Assignment 2

Due: Sunday, 29 April 2018, 11:59 PM on TEACH as a .tar

Classes with Pizza

This program is going to simulate some of the common functionality of a pizza ordering site. The program will allow customers and employees to interact with the site. The program will operate via command line and begin by prompting for which user will be interacting with the system. If the user is an employee, they need to successfully log in to be able to change aspects of the site. Upon successful log in, employees should be provided the option to change the restaurant hours, add or remove items from the menu, or view or remove orders. If the user is a customer, they should be provided with the option to search the menu by price or by ingredients or place an order. When placing an order, a customer may select off the entire menu or from the last search result they examined. Both kinds of users should have the option to view the menu, the restaurant hours, address or phone number. Users may also log out at any time which will return them to the welcome prompt. From the welcome prompt they can end the program or log in as another user.

To save time on this assignment, the restaurant information, menu, and employees of the restaurant will be read in from files. Changes to the menu, hours and the orders will be written back out to their respective files. The file formats can be found at the end of this document.

You will implement this program with the following classes and structs: Restaurant, Menu, Pizza, employee, and hours. Partial outlines of the classes and structs are provided at the end of this document to help get you started. **You will need to add more functions.** You may add variables if they are useful and you can defend their addition.

An example user interface and test run is provided below:

```
Welcome to Bytes Pizza!

Are you a customer (C) or employee (E) or would you like to quit (Q)? E

Please provide your ID number: 1111

Please provide your password: Flam

We do not recognize that password. Please try again.

Please provide your password: Flamingo

Welcome C.J. Cregg! What would you like to do?

    1. Change hours
    2. View Orders
    3. Remove Order
    4. Add Item to Menu
    5. Remove Item from Menu
    6.  View Menu
    7.  View Hours
    8.  View Address
    9.  View Phone
    10. Log out

Selection: 1

Which day would you like to change the hours for? F

What should the opening time be? 8

What should the closing time be? 11

Here are the updated hours:

S 12 5
M 10 9
T 10 9
W 10 9
R 10 9
F 8 11
S 10 11

What would you like to do?

    1.  Change hours
    2.  View Orders
    3.  Remove Order
    4.  Add Item to Menu
    5.  Remove Item from Menu
    6.  View Menu
    7.  View Hours
    8.  View Address
    9.  View Phone
    10. Log out

Selection: 10

Welcome to Bytes Pizza!

Are you a customer (C) or employee (E) or would you like to quit (Q)? C

What would you like to do?

    1.  View Menu
    2.  Search by Cost
    3.  Search by Ingredients
    4.  Place Order
    5.  View Hours
    6.  View Address
    7.  View Phone
    8.  Log out

Selection: 3

Would you like to search for ingredients you want include (I) or exclude (E)? I

What item would you like to include? Pepperoni

Do you want to include another item (Yes/No)? Yes

What item would you like to include? Spinach

Do you want to include another item? No

Here are the pizzas we found:

The_Bent 12 25 30 6 Cheese Pepperoni American_Bacon Italian_Sausage Black_Olives
Mushroom
Corvegas 12 23 28 5 Pesto Pepperoni Mozzarella Roasted_Red_Peppers Artichoke
Meat_Lovers 10 18 25 6 Pepperoni Sausage Smoked_Pork_Belly Mozzarella Parm Marinara


Would you like to reduce your search by excluding some ingredients (Yes/No)? No

Would you like to place an order off this search result (Yes/No)? No

 What would you like to do?

    1.  View Menu
    2.  Search by Cost
    3.  Search by Ingredients
    4.  Place Order
    5.  View Hours
    6.  View Address
    7.  View Phone
    8.  Log out

Selection: 8

Welcome to Bytes Pizza!

Are you a customer (C) or employee (E) or would you like to quit (Q)? Q
```

