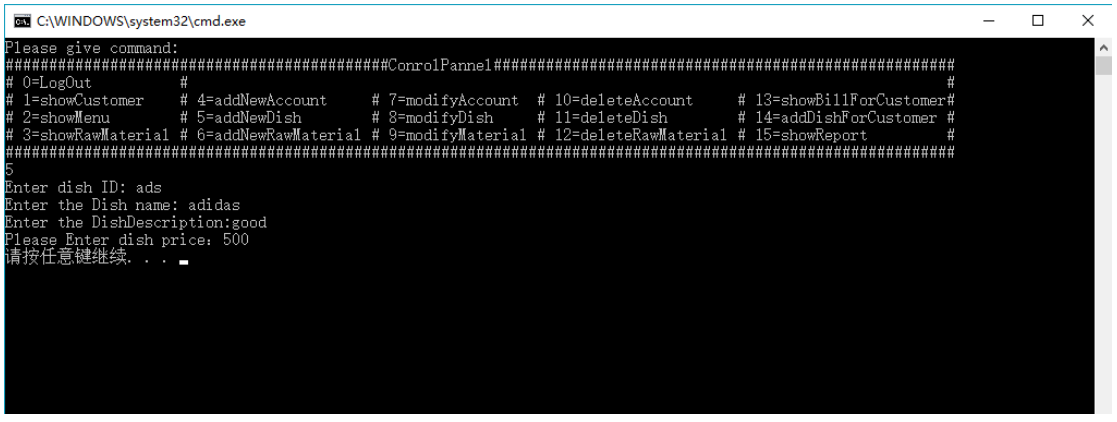
2.4 add new account

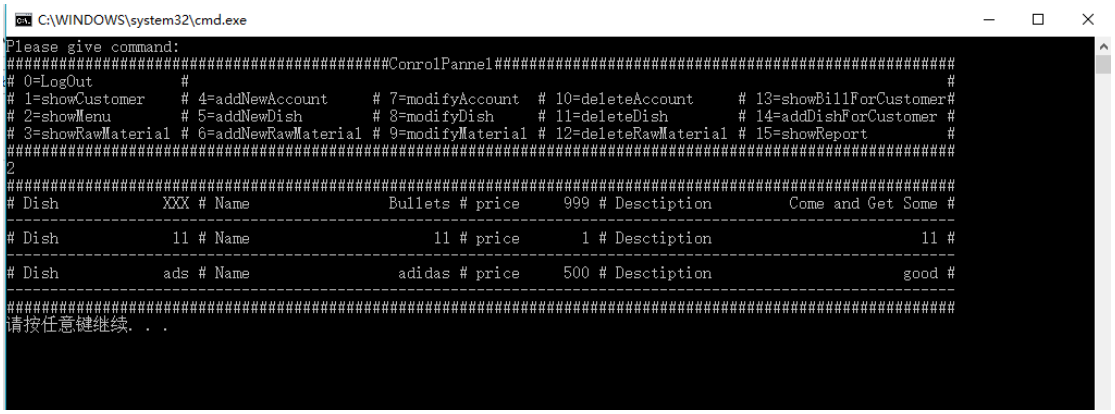
Registering a new customer account for testing.



