UTM
UNIVERSITI TEKNOLOGI MALAYSIA

SCHOOL OF COMPUTING
Faculty of Engineering

Assignment 1 (Group)

SECJ2013 DATA STRUCTURE AND ALGORITHM

SEMESTER I, SESSION 2020/2021

Topic: Food Ordering System

Lecturer: Dr. Johanna binti Ahmad

Name	Matric No.
KONG HAO YANG	A19EC0065
LOO ZHI XUEN	A19EC0078
SEE WEN XIANG	A19EC0206

Section: 06

Programme: Bachelor of Computer Science (Software Engineering)

Table of Contents

1.0 Objective

The main objectives of developing the Food Ordering System is:

- To automate food ordering process in a restaurant
- To apply sorting and searching technique in the system

2.0 Synopsis

Introduction

Food Ordering System is a software application which helps a restaurant to control and optimize over their order. This system is mainly designed for customer by making them to be more convenient in ordering food in a restaurant. For the customer's point of view, the customers are able to view menu, order food, view their order and make payment on their own in an easier manner. Then, staff can view all the payment list after the customers have done their payment.

This system helps the restaurant to reduce the human resources required because once customers step into restaurant, the whole food ordering process from viewing menu until making payment all is fully automated which can be done by customers themselves. Besides that, this system is able to do all the functions of different roles such as waiters and cashier involved in the restaurant more accurately and in a faster way. This system provides options for customer to let them sort food menu based on alphabet of the name of food or based on price of food in either ascending or descending orders. The program enables customers to make payment by searching their table number or their name.

The main goal of this Food Ordering System is to maintain the restaurant's functions in a more accurate manner and effective and also reduce the use of manual entries.

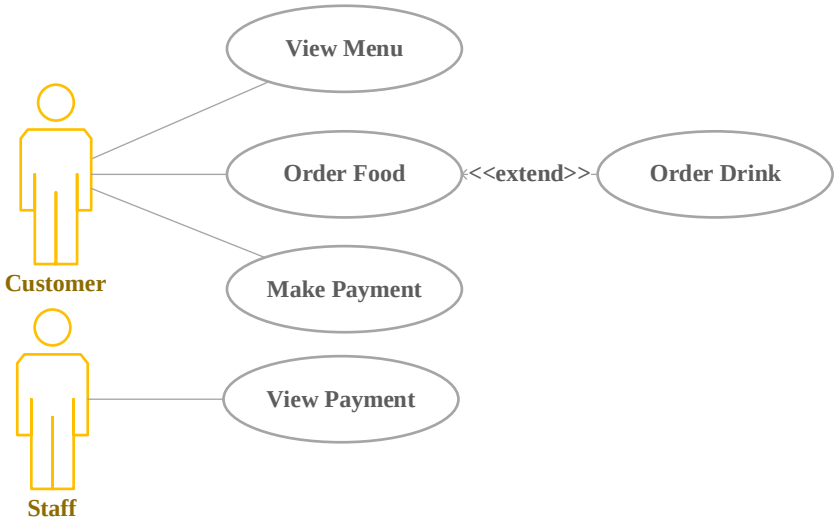

Figure 2.1: Use Case Diagram

System Functionality

Customer

- View the menu and sort the menu based on their preferences.
- Insert name, table number and item code to place the order.
- Add item to the order and remove item from the order.
- Search order based on their name or table number in order to make payment.
- Check the status of the payment whether the order is paid.