(90 pts) Implementation Details

- Each class needs accessor functions, mutator functions, constructors, copy constructor, destructors, assignment operator overload, and use of const where appropriate.
- No memory leaks are permitted.
- All functions are to be under 20 lines including main.
- The program needs to be compiled with a Makefile.
- Files must be separated into a .h and a .cpp for each class. Both structs may occur in the same .h You should have one driver file.
- Text files will be provided for demos. You may use the examples ones provided in this document to test your program.
- The program satisfies all the requirements in the above problem statement as well as in the listed implementation details.
- If an employee provides incorrect login information, the system should reprompt until they get it right or elect to quit.
- Users should be provided their menu of choices after each completed task.
- Logging out ends the program.
- When text files are edited, they should be replaced with their newest version. At the completion of your program there should only be four .txt files. See the Wednesday lecture of Week 2 for examples on how to do this.
- Users should have the option of refining their search when they search by ingredients. This means if they first search for ingredients to include on the pizza, then they may further reduce these results by providing a list of ingredients to not include. Users should then be able to place an order after completing this search.
- Users should be able to place an order after completing the cost search. The program does not need to refine the search after cost.
- If the user does not place an order off of a search result than the default behavior should be to order off the entire menu.

(10 pts) **Program Style/Comments**
In your implementation, make sure that you include a program header in your program, in addition to proper indentation/spacing and other comments! Below is an example header to include. Make sure you review the style guidelines for this class, and begin trying to follow them, i.e. don't align everything on the left or put everything on one line!
http://classes.engr.oregonstate.edu/eecs/spring2018/cs162-001/assignments/162_style_guideline.pdf

```
/*********************************************************
** Program: run_stats.cpp
** Author: Your Name
** Date: 04/08/2017
** Description:
** Input:
** Output:
*********************************************************/
```

Electronically submit your C++ program (**.h, .cpp, and Makefile files**, not your executable) and your test files as a tarred archive by the assignment due date, using TEACH.

You must tar these files together using the following command:

```
tar –cvf assign2.tar <list all of the .h and .cpp files> Makefile
```

****NOTE:** The easiest way to upload your program from ENGR to TEACH is to map a network drive to your home directory on ENGR. Mac or Windows, See: http://engineering.oregonstate.edu/computing/fileaccess/

File Formats

menu.txt (click for example)

Pizza_name Small_Cost Medium_Cost Large_Cost Num_Ingredients Ingredients …

…<Repeats n times>…

employee.txt (click for example)

ID First_Name Last_Name Password

…<Repeats n times>…

restaurant_info.txt (click for example)

Name

Number

Address

Num_Days_Open

Day Open_time Close_time

…<Repeats Num_Days_Open>…

orders.txt (click for example)

Order_Num Customer_Name Customer_CC Delivery_Address Customer_Phone Pizza_name size quantity <repeat for each distinct pizza on order>

Structs and Classes

```
struct employee {
        string id;
        string first_name;
        string last_name;
        string password;
};
struct hours {
        string day;
        string open_hour;
        string close_hour;
};
class Pizza {
        private:
                string name;
                int small_cost;
                int medium_cost;
                int large_cost;
                int num_ingredients;
                string* ingredients;
        public:
                //need to include accessor functions and mutator functions for each private
member
                //need to include constructors, copy constructors, assignment operator overload,
                //and destructors where appropriate
};
class Menu {
        private:
                int num_pizzas;
                Pizza* pizzas;
        public:
                //need to include accessor functions and mutator functions for each private
member
```

```
        //need to include constructors, copy constructors, assignment operator overload,
        //and destructors where appropriate
        Menu search_pizza_by_cost(int upper_bound, string size);
        Menu search_pizza_by_ingredients_to_include(string* ingredients, int
num_ingredients);
        Menu search_pizza_by_ingredients_to_exclude(string* ingredients, int
num_ingredients);
        void add_to_menu(Pizza pizza_to_add);
        void remove_from_menu(int index_of_pizza_on_menu);
};
class Restaurant {
    private:
        Menu menu;
        employee* employees;
        hours* week;
        string name;
        string phone;
        string address;
    public:
        //need to include accessor functions and mutator functions for each private
member
        //need to include constructors, copy constructors, assignment operator overload,
        //and destructors where appropriate
        void load_data(); //reads from files to correctly populate menu, employees, hours,
etc.
        bool login(string id, string password);
        void view_menu();
        void view_hours();
        void view_address();
        void view_phone();
```

```
		void search_menu_by_price();
		void search_by_ingredients();
		void place_order(Pizza* selection);
		void change_hours();
		void add_to_menu();
		void remove_from_menu();
		void view_orders();
		void remove_orders();
};
```