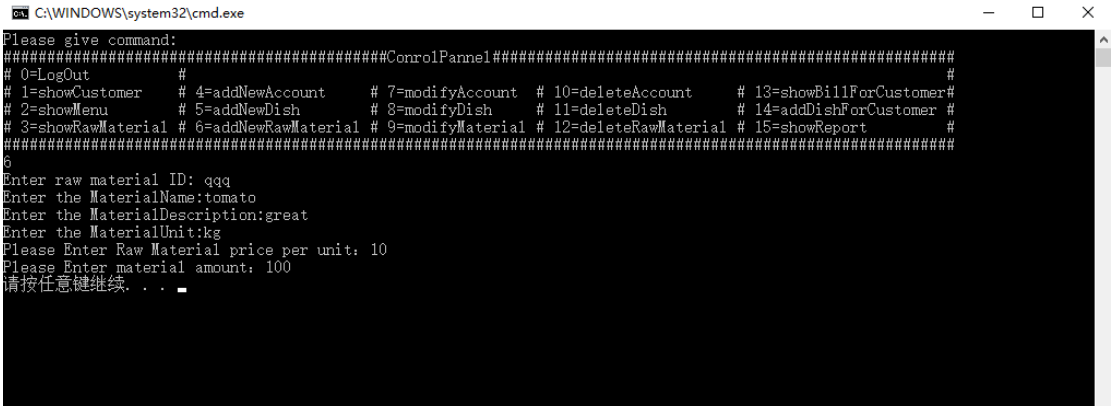


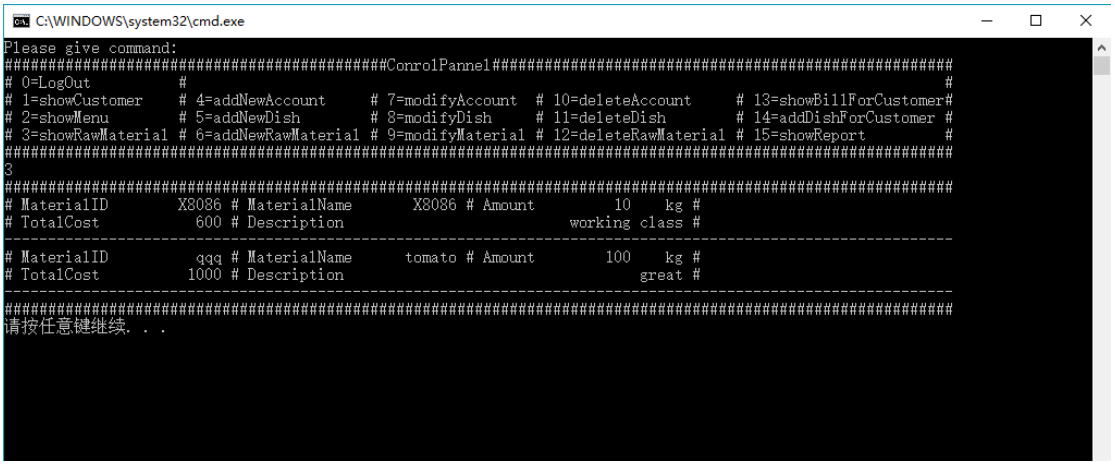
2.5 add new dish



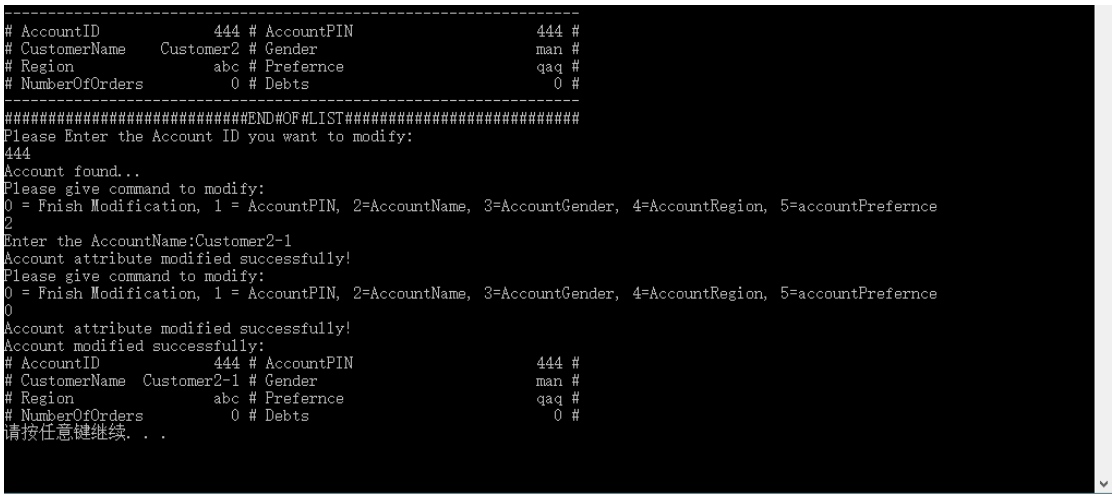


2.6 add new raw material

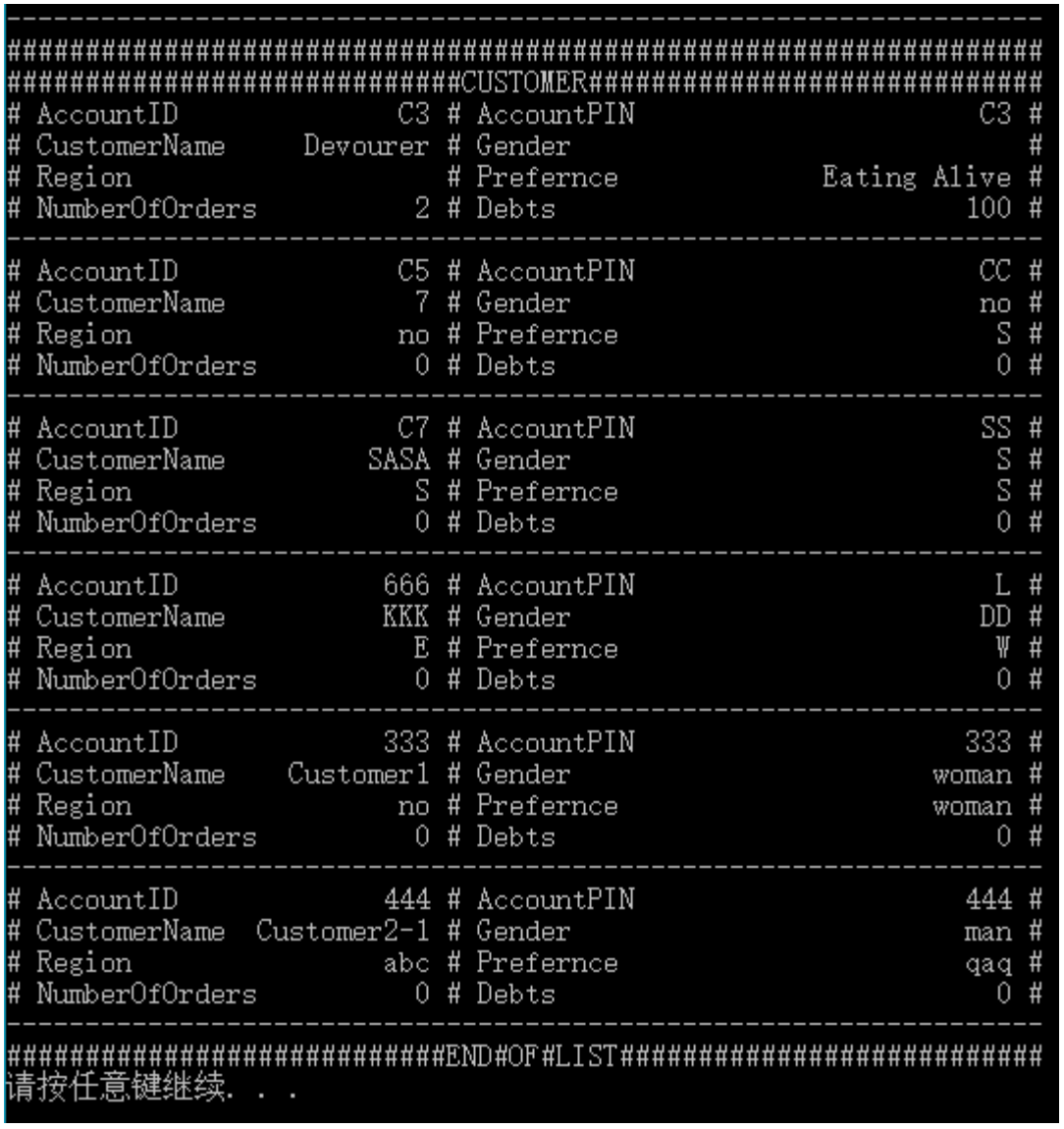




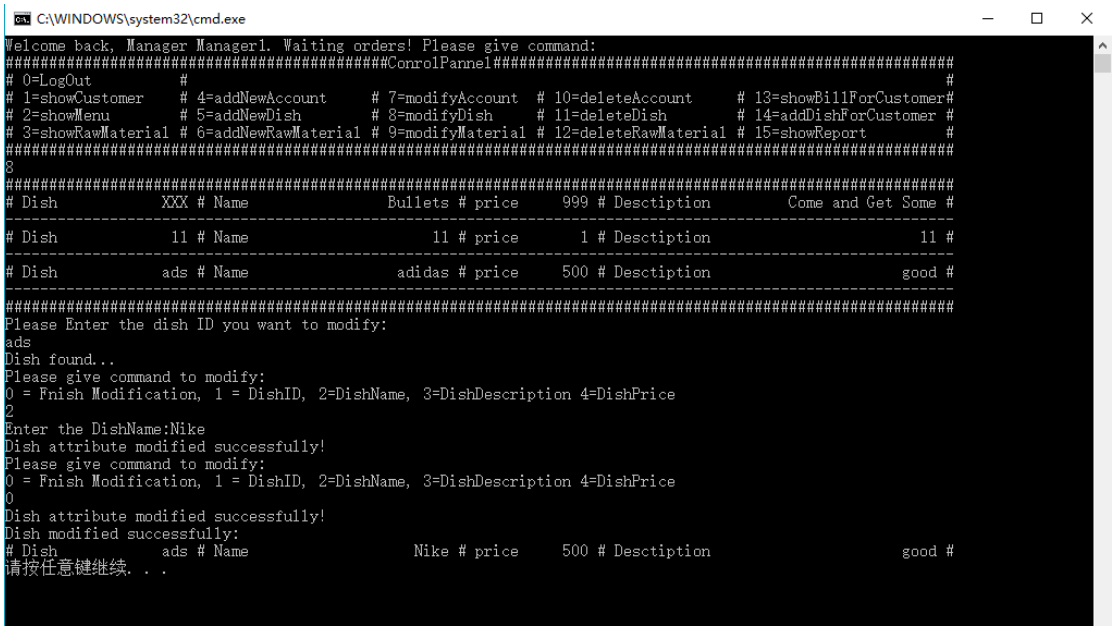
2.7 modify account



Run show customer function, the name has been changed.