Staff

- View the list of all customer's payments

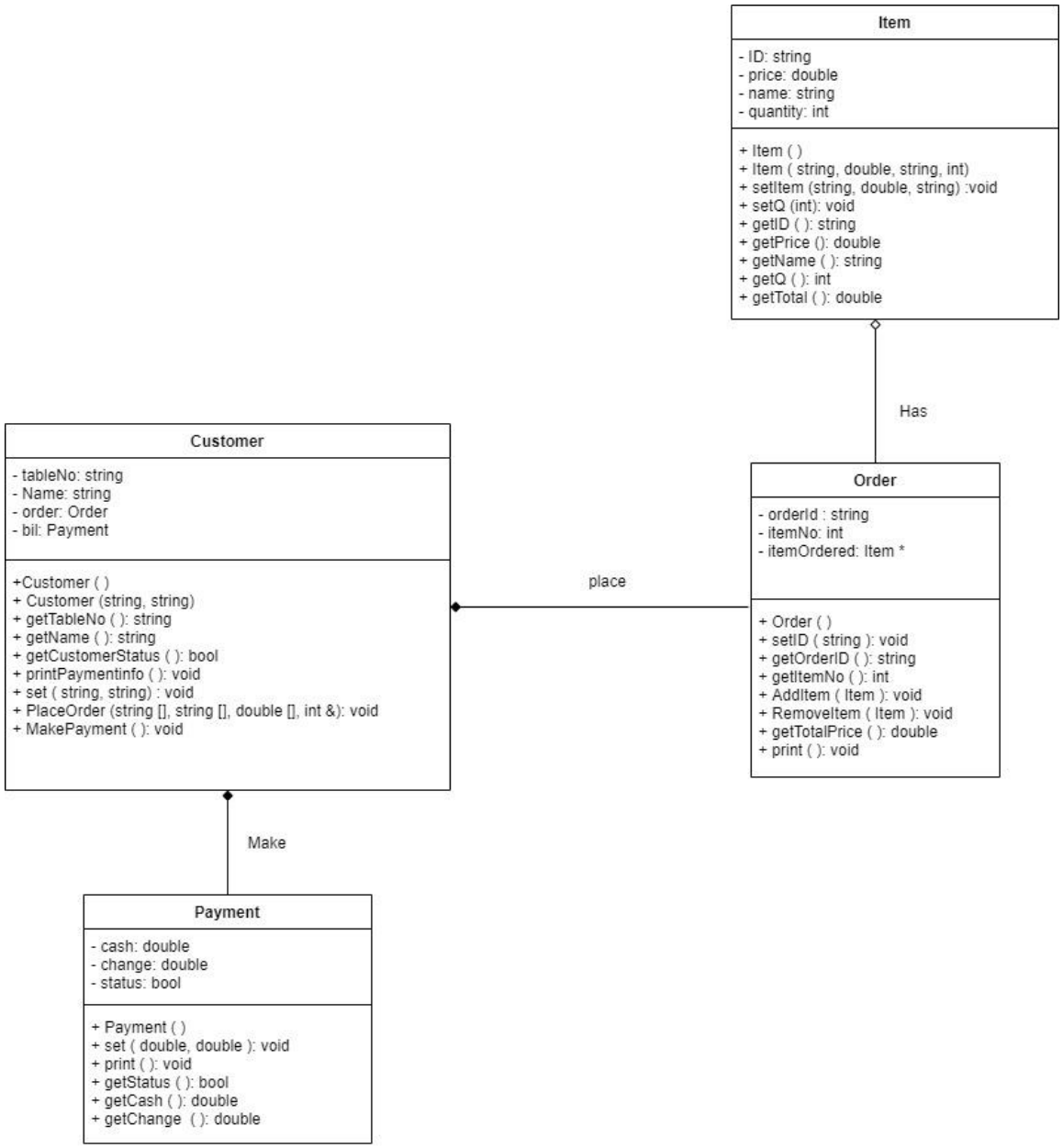

Figure 2.2: UML Diagram of Food Ordering System

3.0 Appendix

Menu.txt

```
N01 NasiLemak 2.50
C01 ChickenRice 3.50
R01 RotiCanai 1.20
F01 FriedRice 3.50
L01 Laksa 4.50
M02 Milo 2.80
C02 Coffee 2.20
S01 SoyBean 1.70
H01 HerbalTea 1.60
M01 MilkTea 8.80
```

Item.h

```
#ifndef _ITEM_H_
#define _ITEM_H_

#include <string>
using namespace std;

class Item{
      private:
            string ID;
            string name;
            double price;
            int quantity;

      public:
            Item ();
            Item (string, double, string, int);
            void setItem(string, double, string);
            void SetQ(int);
            string getID();
            double getPrice();
            string getName();
            int getQ();
            double getTotal();
};

#endif
```

Item.cpp

```
#include "Item.h"
#include <iostream>
using namespace std;
Item::Item()
{
	ID="";
	price=0;
	name="";
	quantity=0;
}
Item::Item(string w, double x, string y, int z){
	ID=w,
	price=x;
	name=y;
	quantity=z;
}
void Item::setItem(string w, double x, string y ){
	ID = w;
	price = x;
	name = y;

}
void Item::SetQ(int x)
{
	quantity=x;
}
string Item::getID(){
	return ID;
}
double Item::getPrice(){
	return price;
}
string Item::getName(){
	return name;
}
int Item::getQ(){
	return quantity;
}
double Item::getTotal(){
	return price * quantity;
}
```

