**自助订餐系统**

**一．实验题目**

请设计一个点餐系统。程序运行后，首先选择用户类型，然后从文本文件读入菜单信息

进行初始化，根据用户类型不同在控制台界面上提供不同功能选择。用户选择某项功能

后，根据提示进行操作；操作完成后，能返回功能选择菜单重新选择，直至用户退出。

客户功能：

1) 预订——打印出可选菜单，提示用户选择；在用户选择后，提示用户输入个人

信息，记录下信息，并保存至本地文件。

2) 查询/退订——显示所有的已订的订单列表，提示输入退订订单；用户输入后，

如果卖家尚未确认订单，则成功退订并修改本地文件；否则提示退订失败。

卖家功能：

1) 添加/删除菜单——菜单根据文件初始化后，卖家可以对其进行修改，包括添加

和删除菜式等，修改后将新菜单保存至文件，下次初始化仍可用。

2) 查询/修改订单——读取本地文件，显示所有的订单及其状态，提示确认订单或

不进行操作；卖家选择订单后，修改该订单为确认状态。

**二．设计过程与讨论**

1.设计思路

作为一个自助订餐系统，除了实现基本功能，还要能提供友好的用户界面，有良好的提示语句，做到不错的用户体验，并处理大部分的用户错误输入。

菜单：包含编号，菜名，价格。

订单：包含菜单，用户姓名，用户电话，用户地址，数量，订单日期，订单状态。

首先考虑买家功能。买家的类里存着姓名，地址，电话，点餐数量，订餐日期，以及所有该用户的订单。

预订：如果没有可选菜单，输出提示语句。否则，输出所有菜单。提示用户输入选择。若输入的菜单信息无效，则提示找不到菜单，请用户重新输入。若有效，测提示用户继续输入，或者用户选择下单结束。点完所有菜时，将所有用户刚点的菜打印出来，提示用户输入个人信息，包括姓名，电话，地址，以及订单的日期。完毕后，将订单保存到本地文件。然后返回上一层。

查询：若用户之前输入过姓名，则直接输出所有已下订单。否则，提示用户输入姓名，并根据姓名在订单里查找，然后输出，返回上一层。

退订：若该用户没有任何订单，则输出提示并返回上一层。否则，输出所有该用户已下订单的具体情况，并提示用户输入想要取消的订单，若错误输入，提示用户继续输入。否则判断该订单是否已确认，已确认则提示用户订单不可修改，否则输出提示语句，取消订单。然后将新的订单写入本地文件。

然后考虑卖家功能。

进入卖家菜单，需要输入密码。密码保存在本地文件，默认为888888。当用户输入正确密码，才可以进入系统。输入错误5次会自动退出。进入后，有5个功能供选择。

添加菜单。提示用户输入菜式编号，菜名，价格，可以继续添加，然后选择退出。程序加添加后的菜单保存至本地文件。

删除菜单。打印出所有菜单，提示用户输入编号，输入错误则提示无此菜式。否则提示菜式已删除。然后用户可以选择继续删除或返回上一层。

查询订单。若无订单，则提示没有订单。否则，输出所有订单的详细信息。

修改订单。若没有未确认的订单，则提示所有订单已确认。否则输出所有未确认的订单，并为每一个订单规定一个编号。用户可以选择一键确认或者逐一确认。一键确认则程序修改所有订单状态并保存至本地文件。逐一确认则由用户输入编号确认，最后在用户放回上一层的时候保存至本地文件。

修改密码。先由用户输入旧密码，正确之后提示输入新密码。新密码需输入两次，若两次都一样，则修改密码并保存至本地文件。

类设计图：

|  |  |
| --- | --- |
| **Menu** | |
| -  -  - | dishName: string  dishID: string  price: double |
| +  +  +  +  +  +  +  +  +  +  + | Menu()  Menu(string dishName,string dishID,double price)  setDishName(string): void  setDishID(string): void  setPrice(double): void  setDishCount(int): void  getDishName(): string  getDishID(): string  getPrice(): double  getDishCount(): int  ~Menu() |

|  |  |
| --- | --- |
| Date | |
| -  -  -  - | year: int  month: int  day: int  hour: int |
| +  +  +  +  +  +  +  +  +  +  + | Date()  Date(int y ,int m ,int d ,int h )  operator ==(Date date2): bool  getYear(): int  getMonth(): int  getDay(): int  getHour():int  setYear(int): void  setMonth(int): void  setDay(int): void  setHour(int): void |

|  |  |
| --- | --- |
| Order | |
| -  -  -  -  -  - | customerName: string  address: string  phone: string  num: int  modify: bool  bookDate: Date |
| +  +  +  + +  + +  +  +  +  +  +  +  +  +  +  + | Order()  Order(string dishName,string dishID,double price,string customerName,string adress,string phone,Date bookDate)  Order(string dishName,string dishID,double price)  Order(Menu,int)  const bool operator==(Order)  getCustomerName(): string  getAdress(): string  getPhone(): string  getBookDate(): Date  getModify(): bool  getNum(): int  setCustomerName(string customerName): void  setAdress(string adress): bool  setPhone(string phone): bool  setnum(int): bool  setBookDate(Date bookDate): bool  setModify(): void |

|  |  |
| --- | --- |
| User | |
| +  + | modifyOrder(vector<Order>&)=0: virtual void  inquireOrder(vector<Order>&)=0: virtual void |

|  |  |
| --- | --- |
| Buyer | |
| -  -  -  -  -  - | myOrder: vector<Order>  name: string  adress: sting  phone: string  bookDate: Date  num: int |
| +  +  +  +  +  +  + | Buyer()  bookOrder(vector<Menu>&,vector<Order>&);  modifyOrder(vector<Order>&): void  inquireOrder(vector<Order>&): void  getNum(): int  getName(): string  setNum(int): void |

|  |  |
| --- | --- |
| Seller | |
| - | passord : string |
| +  +  +  +  +  +  + | Seller()  modifyOrder(vector<Order>&): void  inquireOrder(vector<Order>&): void  appendMenu(vector<Menu>&): void  cancelMenu(vector<Menu>&): void  setPassword(string): void  getPassword(): string |

主要流程图：

Main函数流程图：

开始

用户选择

**1 2**

判断输入

**3**

用户输入密码

结束

创建买家对象