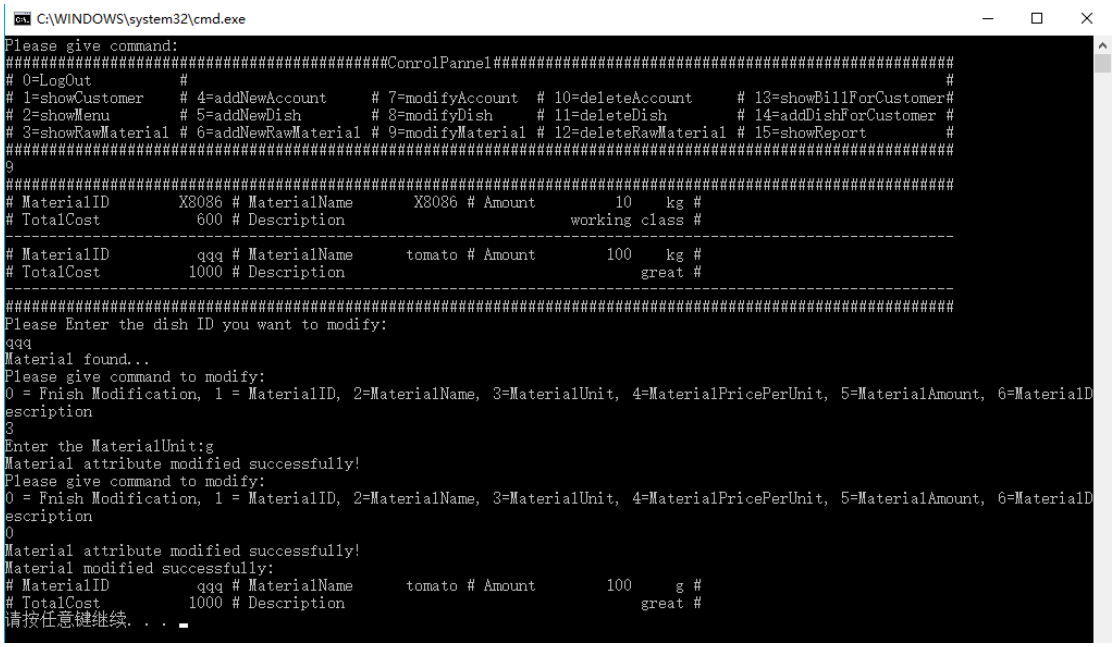


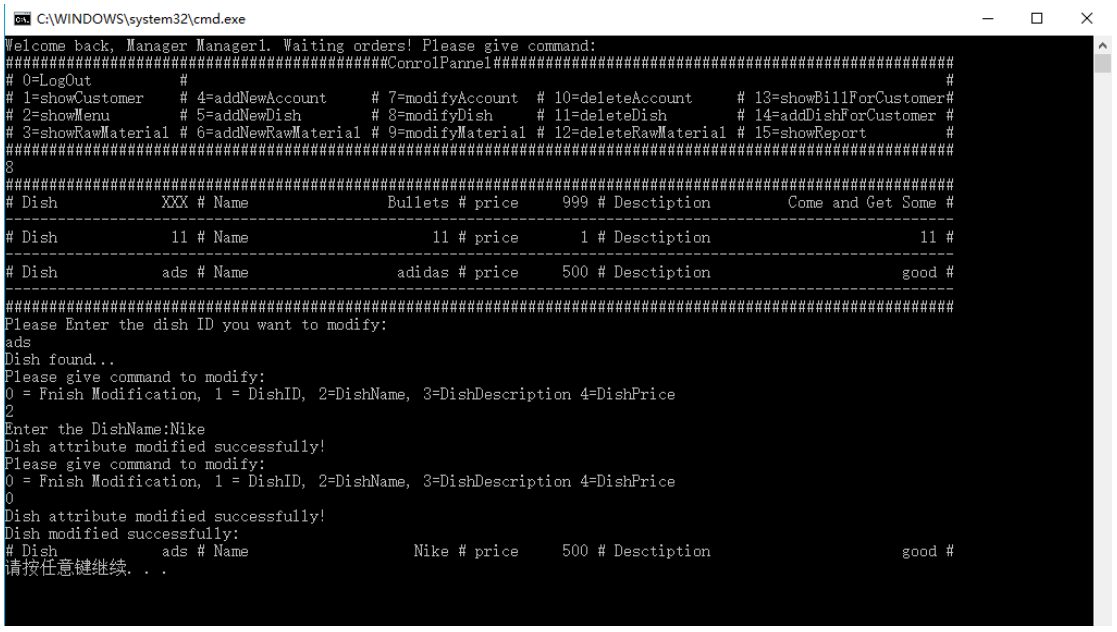
2.8 modify dish



Run show menu function, the name has been changed.