Order.h

```
#ifndef ORDER_H
#define ORDER_H
#include "Item.h"
using namespace std;

class Order
{
      private:
      string orderId;
      int itemNo;
      Item *itemOrdered;
      public:
            Order ();
            void setID (string);
            string getOrderID();
            int getItemNo ();
            void AddItem (Item);
            void RemoveItem (Item);
            void print ();
            double getTotalPrice ();

};

#endif
```

Order.cpp

```
#include "Order.h"
#include "Item.h"
#include <iostream>
#include <iomanip>
#include <stdlib.h>
using namespace std;

Order::Order()
{
	orderId="";
	itemNo=0;
	itemOrdered=0;

}
void Order::setID(string id)
{
	orderId=id;
}
string Order::getOrderID()
{
	return orderId;
}
void Order::AddItem (Item food)
{
	if (itemNo==0)
		{
			itemOrdered=new Item [1];
			itemOrdered[0]=food;
			itemNo++;
		}
		else// Update the size of array itemOrdered and add new
item into it
		{

			int num=itemNo+1;
			Item temp [num];
			for (int j=0;j<itemNo;j++)
			{
				temp[j]=itemOrdered[j];
			}
			temp[itemNo]=food;
			delete [] itemOrdered;
			itemOrdered=new Item [itemNo+1];
```

```
                    for (int k=0;k<itemNo+1;k++)
                    {
                          itemOrdered[k]=temp[k];
                    }

                    itemNo++;
              }
}

void Order::RemoveItem(Item food)
{
              int index;
              bool validity=false;
              if (itemNo==0)
              {
                    cout<<"Invalid action !!!"<<endl;
                    cout<<"There is no item added in this order."<<endl;
              }
              else// Remove an item from itemOrdered array and update
it's size
              {
                    for (int j=0;j<itemNo;j++)
                    {
                          if (itemOrdered[j].getID()==food.getID())
                          {
                                index=j;
                                validity=true;
                                break;
                          }
                    }

                    if (validity)
                    {
                          Item temp [itemNo-1];
                    for (int p=0;p<index;p++)
                    {
                          temp[p]=itemOrdered[p];
                    }

                    for (int y=index+1;y<itemNo;y++)
                    {
                          temp[y-1]=itemOrdered[y];
                    }
```

```
                        itemNo--;
                        delete [] itemOrdered;

                        itemOrdered=new Item [itemNo];

                        for (int j=0;j<itemNo;j++)
                        {
                              itemOrdered[j]=temp[j];
                        }

                        }

                }
}
int Order::getItemNo()
{
      return itemNo;
}
double Order::getTotalPrice()// Return total price of all items in the order
{
      double totalPrice=0;
      for (int j=0;j<itemNo;j++)
      {
            totalPrice+=itemOrdered[j].getTotal();
      }
      return totalPrice;
}
void Order::print ()
{
      cout<<"Order ID: "<<orderId<<endl;
      cout<<"Order List"<<endl;
      cout<<"----------"<<endl;
      cout<<left<<setw(10)<<"No"<<setw(20)<<"ID"<<setw(40)<<"Name"<<setw(2
0)<<"Unit                Price/RM"<<setw(20)<<"Quantity"<<setw(20)<<"Total
Price/RM"<<endl;

      for (int j=0;j<itemNo;j++)
      {

      cout<<left<<setw(10)<<j+1<<setw(20)<<itemOrdered[j].getID()<<setw(40
)<<itemOrdered[j].getName();
            cout<<left<<setw(20)<<fixed<<setprecision
(2)<<itemOrdered[j].getPrice()<<setw(20)<<itemOrdered[j].getQ();
```

```
	cout<<left<<setw(20)<<fixed<<setprecision(2)<<itemOrdered[j].getTota
l()<<endl;
	}
	cout<<endl<<"Subtotal:
RM"<<fixed<<setprecision(2)<<getTotalPrice()<<endl<<endl;
}
```

