16:332:503 Project: Account Management System

1 INTRODUCTION

For this project I have used an inheritance structure for the classes used in the program. I have used five classes apart from main function. There are a total of three header files containing Account class, Stock class and Bank account class respectively. The doubly linked list and node classes are included in the .cpp file of stock class. The parent class Account is inherited by its child classes Stock class and Bank account class respectively. The functions defined in Account class are effectively used by both the child classes. The classes Bank and Stock account share a common balance and the stock transactions are printed in bank transaction history as deposit or withdraw cash transactions. The menu is accessed through main function.

2 MAIN BODY

The main body consists of two functions stock menu and bank menu. A switch case has been used to access the menu. Further, as one of the menu is selected, its content is displayed with the help of another switch case. Under every case, the corresponding function call is made to carry out the requested process. The OOP aspect of C++ has been effectively used to achieve this. Error handling and conditional checks have been used throughout the main body and function members to avoid unwanted results.

3 PARENT CLASS: ACCOUNT CLASS

The Account class contains the functions and data members accessible to both its child classes. The *constructor* of the class sets the balance as $10000 or reads balance from file and assigns its value to the variable balance. The *virtual float readfile()* function returns the value of balance to other functions. Here we use the concept of polymorphism. The function *void write_cashbalance(float)* takes balance and writes it to text file Cash_Balance. The function *void history(void)* displays the data from text file Bank_Balance. The function *void writefile(float, float)* writes deposit/withdraw transactions to file Bank_Balance by taking two parameters one to switch between deposit and withdraw; and other takes balance as parameter respectively. This function takes into account the stock transactions as well. These are the functions defined in the Account class which are used by the Bank account and stock account class respectively.

4 CHILD CLASS: BANK ACCOUNT

The Bank account class has three primary functions, namely to display balance, to deposit money and to withdraw balance. The *void display()* function displays the value of current available balance by reading it from a text file. The function *void deposit()* deposits money in the bank account and increases the balance accordingly. The input is verified before making the transaction. The *float readfile()* function returns the

value of balance to other functions. The *void withdraw()* function withdraws money from the bank account and deducts the balance accordingly. One of the conditions checked by this function is that sufficient balance is available for withdrawal. Also, it restricts user from entering invalid inputs. If such a situation arises, user is prompted to re-enter the input.

5 CHILD CLASS: STOCK ACCOUNT

5.1 CLASS NODE

This class is responsible for defining the number of elements in each node and assigning values to the elements of node. This node holds the name of the stock and the number of stocks.

5.2 CLASS DOUBLY LINKED LIST

This class contains functions like adding node to the list, finding a node in the list and editing the elements in the list, sorting the list and many more. This is the most important class of the entire project.

The *constructor* of this class loads values to the elements of the nodes from text file Current_Portfolio. The function void *AddToEnd(Node*)* adds a node to the end of the linked list. The function *void printlist()* prints the data stored in all the nodes of the linked list. The function *void find_string(string, int)* finds the stock name in the list to add or subtract number of stocks. If name is not found then adds to the end of the node. The function *void write_to_file()* writes the entire linked list to Current portfolio text file. The function *void sell_stock(string, int)* finds the stock in the portfolio, subtracts the number of shares; if the residual is zero, deletes the node. The function *bool removeFromStart()* removes the first node from the list. Similarly, the function *bool removeFromEnd()* removes the last node from the list. To delete a node from any intermediate location, we call the function *void node_delete(string)*. The function *void write_sell(string, int)* writes the sell transaction to the file Stock_Transaction_History. Every time the linked list is updated we call the void *sort()* function to arrange the list in descending order of the total value of shares.

5.3 CLASS STOCK ACCOUNT

The stock account class contains four functions namely display price, display portfolio, buy share and sell share. The *float readfile()* function returns the value of balance to other functions. The function *float display_price(string)* displays and returns the price of the stock from two files randomly. The function *void display_portfolio()* reads and displays the current portfolio text file. The function *void buy_shares()* buys the given share after checking various conditions. The function *void sell_share()* sells the given share by calling linked list. The function *void tran_hist()* reads and displays the content of Stock_Transaction_History text file. The function void *write_file()* writes the buy transaction to Stock_Transaction_History text file. The function *int search(string)* finds the name of the stock from the list of available stocks. The function *void write_portfolio_value(float)* writes the value of total portfolio to portfolio_value text file. This file is used to read the total portfolio values to plot the graph on MATLAB. The function *void mat_graph()* reads data and number of data points from the text file portfolio_value, passes the values to MATLAB via array and the corresponding graph of portfolio value pops up when user inputs that choice.