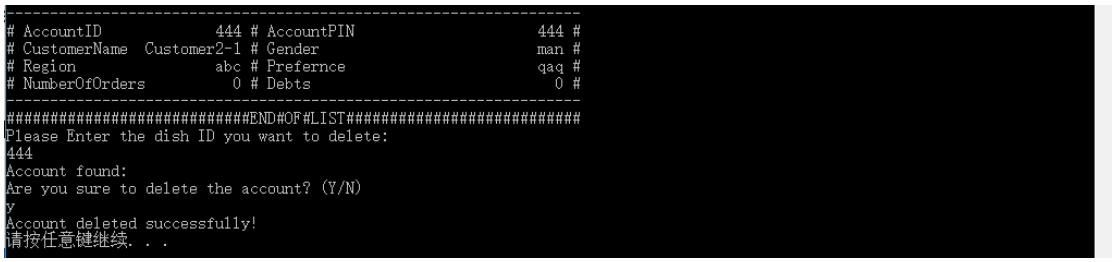
2.9 modify raw material

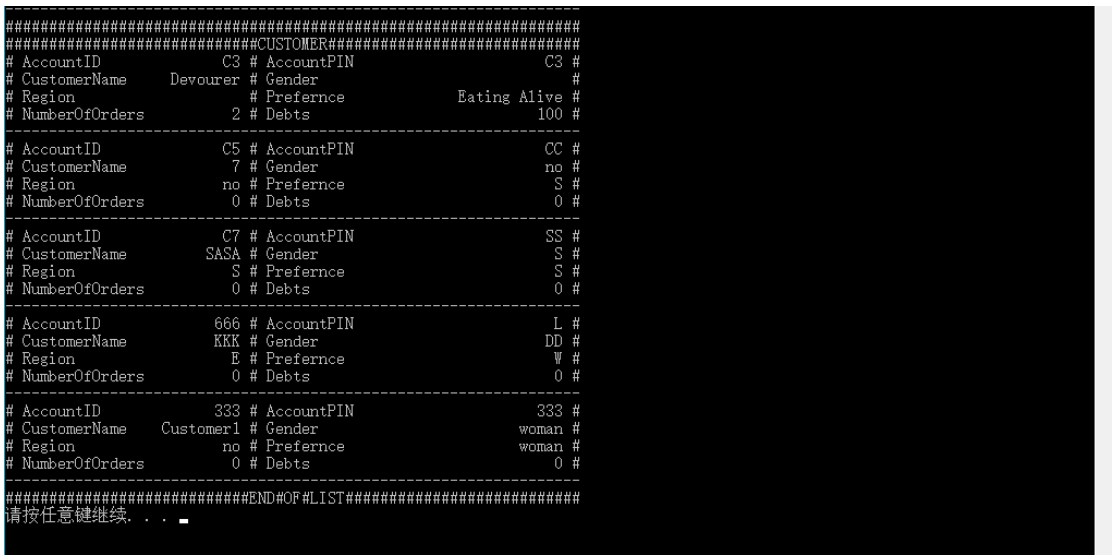




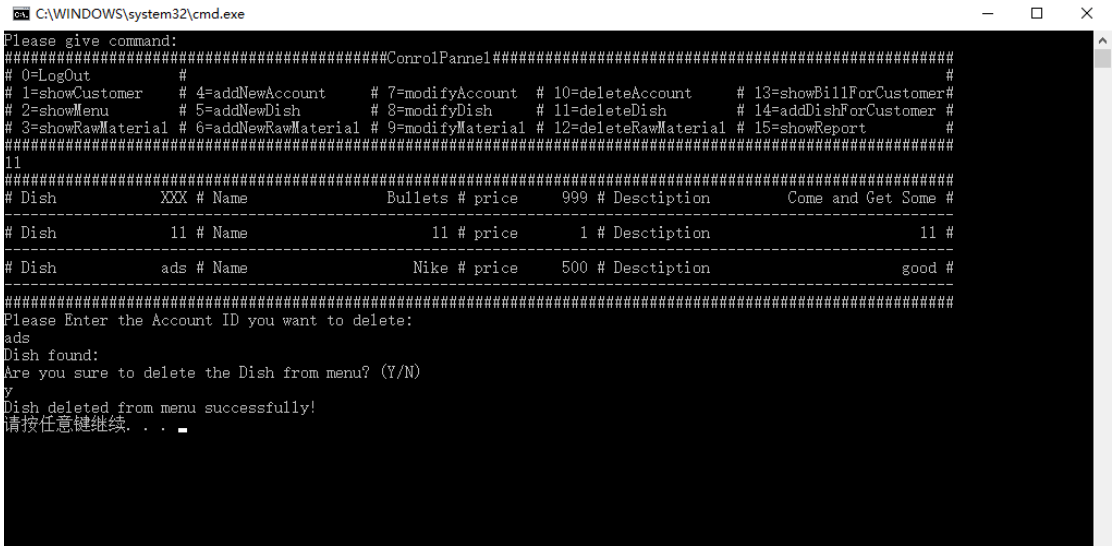
Run show raw material function, the unit has been changed.