Customer.h

```
#ifndef CUSTOMER_H
#define CUSTOMER_H
#include "Payment.h"
#include "Order.h"
#include "Item.h"
#include <iostream>
using namespace std;

class Customer
{
	private :
		string tableNo;
		string Name;
		Order order;
		Payment bill;
	public:
		Customer ();
		Customer (string, string);
		string getTableNo();
		string getName();
		bool getCustomerStatus();
		void printPaymentinfo ();
		void set (string, string);
		void PlaceOrder (string [],string [],double [],int,int& );
		void MakePayment ();
};

#endif
```

Customer.cpp

```
#include "Customer.h"
#include "Payment.h"
#include "Order.h"
#include "Item.h"
#include <iostream>
#include <stdlib.h>
#include <iomanip>
using namespace std;

Customer::Customer()
{
	tableNo="";
	Name="";
}
Customer::Customer(string tn, string n)
{
	tableNo=tn;
	Name=n;
}
string Customer::getTableNo()
{return tableNo;}

string Customer::getName()
{return Name;}

bool Customer::getCustomerStatus()
{
	return bill.getStatus();
}
void Customer::printPaymentinfo ()
{
	cout<<left<<setw(20)<<tableNo<<setw(40)<<Name<<setw(20)<<order.getTo
talPrice()<<setw(20)<<fixed<<setprecision(2)<<bill.getCash()<<setw(20)<<f
ixed<<setprecision(2)<<bill.getChange();
	if (bill.getStatus())
	{
		cout<<left<<setw(10)<<"Paid"<<endl;
	}
	else
	{
		cout<<left<<setw(10)<<"Unpaid"<<endl;
	}
}
```

```
void Customer::set(string tn, string n)
{
      tableNo=tn;
      Name=n;
}

void Customer::PlaceOrder (string menuid [], string menuname [], double
price[],int count,int &customerTotal)
{

                  int itemquantity;
                  string itemid,itemname;
                  double itemprice;
                        string tempid;
                  int operation;
                  cin.ignore();
                  cout<<"Please enter your name and table number"<<endl;
                  cout<<"Name: ";
                  getline(cin,Name);
                  cout<<"Table Number: ";
                  getline(cin,tableNo);
                  customerTotal++;
                  system("CLS");

                  do
                  {
                              cout<<"Choose the below operations"<<endl;
                  cout<<"1. Add item into your order \n2. Remove item from
your order \n3. Return to customer menu"<<endl;
                  cout<<"Choice: ";
                  cin>>operation;
                  system("CLS");
                  if (operation==1)
                  {

                  cout<<"Please place your order based on the menu"<<endl;
                  cout<<left<<setw(20)<<"Item ID"<<setw(40)<<"Item
Name"<<setw(20)<<"Price/RM"<<endl;
                  for(int i=0; i<count; i++){

      cout<<left<<setw(20)<<menuid[i]<<setw(40)<<menuname[i]<<setw(20)<<pr
ice[i]<<endl;
                  }
                  bool loop=true;
```

```
                int index=-1;
                      tempid="OR"+tableNo+"/"+Name;

                      cout<<endl<<"Your order ID: "<<tempid<<endl;
                      order.setID(tempid);
                do
                {

                      int pilihan;
                      cin.ignore();
                Q:
                      cout<<"Item ID: ";
                      getline (cin,itemid);
                      cout<<"Quantity: ";
                      cin>>itemquantity;
                      cin.ignore();
                      for (int j=0;j<count;j++)
                      {
                            if (itemid==menuid[j])
                            {
                                  index=j;
                                  break;
                            }
                      }
                      if (index==-1)
                      {
                            cout<<"Invalid item id"<<endl;
                            goto Q;
                      }

                      itemname=menuname[index];
