ASSIGNMENT NO 1

Title: Pharmacy Management System

Made By Abdullah Usman

Submitted to Dr. Rabeeh A. Abbasi

Overview:

In the First Part of Assignment, I designed a Pharmacy management system which inputs and stores the records of medicines such as Tablets, Ointments, Syrups entered by the user. User can also update any record and delete some record. He can import the record from csv file and export all the record to some a csv file.

Implementation:

I created the assignment using Object Oriented approach. Three structures

- **Tablet**
- **Ointment**
- **Syrup**

I gave these structures some data members such as

- **Code** (This is a unique id assigned to every item record)
- **Name** (a character array of size 30)
- **Potency** (for Tablets) / **Quantity** (for Ointments/ Syrups)
- **Stock** (which represents the available no of items)

After that I created a class Named **Pharmacy** which private numbers are:

- **Tablet * tablets**
- **Ointment *ointments**
- **Syrup * syrups**

The dynamic array of different structs to store the data easily.

The Public members of **Pharmacy** class are:

- **Pharmacy();**
- **~Pharmacy();**
- **void input_Tablets(Tablet*)**
- **void input_Ointments(Ointment*)**
- **void input_Syrups(Syrup*)**
- **void search(bool,bool)**
- **void update_tab(Tablet &)**
- **void update_oint(Ointment&)**
- **void update_syrup(Syrup&)**
- **bool code_validate(int)**
- **void write_code(int)**
- **void del_code(int)**
- **void del_tab(Tablet*,int&,int)**
- **void del_syp(Syrup*,int&,int)**
- **void del_oint(Ointment*,int&,int)**
- **void display_all()**

I overloaded the extraction operator (**<<**) to take input from the user

- **friend std::istream& operator>>(std::istream&, Pharmacy&)**

The Import and Export function are also written but I made them non member

- **void Export()**
- **void Import(const std::string &)**

Explanation of Code & Working:

All the Data input by the user is stored in the binary files. I have separate binary files for Tablets, Ointments, Syrups. When the program runs it displays the following menu:

Add New Items [A/a]:

Search Any Item [S/s]:

Update Any Item [U/u]:

Delete Any Item [D/d]:

Print all Data: [P/p]:

Import Data [I/i]:

Export Data [E/e]:

To Terminate [T/t]:

Please Enter Your Choice:

It will ask you to input your choice and if the choice input by user is invalid it will display again until the character **[T/t]** is entered.

- **Add New Items [A/a]:**

When user choose this it will display another menu to him/her asking for the product he/she wants to enter.

What to want to Enter:

Tablets [T/t]:

Ointments [O/o]:

Syrups [S/s]:

Two Items [2]:

All Items [A/a]:

Exit [E/e]:

Working of (>>) operator:

The extraction operator will take input from the user according to the choice and I have wrote multiple if statements in order to manage the choices. Inside that if statements I am calling the input functions. For example the user choose [T / t] to input Tablets I am calling the input_tablets() function. So, all the other options are written this way/

- **Working of input functions (input_Tablets(Tablet*), input_Ointments(Ointment*), input_Syrup(Syrup*))**

  First, I opened the file **(Tablets.dat)** in input mode and if it is not empty and read the size of Tablet array from that file and allocate the memory to new array and reads all the data

from file. After that ask the user to input data. When the User inputs the data I opened the file again in writing mode and writes the new size, previous data (which is being stored in an array) and new data entered by the user. Other two functions follows the same logic.

- **Search Any Item [S/s]:**

This option allows the user to search any record and program will display its information to user if the record does not exist it will simply terminates with a message 'Record Not Found'

It Also display a menu that ask user the item that he wants to search is of what type

Tablet [T/t]:

Ointment [O/o]:

Syrup [S/s]:

Again program will ask the user if he/she wants to search the item by its Name or by its Code?

To Search By Tablet's Code Press [C/c]:

To Search By Tablet's Name Press [N/n]:

- Working of Search function (**search()**):

The program traverse through the array and compares the code data member of Tablet array with code input by the user. If found it displays the details. In case of name it does the same but instead of comparing the code it compares the data member name with the name entered by the user.

- **Update Any Item [U/u]:**

The Update function asks the user what type of medicine you want to update. Then it asks for which data member he wants to update.

Name [N/n]:

Potency [P/p]:

Stock [S/s]:

- **Working of Update function (update())**

Basically the update function is implemented inside the search function in search function parameters default value is set zero. When we have to call the update just need to set the value to true. Now In search function function when the value is found . The related update function is called for when if the user wants to update Tablet type data **updade_tablet()** function will be called which takes Tablet type array as a parameter. And updates the data members accordingly and also updates it in the according file.

- **Delete Any Item [D/d]:**

Functionality of Delete and Update are mostly same. It is also called inside search function. By default its value is false (0).

Its Woking:

To delete the record inside an array I use a very basic approach I passed the array ,its size and index of the where the element to be deleted is present then I start the loop from that index and overwrites its value with the element most right to it in array until the (size-1) array in finished. Then decrement the size of array (which I passed in func argumnent by reference) after that back to search function I update the according file.

- **Print all Data: [P/p]:**

For this part I defined the **display_all()** function which loads the data from the binary files and simply displays them.

- **Import Data [I/i]:**

In this part I use the **import()** function which takes the filename/file Path and if the file path is correct then I take a string, read the string, and assign the values accordingly until the last line.

- **Export Data [E/e]:**

In this part I opened the all my Binary files and loads their data in temporary struct type arrays and then writes all of that data in binary file name **Pharamcy_Data.csv**.

- **<u>Handling Duplicate Codes</u>**

Users may input the same code for multiple records to handle this situation. I have created two functions **write_code()** & **code_validate()** both takes int as a parameter.

write_code(int code):

in constructor I have created an std::fstream object in which I created the file **Productcodes.dat.** I wrapped these input code statements in do while and in while condition I call the **code_validate()** function. If the code already exists it will ask the user to input again. If the code is unique call **code_write()** function to save it in **Productcodes.dat** file.

Implementation of both Functions…in **write_code()** I open the file as std::ios::in checks if there is any previous data. If there is store them in temporary array and then open again std::ios::out reads the array again with new entered data with value of size incremented by one. In **code_validate()** load the entire data in temp array and compare each previous code with the current code entered by the user.