2.10 delete account

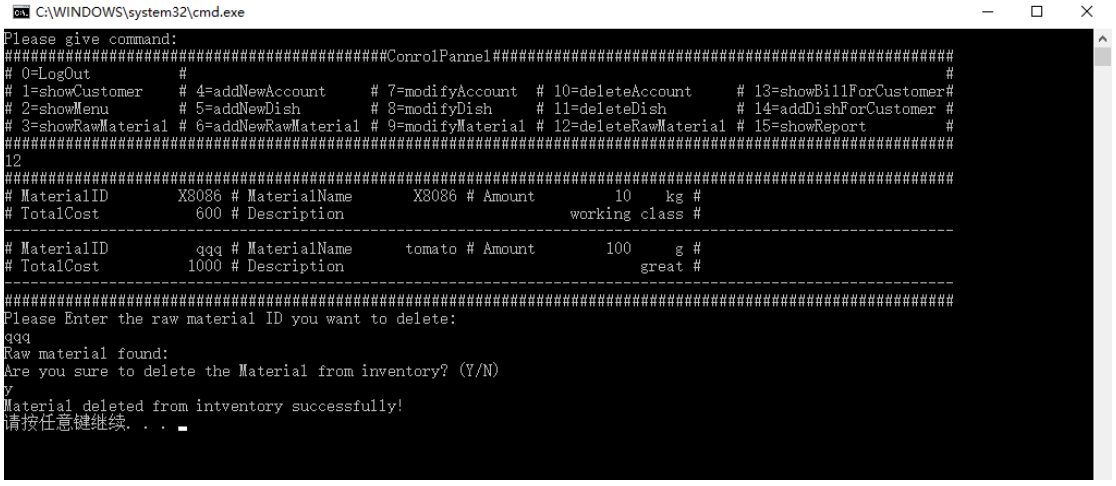


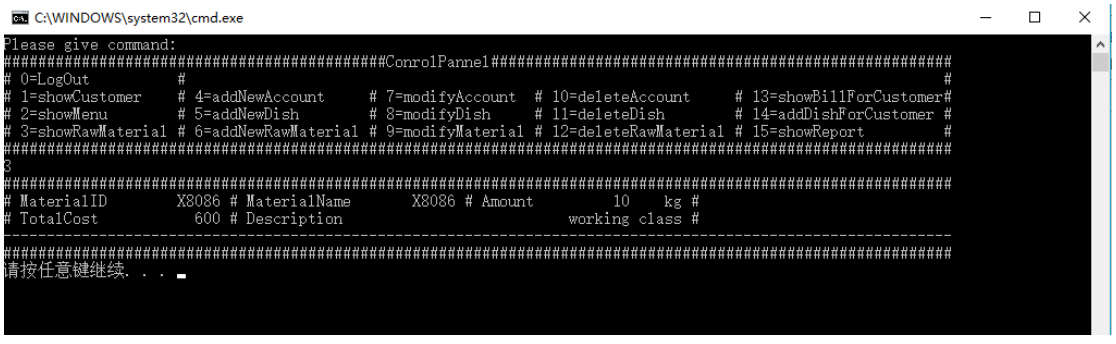


2.11 delete dish

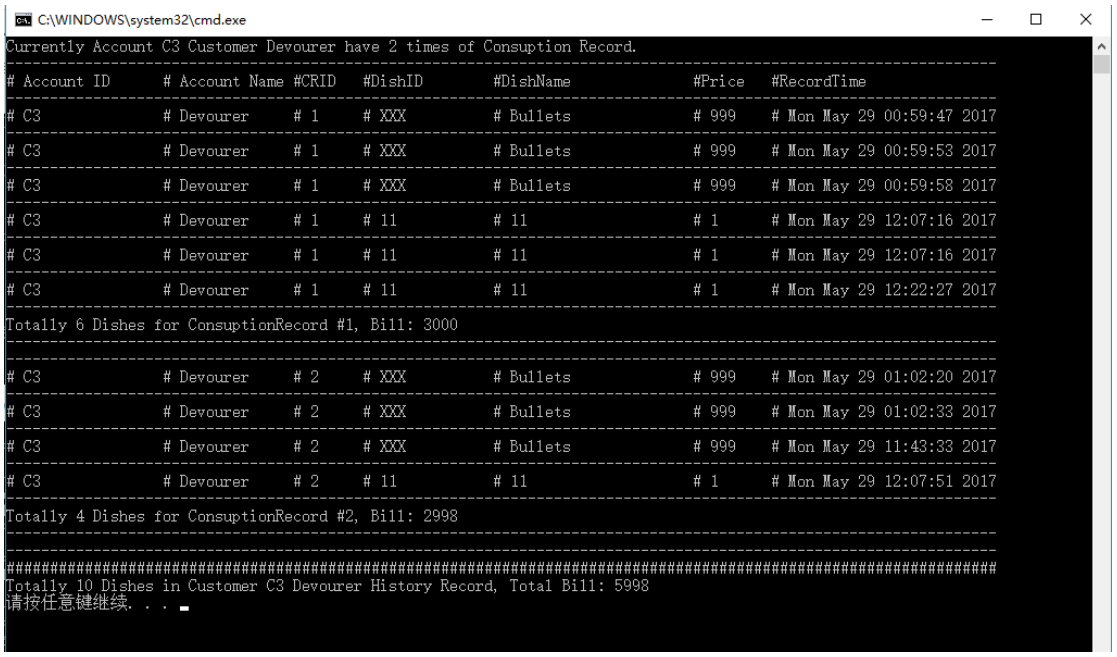


2.12 delete raw material

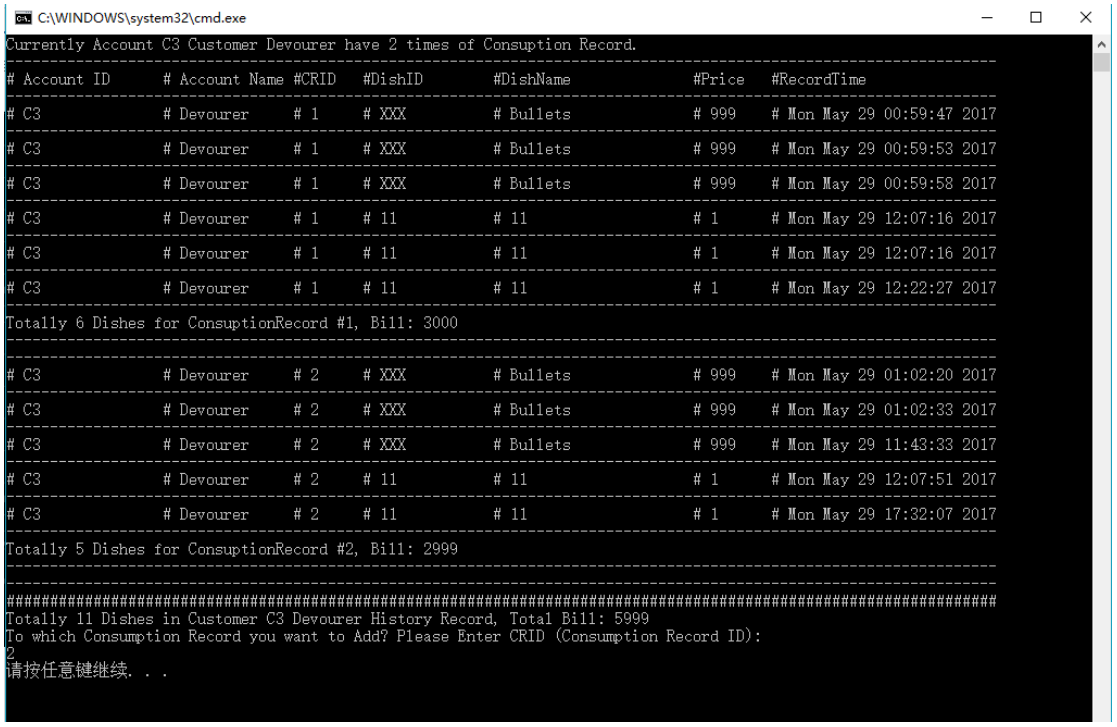


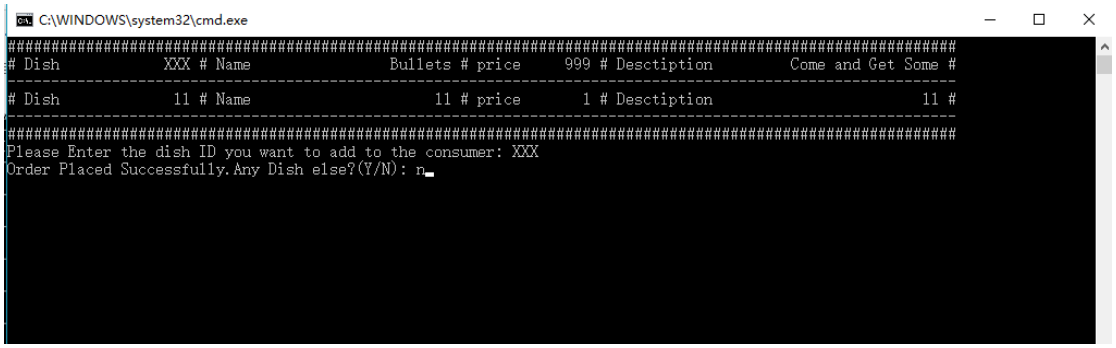


2.13 show bill for a customer

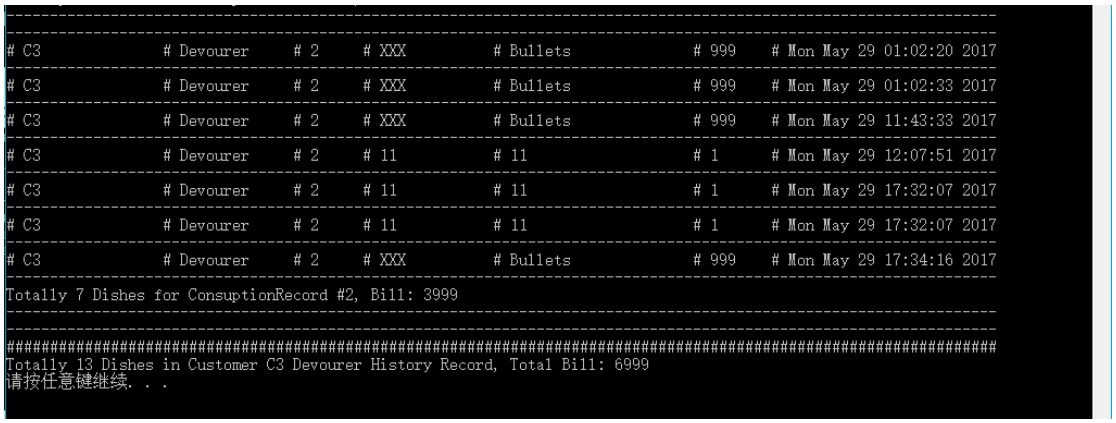


2.14 add dish for a customer

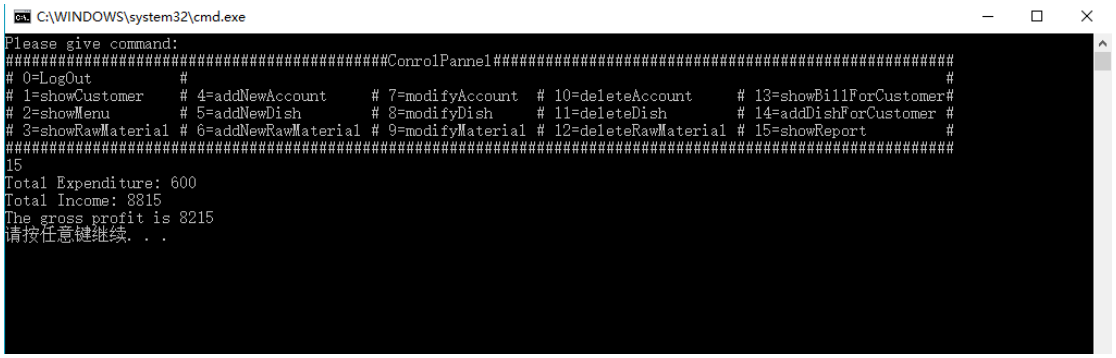




Show bills of customer C3.