                      itemprice=price[index];
                      Item temp(itemid,itemprice,itemname,itemquantity);
                      order.AddItem(temp);
                E:
                      cout<<"Do you wish to add another item into your
order list? \n 1. Yes \n 2. No"<<endl;
                      cout<<"Choice: ";
                      cin>>pilihan;
                      if (pilihan==1)
                      {
                            loop=true;

                      }
```

```
                        else if (pilihan==2)
                        {
                              loop=false;
                        }
                        else
                        {
                              cout<<"Invalid option...Please try
again"<<endl;
                              goto E;
                        }

                  }
                  while (loop);
                  system("CLS");
                  order.print();

            }
            else if (operation==2)
            {

                  cout<<"Please choose the item that you wish to remove
from your order"<<endl;
                  order.print();
                  bool loop2=true;
                  int index2;
                  int pilihan;
                  do
                  {
                        cin.ignore();
                        cout<<"Item ID: ";
                        getline (cin,itemid);

                        for (int j=0;j<count;j++)
                        {
                              if (itemid==menuid[j])
                              {
                                    index2=j;
                                    break;
                              }
                        }

                        itemname=menuname[index2];
                        itemprice=price[index2];
                        Item temp(itemid,itemprice,itemname,itemquantity);
```

```
                        order.RemoveItem(temp);
                H:
                        cout<<"Do you wish to remove another item into your
order list? \n 1. Yes \n 2. No"<<endl;
                        cout<<"Choice: ";
                        cin>>pilihan;
                        if (pilihan==1)
                        {
                              loop2=true;

                        }
                        else if (pilihan==2)
                        {
                              loop2=false;
                        }
                        else
                        {
                              cout<<"Invalid option...Please try
again"<<endl;
                              goto H;
                        }

                }
                while (loop2);

                order.print();
          }

          else if (operation==3)
          {
                cout<<"The screen is returning to customer menu"<<endl;
                system("CLS");
          }
                }
                while (operation==1||operation==2);

}

void Customer::MakePayment ()
{
     if (!bill.getStatus()==false)// Checking whether the order has been
made payment or not
     {
          cout<<"You have made payment for this order."<<endl;
```

```
	}
	else
	{


	double total=0,change,cash;
	cout<<"Order ID: "<<order.getOrderID()<<endl;
	cout<<"Total amount payable: RM "<<order.getTotalPrice()<<endl;
	cout<<"This machine can only receive \n1. RM 1\n2. RM 5\n3. RM
10\n4. RM 50\n5. RM 100 \nnotes"<<endl<<endl;

		do
		{

		cout<<"Note inserted: RM";
		cin>>cash;
		if (cash==1||cash==5||cash==10||cash==50||cash==100)
		{
			total=total+cash;
				cout<<"\nTotal amount inserted : RM"<<total<<endl;
		}
		else
		{
				cout<<"Invalid note inserted"<<endl;
		}

		}
		while
((cash!=1||cash!=5||cash!=10||cash!=50||cash!=100)&&(total<order.getTotal
Price()));

	change=total-order.getTotalPrice();
	bill.set(total, change);
	order.print();
	bill.print();
}
}
```