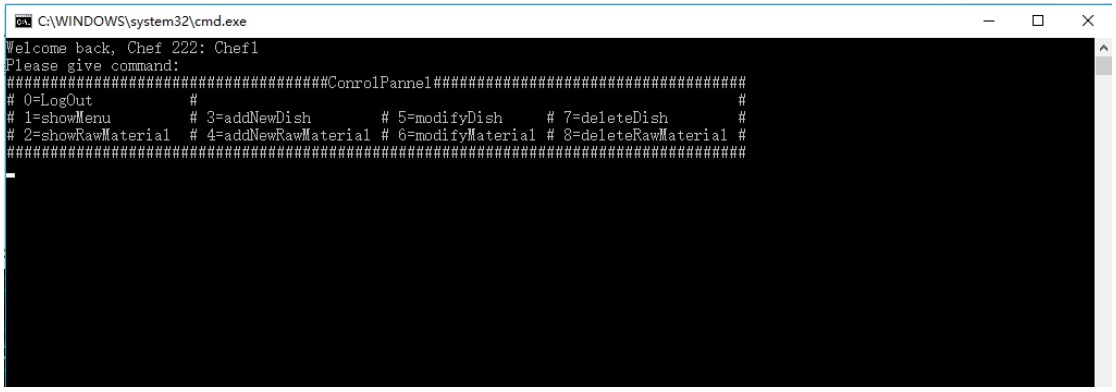


2.15 show report



5.3. Test for authority of the chef

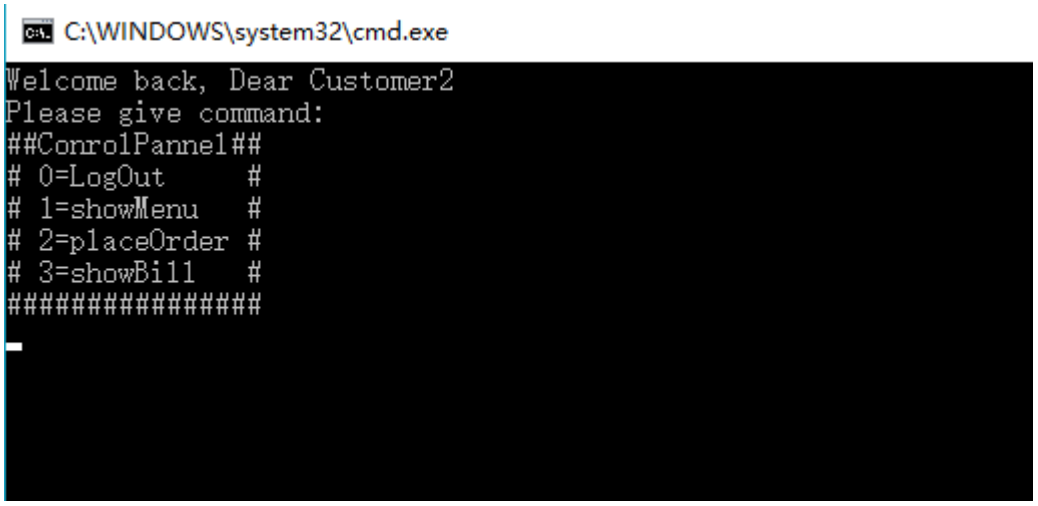
Interface:



The functions from 1 to 8 are all same as those of a manager.

5.4. Test for authority of the customer

Interface:

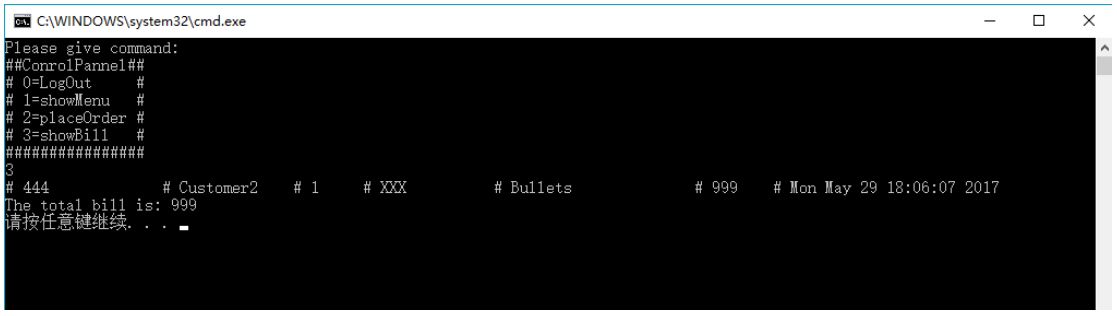


The function 1, which is showMenu, is same as that of a manager.

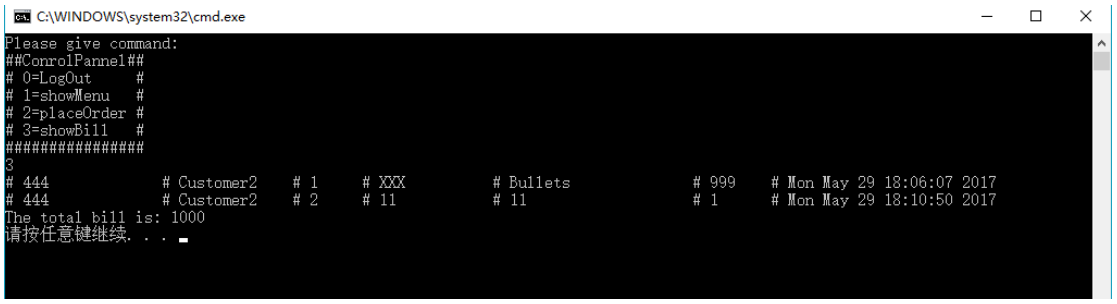
4.1 place order

4.2 show bill

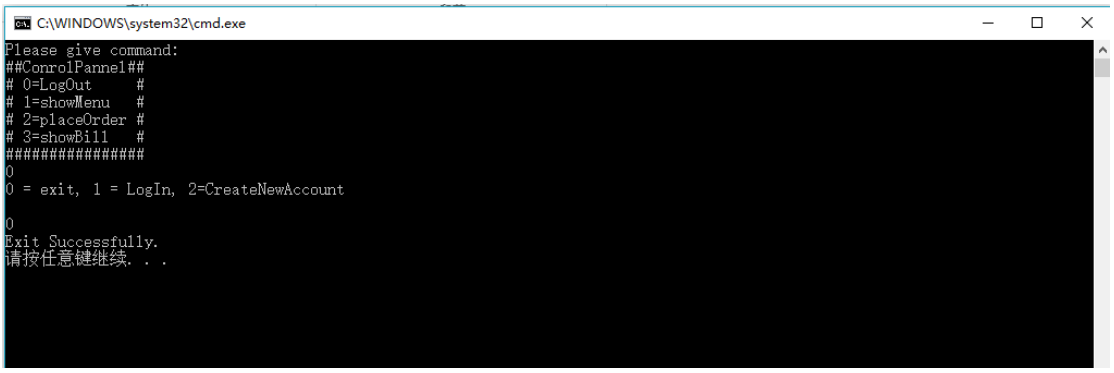
Continue for 4.1.



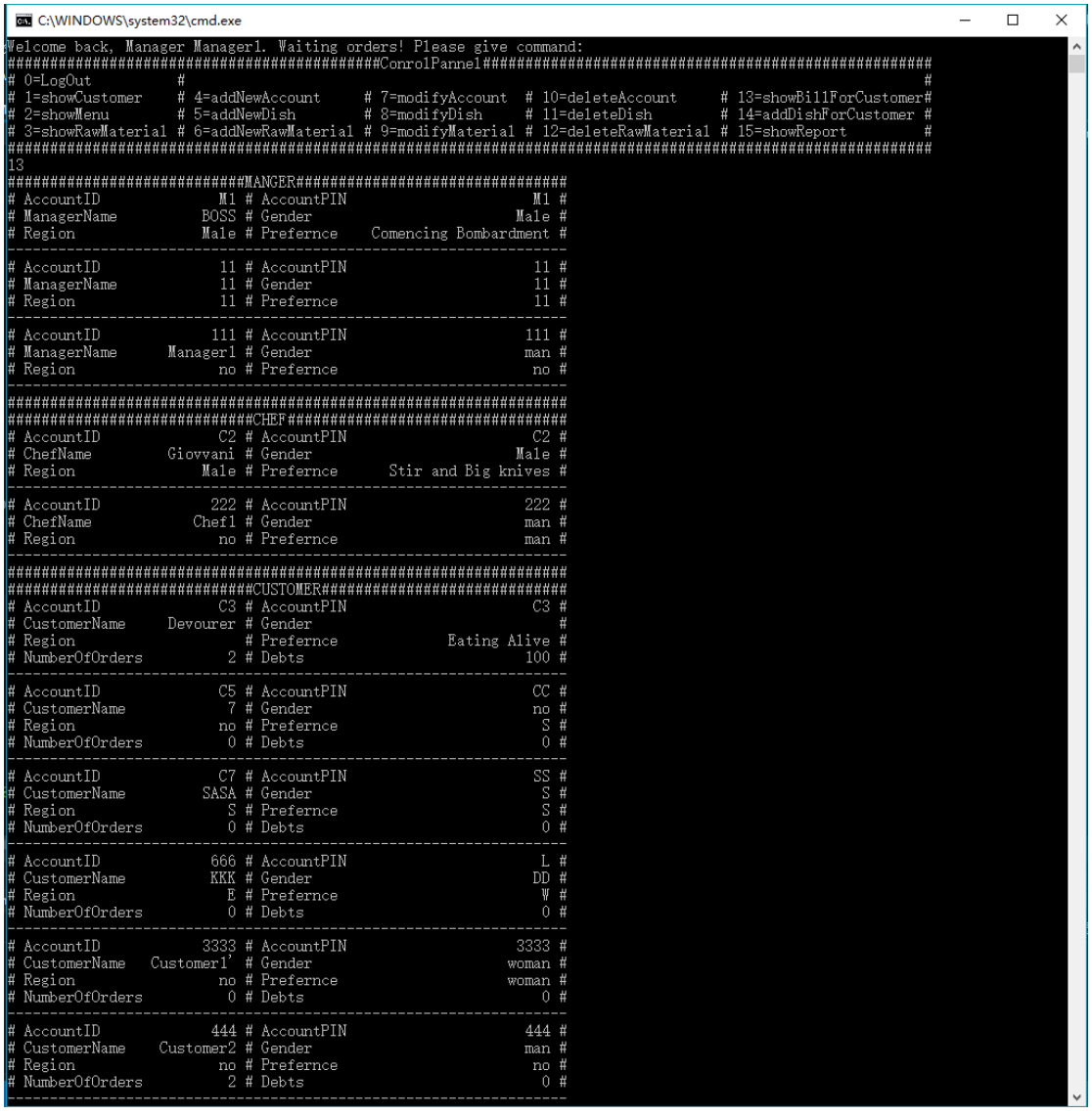
If give another shoot.



5.5. Test for File data



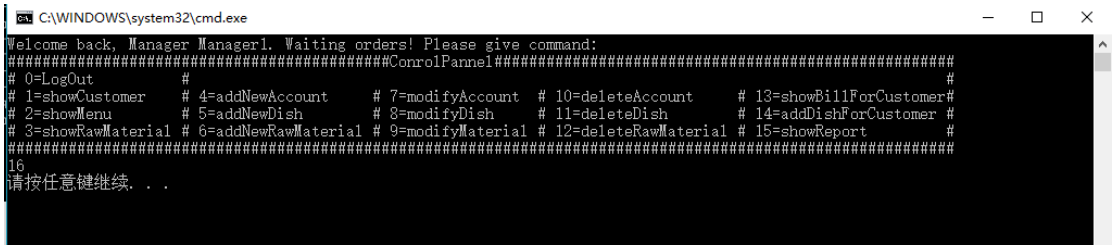
Exit then restart the program, check all data if it is reserved.



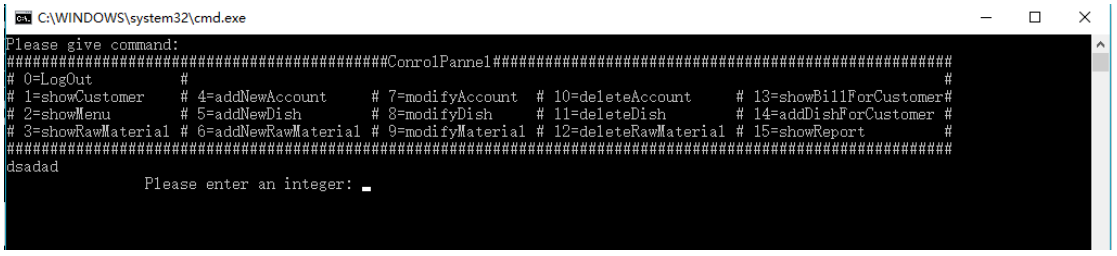
Check the bills.

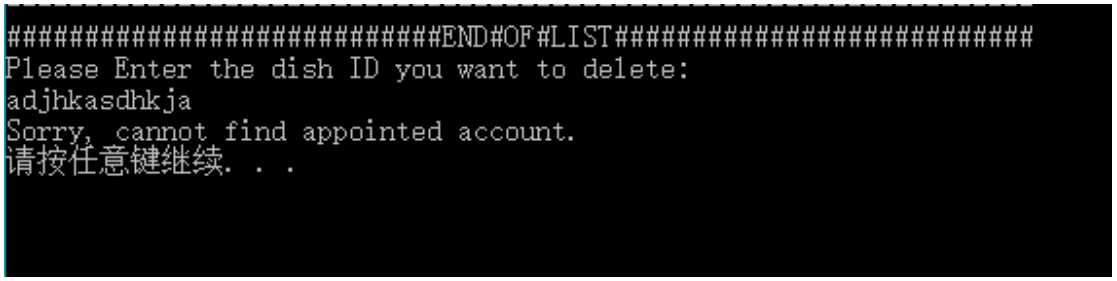
5.6. Test for erroneous input

(1) Inputs that exceed the range



(2) Inputs that are invalid

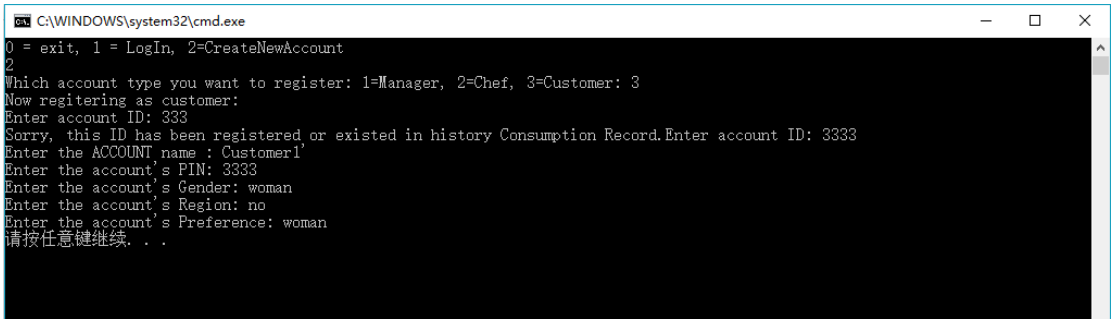




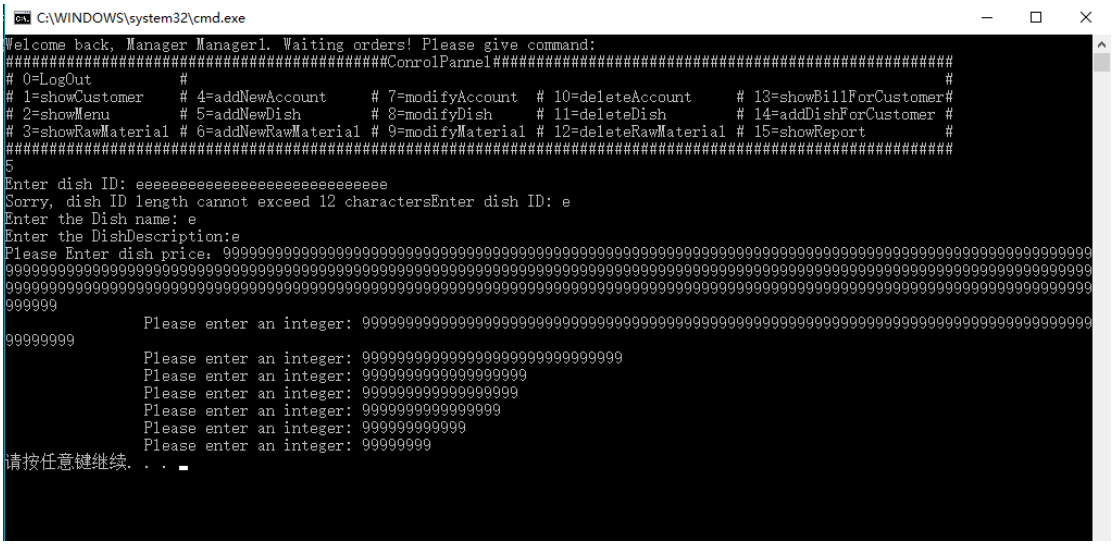
(3) Inputs that exceed the length of strings

5.7. Test for several special cases

(1) After deleting the account of customer 333. The program will prevent you from registering the same ID account.