Payment.h

```
#ifndef PAYMENT_H
#define PAYMENT_H

using namespace std;

class Payment
{
      private :
            double cash;
            double change;
            bool status;
      public:
            Payment ();
            void set (double,double);
            bool getStatus();
            double getCash();
            double getChange ();
            void print ();

};

#endif
```

Payment.cpp

```
#include "Payment.h"
#include <iostream>
#include <iomanip>
using namespace std;

Payment::Payment()
{
	cash=0;
	change=0;
	status=false;

}

void Payment::print()
{
	cout<<"Cash: "<<"RM "<<fixed<<setprecision(2)<<cash<<endl;
	cout<<"Change: "<<"RM "<<fixed<<setprecision(2)<<change<<endl;
}
bool Payment::getStatus ()
{
	return status;
}
double Payment::getCash()
{
	return cash;
}
double Payment::getChange()
{
	return change;
}
void Payment::set(double cash,double change)
{
	this->cash=cash;
	this->change=change;
	status=true;

}
```

main.cpp

```
#include <iostream>
#include <iomanip>
#include <stdlib.h>
#include <fstream>
#include <string>
#include "Order.h"
#include "Item.h"
#include "Customer.h"
#include "Payment.h"
using namespace std;

string user_interface = "Welcome to food ordering system. ";
string category_interface="Please choose your category.\n1. Staff\n2. Customer\n";
string customer_menu = "1 (View menu) \n2 (Place Order) \n3 (Make Payment)\n4 (Return to user menu)";
string menu_sorting = "1 (Sort ascendingly based on alphabet) \n2 (Sort descendingly based on alphabet) \n3 (Sort ascendingly based on price) \n4 (Sort descendingly based on price)";
int maxsize=200;




int search(string array[],string key, int size)// sequence searching
{
	int index=-1;
	for (int k=0;k<size;k++)
	{
		if (key==array[k])
		{
			index=k;
			break;
		}
	}

	return index;
}

void swap(double* a, double* b)   // swap for price
```

```
{
	double t = *a;
	*a = *b;
	*b = t;
}

void swap(string* a, string* b)  // swap for code
{
	string t = *a;
	*a = *b;
	*b = t;
}

int partition (double arr[],string n[], string c[], int low, int high,int
choice) //partition for price
{
	double pivot = arr[high];
	int i = (low - 1);

	for (int j = low; j <= high- 1; j++)
	{
		if ((arr[j] <= pivot && choice ==3)|| (arr[j] >= pivot && choice
==4))
		{
			i++;
			swap(&arr[i], &arr[j]);
			swap(&n[i],&n[j]);
			swap(&c[i],&c[j]);
		}
	}
	swap(&arr[i + 1], &arr[high]);
	swap(&n[i + 1], &n[high]);
	swap(&c[i + 1], &c[high]);
	return (i + 1);
}

void quickSort(double arr[],string n[], string c[], int low, int high,int
choice)  //quickSort for price
{
	if (low < high)
	{
```

```
            int pi = partition(arr,n,c, low, high,choice);
            quickSort(arr,n,c,low, pi - 1,choice);
            quickSort(arr,n,c,pi + 1, high,choice);
      }
}

int partition2 (double arr[],string n[], string c[], int low, int high,int
choice) //partition for code
{
      string pivot = c[high];
      int i = (low - 1);

      for (int j = low; j <= high- 1; j++)
      {
            if ((c[j] <= pivot && choice ==1)|| (c[j] >= pivot && choice
==2))
            {
                  i++;
                  swap(&arr[i], &arr[j]);
                  swap(&n[i],&n[j]);
                  swap(&c[i],&c[j]);
            }
      }
      swap(&arr[i + 1], &arr[high]);
      swap(&n[i + 1], &n[high]);
      swap(&c[i + 1], &c[high]);
      return (i + 1);
}

void quickSort2(double arr[],string n[], string c[], int low, int high,int
choice) //quick sort for code
{
      if (low < high)
      {
            int pi = partition2(arr,n,c, low, high,choice);
            quickSort2(arr,n,c,low, pi - 1,choice);
            quickSort2(arr,n,c,pi + 1, high,choice);
      }
}
```

```
int main(){

	int category;
	bool loopcategory=true;
	Customer list[maxsize];
	int totalcustomer=0;
	fstream infile;
	cout<<user_interface<<endl<<endl;
	system ("PAUSE");
	system("CLS");
	int count=0;
	infile.open("menu.txt",ios::in);

	string msg;
	if (!infile)
	{
		cout<<"The input file does not exist."<<endl;
	}
	else
	{

	// Get the number of items in the menu
	while (getline(infile,msg)){
		count++;
	}

	infile.close();

	infile.open("menu.txt",ios::in);

	string code [count];
	string name[count];
	double price [count];

	for(int i=0; i<count; i++){
		infile>>code[i]>>name[i]>>price[i];
	}
	infile.close();
	do
	{
```

```
	cout<<"USER MENU"<<endl;
	cout<<category_interface<<endl;
	cout<<"Or\n3. Exit program"<<endl;
	cout<<"Option: ";
	cin>>category;
	cin.ignore();
	system("CLS");
	if (category==1)
	{
		int staff_operation;
		do
		{
			cout<<"1. View customer's payment\n2. Return to user
menu"<<endl;
			cout<<"Option: ";
			cin>>staff_operation;
			system("CLS");
			if (staff_operation==1)
			{
				cout<<left<<setw(20)<<"Table
Number"<<setw(40)<<"Customer's                    Name"<<setw(20)<<"Total
Amount/RM"<<setw(20)<<"Cash/RM"<<setw(20)<<"Change/RM"<<setw(10)<<"Status
"<<endl;

				if (totalcustomer==0)
				{
					cout<<"There is no customer yet."<<endl;
				}
				else
				{
				for (int j=0;j<totalcustomer;j++)
				{

					list[j].printPaymentinfo();
				}
				}

			}
		else if (staff_operation==2)
		{
			cout<<"Returning to user menu....."<<endl;
```

```
                  system("CLS");
            }
      }
      while (staff_operation!=2);


      }
      else if (category==2)
      {



      A:
            cout<<"CUSTOMER MENU"<<endl;
            cout<<"-------------"<<endl;
            cout<<customer_menu<<endl;
            int command;
            cout<<"Insert your command: ";
            cin>>command;
            system("CLS");
            while (command == 1){
                  cout<<"Food Menu"<<endl;