5.8. Bugs report


C:\WINDOWS\system32\cmd.exe
Welcome back, Manager Manager1. Waiting orders! Please give command:
##########ConrolPannel##########
0=LogOut #
1=showCustomer # 4=addNewAccount # 7=modifyAccount # 10=deleteAccount # 13=showBill1ForCustomer#
2=showMenu # 5=addNewDish # 8=modifyDish # 11=deleteDish # 14=addDishForCustomer
3=showRawMaterial # 6=addNewRawMaterial # 9=modifyMaterial # 12=deleteRawMaterial # 15=showReport
5
Enter dish ID: eeeeeeeeeeeeeeeeeeeeeeeeee
Sorry, dish ID length cannot exceed 12 charactersEnter dish ID: e
Enter the Dish name: e
Enter the DishDescription:e
Please Enter dish price: 999999999

6. User Manual

This is the user manual which give a brief description for each function for the restaurant system users.

In the initial interface, there are three choices for the users.

1. Exit:

Allow the users to quit the operation system.

2. Log in:

Provide the function of logging in the specific account.

The log process need the accountID and the corresponding PIN.

3. CreatNewAccount:

There are three kinds of account the users can provide：manager, chef and customer. Each account contains the basic messages which the users need to input: accountID, name, PIN, gender, region, preference.

3.1 manager:

The user can materialize the specific functions by entering the specific number.

a. Entering “0”: logout

If the user input “0”, the system would quit the current account and return the initial interface.

b. Entering “1”: showCustomer

If the user inputs “1”, the system will display the basic information of all the system users.

c. Entering “2”: showMenu,

If the user inputs “2”, the system will display the information of the menu, however, if no dishes have been added, the users will be told the menu is empty.

d. Entering “3”: showRawMaterial

If the user inputs “3”, the system will display the information of the menu, however, if no raw materials have been added, the users will be told the inventory is empty.

e. Entering “4”: addNewAccount

If the user inputs “4”, a new account will be added. After choosing the account type, the corresponding basic messages need to be completed.

f. Entering “5”: addNewDish

If the user inputs “5”, a new dish will be added to the menu. Additionally, the corresponding basic information includes dish ID, dish name, dish description and prize need to be completed.

g. Entering “6”: addNewRawMaterial

If the user inputs “6”, a new raw material will be added to the inventory. Additionally, the corresponding basic information need to be completed. The information includes material

ID, material name, material description, material unit, the prize per unit and the material amount.

h. Entering “7”: modifyAccount

If the user inputs “7”, the system will display the information of the current account and ask the user to input the specific ID of the corresponding account. If the account was found, the user can modify the information by inputting the specific number. After the modifying is over, the new account information will be displayed again.

i. Entering “8”: modifyDish

If the user inputs “8”, the system will display the information of the current menu and ask the user to input the specific ID of the corresponding dish. If the dish was found, the user can modify the information by input the specific number. The new information will be shown again after the modifying.

j. Entering “9”: modifyMaterial

If the user inputs “9”, the system will display the information of the current inventory and ask the user to input the specific ID of the corresponding material. If the material was found, the user can modify the information by input the specific number. The new information of the materials will be shown again after the modifying.

k. Entering “10”: deleteAccount

If the user inputs “10”, the system will display the information of the current account and ask the user to input the ID of the account which the user want to delete. Then, the system will ask the user Y/N for this decision, if the customer input Y and the account will be deleted successfully.

l. Entering “11”: deleteDish

If the user inputs “11”, the system will display the information of the current account and ask the user to input the ID of the account which the user want to delete. Then, the system will ask the user Y/N for this decision, if the customer input Y and the account will be deleted successfully.

m. Entering “12”: deleteRawmaterial

If the user inputs “12”, the system will display the information of the current inventory and ask the user to input the ID of the material which the user want to delete. Then, the system will ask the user Y/N for this decision, if the customer input Y and the material will be deleted successfully from the inventory.

n. Entering “13”: showBillForCustomer

If the user inputs “13”, the whole information for all users will be displayed. Then, the system will ask the users to choose the account ID which need to be shown the bill. However, if the account of manager or chef is chosen, the system will not give the bill for they don’t have the record of debts. After choosing the account ID of the customer, the bill contains the information of ordered dishes as well as the expenses will be displayed.

o. Entering “14”: addDishForCustomer

If the user inputs “14”, the whole information for all users will be displayed. Then, the system will ask the users to choose the account ID which need to be shown the bill. However, if the account of manager or chef is chosen, the system will not give the response that they can’t order dishes. After choosing the account ID of the customer, the user can also choose the assumption record from the system.

p. Entering “15”: showReport

If the user inputs “15”, the report contains the expenditure and income as well as the gross profit will be displayed.

3.2 Chef:

a. Entering “0”: logout

If the user input “0”, the system would quit the current account and return the initial interface.

b. Entering “1”: showMenu

If the user inputs “1”, the system will display the information of the menu, however, if no dishes have been added, the users will be told the menu is empty.

c. Entering “2”: showRawMaterial

If the user inputs “2”, the system will display the information of the menu, however, if no raw materials have been added, the users will be told the inventory is empty.

d. Entering “3”: addNewDish

If the user inputs “3”, a new dish will be added to the menu. Additionally, the corresponding basic information includes dish ID, dish name and prize need to be completed.

e. Entering “4”: addNewRawMaterial

If the user inputs “6”, a new raw material will be added to the inventory. Additionally, the corresponding basic information need to be completed. The information includes material ID, material name, material description, material unit, the prize per unit and the material amount.

f. Entering “5”: modifyDish

If the user inputs “5”, the system will display the information of the current menu and ask the user to input the specific ID of the corresponding dish. If the dish was found, the user can modify the information by input the specific number. The new information will be shown again after the modifying.

g. Entering “6”: modifyMaterial

If the user inputs “6”, a new raw material will be added to the inventory. Additionally, the corresponding basic information need to be completed. The information includes material ID, material name, material description, material unit, the prize per unit and the material amount.

h. Entering “7”: deleteDish

If the user inputs “11”, the system will display the information of the current account and ask the user to input the ID of the account which the user want to delete. Then, the system will ask the user Y/N for this decision, if the customer input Y and the account will be deleted successfully.

i. Entering “8”: deleteRawMaterial

If the user inputs “12”, the system will display the information of the current inventory and ask the user to input the ID of the material which the user want to delete. Then, the system will ask the user Y/N for this decision, if the customer input Y and the material will be deleted successfully from the inventory.

3.3 Customer:

a. Entering “0”: logout

If the user input “0”, the system would quit the current account and return the initial interface.

b. Entering “1”: showMenu

If the user inputs “1”, the system will display the information of the menu, however, if no dishes have been added, the users will be told the menu is empty.

c. Entering “2”: placeOrder

If the user inputs “2”, the menu will be displayed. Then, the user can input the specific ID to order the corresponding dish. After that, the system will ask the user Y/N for this decision, if the customer input Y and the dish will be ordered successfully.

d. Entering “3”: showBill

If the user inputs “3”, the bill contains the information of ordered dishes as well as the expenses will be displayed.