                  cout<<left<<setw(20)<<"Item          ID"<<setw(40)<<"Item
Name"<<setw(20)<<"Price/RM"<<endl;
                  for(int i=0; i<count; i++){

      cout<<left<<setw(20)<<code[i]<<setw(40)<<name[i]<<setw(20)<<price[i]
<<endl;
                  }

                  cout<<"Do you wish to sort the menu? Press 1 for YES and
Press 2 for NO: ";
                  int sort_menu;
                  cin>>sort_menu;
                  if (sort_menu == 1){
                        cout<< menu_sorting <<endl;
                        int sort_option;
                        cout<<"Please choose your sorting preference: ";
                        cin>>sort_option;
                        if (sort_option == 3 || sort_option ==4)
                              quickSort(price,   name,   code,   0,   count-
```

```
1,sort_option);

                    else if (sort_option == 1 || sort_option == 2)
                         quickSort2(price,   name,   code,   0,   count-
1,sort_option);
          }

          else {
               cout<<"Press 1 to continue to view menu and 2 to return to
customer Menu : ";
                    cin>>command;
                    if (command==2)
                    {
                         system("CLS");
                         goto A;

                    }

               }
               system("CLS");
          }
          while (command==2)
          {

     list[totalcustomer].PlaceOrder(code,name,price,count,totalcustomer);
               goto A;
          }
          while (command==3)
          {
                    int option,index_match,operation2;
                    string key;
                    string tempArray[totalcustomer];
               V:
                    cout<<"1. (Search by table number)\n2. (Search by
customer's name)"<<endl;
                    cout<<"Option: ";
                    cin>>option;
                    cin.ignore();
               if (option==1)
                    {
```

```
                              cout<<"Please enter your table number"<<endl;
                              cout<<"Table No: ";
                              getline(cin,key);
                              system("CLS");
                              for (int k=0;k<totalcustomer;k++ )// Copy all
the table Number from list array into tempArray
                              {
                                    tempArray[k]=list[k].getTableNo();
                              }

      index_match=search(tempArray,key,totalcustomer);

                              if (index_match==-1)
                              {
                                    cout<<"No  result  is  found.\nPlease  try
again"<<endl;
                                    goto V;
                              }
                              else
                              {
                                    list[index_match].MakePayment();
                                    cout<<"Do you wish to"<<endl;
                                   cout<<"1. Make another payment"<<endl<<"2.
Return to customer menu"<<endl;
                                    cout<<"Option: ";
                                    cin>>operation2;
                                    system("CLS");
                                    if (operation2==1)
                                    {
                                          goto V;
                                    }
                                    else
                                    {
                                          goto A;
                                    }
                              }
                        }
      else if (option==2)
      {

                  cout<<"Please enter your name"<<endl;
```

```
                              cout<<"Name: ";
                              getline(cin,key);
                              for (int k=0;k<totalcustomer;k++ )// Copy all
the customer name from list array into tempArray
                              {
                                    tempArray[k]=list[k].getName();
                              }

      index_match=search(tempArray,key,totalcustomer);

                              if (index_match==-1)
                              {
                                    cout<<"No  result  is  found.\nPlease  try
again"<<endl;
                                    goto V ;
                              }
                              else
                              {
                                    list[index_match].MakePayment();
                                    cout<<"Do you wish to"<<endl;
                                   cout<<"1. Make another payment"<<endl<<"2.
Return to customer menu"<<endl;
                                    cout<<"Option: ";
                                    cin>>operation2;
                                    system("CLS");
                                    if (operation2==1)
                                    {

                                          goto V;
                                    }
                                    else
                                    {
                                          goto A;
                                    }
                              }
      }
}
                  if (command==4)
                  {
                        cout<<"Returning to user menu....."<<endl;
                        system("CLS");
```

```
                }

}
else if (category==3)
{
      loopcategory=false;
}
else
{
      cout<<"Invalid input.....Please try again"<<endl<<endl;
      system("PAUSE");
      system("CLS");
}
}
while (loopcategory);
}
      return 0;
}
```