Final Project

Name	ID
Mohamed Moeen AbdelAziz	8737
Omar Adel Abdullah	8746
Amr Attia Abdelraouf	8907
Seif El-din Abd Al-moniem Shaheen	8773
Nada Ibrahim Elmetwally	8770

Description of work

The bank system helps us to do the Banking transactions and to store the data for every person. In this project the user can modify his account information, add a new account, delete existing account, search for the data from account number or his name, withdraw, deposit, transfer money and can print the data sorted by (name, date, and balance) and the last five transactions the user has done.

int main()

in the main function we call the mini menu to ask the user if he wants to login or quit if the user logged in the main function call the load function to load the data from the files and then print the username and the current date above the menu and the main function call the menu and after the user do what he want the main function is responsible for asking him to continue and clear the terminal.

void load_accounts()

We use the function load_accounts to read data from a file named "accounts.txt" and place it into main storage by storing it in an array of structures.

Each customer's data is stored in the file on one line, each information separated by a comma. The types of information written in the line are different, so we must read it in an array of structures.

In the struct, the data type will be chosen according to each piece of information related to the customer's account As follows: customer name, email, bank account number, and phone number in the array of sting .The bank account number and phone number are not read as a number because they can start with zero and they are large numbers, if they are saved as numbers, overflow is possible occur . The balance will read as float number. The as float number. The date will be read as integer numbers.

The function begins by declaring a FILE pointer variable fptr and opening the file "accounts.txt" in read mode. After that the information in the file is printed in its variables in the struct by specifying the place where the printing stops, such as a comma or a hyphen. We take a copy of the structure so that other operations can be performed on it without affecting the basic data.

int mini_menu()

in the mini menu we ask the user to choose one of two choices either login or quit when user selects login he goes to the login function, if he logs in successfully it returns 1 to the main and if he logs in with incorrect information the function goes to a label called mini_menu to repeat the function without using the stack to avoid stack overflow, and if he selects quit the program executes exit(0) which quits the program.

void login()

In the login function, we first take a username and password making sure they don't cause a buffer overflow. We then append the username and password together into one string and compare it with every line in the users.txt file individually. If we don't find a match before reaching the end of the file, an error message is printed, and the user is prompted to try again. If it is found, we return a positive number, calling the load_accounts function.

void menu()

menu function is responsible for asking the user to choose one of 12 choices and we have choices (add, delete, modify, search, advanced search, withdraw, deposit, transfer, report, print, logout, and quit) and in each choice we have a function to execute if the user enters a choice which is not available the function prints not found command and finishes which will lead the user to the main function which will return him again to the menu with cleared terminal.

void print_account(int i)

in print_account function the puts the account number and he searches for its index in another function and gives this index as a parameter in print_account function then this function prints account number, name, email, balance, phone and calls a function called print_date and sends the date struct of this account as an argument which will print the date of opening of this account with switch statement as the month is a number and wanted to print a string then the function finishes and returns to the menu which will lead to the main again.

int ask_to_save()

in this function the program asks the user whether he wants to save or not if the choice is y the function returns 1 in the if condition of the menu and calls the

function save() to save the data in the main array of structs and if the choice is n it returns 0 to the menu and calls the function cancel which returns the old data into the temp array of structs and doesn't record the changed data and if the input is not y or n the function prints that the input is wrong and go to a label called ask_save to ask him again.

void save()

save function opens a file called "accounts.txt" in write mode and puts all the temp array to the main array of structs then prints this data in the file "accounts.txt" and closes the file and prints that the changes are saved to let the user to know that the function worked successfully and finishes to the menu function then to the main function.

void cancel()

cancel function returns the changes that happened in the temp array of structs from the main array of structs and finishes to the menu function then to the main function.

void clear()

this function first detect the system that running the program then runs the command "cls" or "clear" depends on the machine which clears the terminal and gives better clean terminal.

void advanced_search()

In the advanced search function, we seek to make the user prompted of any data of an account that its account name contains the keyword that the user enters in the program.

int search()

The search function is used to verify whether the entered bank account exists among the accounts registered in the bank's files or not. This is done by comparing each string with the bank account number in the struct with the account number entered. If it exists, the function is returned the index and if not, it is returned -1

void add_account()

The add_account function make the user prompted of adding a new account to the database. The function has four arrays of characters: addname, addemail,addmobile. We store the input values of the new account name, email, and mobile in them. We did the same as the validation checks in modify account function. In addition, we generated a random account number to each of the added account and made the proper validation checks to check if this random account number is already in the current accounts in the bank. Finally, after all validation checks we stored the input data in the accounts' array of struct and saved the new account in accounts file.

void modify_account()

in the modify account function, we seek to give the access to the user to modify any modifiable data such as account name, mobile number, and email address.

Also, we have made some validation checks upon the input data.

For example, the length of the modified account name and the available characters in it and some other validation checks upon the mobile number, its length, the first three digits in it and the available characters, in addition we have made the same thing with the input email address. Here is briefly described pseudocode of the function:

void delete()

in the delete function we take the account number from the user and check if the account number is exist and check if the balance is zero and print appropriate message if any error has done, if everything is okay we decrease the account counter which represents the number of accounts and then move the rest of the accounts which right this account to the left to fill the gap.

void_withdraw()

This function takes an account number as input, makes sure the account number the user entered doesn't cause a buffer overflow by inputting a large account number. It then searches for the given account number in the accounts available and it returns a value depending on if it found the account number or not. If it is not found it says that the account number is incorrect. If it is found, we take the account number and form a string with it to open the file containing all previous transactions of this account to append to later. We then take the amount needed to withdraw from the user, confirm the user only entered numbers and not symbols or

letters, then we check if the amount entered is less than or equal to 10,000 which is the maximum amount that can be withdrawn. If it is, we then check that the account has more or equal to the amount we're withdrawing. We then ask the user if he wants to save this transaction or not. If he does, then we subtract the withdrawn balance from his account's balance and append this transaction onto his report file containing all his transactions.

void_deposit()

This function works very similarly to the withdraw function so we're going to state only the differences. Depositing also has a limit of 10,000 so we validate that the amount entered to deposit doesn't exceed 10,000, but unlike withdraw we don't need to check if the existing balance of the account is greater than or equal to the amount entered. Other than this step, everything in deposit operates identical to withdraw.

void_transfer()

Transfer operates a bit differently. Here, we take two account numbers and do validation on them like we did before to prevent buffer overflow. We then search for them in our data and print an error message if either of them is not found. If both account numbers entered are the same, we print an error message as you can't transfer to your account. We then open two files instead of one, one for each account. We read the balance the user wants to transfer and validate it. There is no 10,000 limit on transferring unlike withdrawal or depositing. We check that the account we're transferring from has sufficient balance, and then ask the user if he wants to save the operation. If he does, we save it and append the transaction into both accounts' files.

Algorithms

Sort algorithm

In sorting we used the merge sort algorithm as it is the most efficient way to sort the data, we used this algorithm in Balance, Name and Date

The sort algorithm uses the divide and conquer approach, so we divide the array into subarrays and sort the smaller sub arrays and then merge them sorted.

We first determine the middle index of the array and call the same function tell the array is divided into subarrays with one element which are already sorted.

Then we make another function to merge the two sorted arrays by making two temporary arrays and store the left array in one and the right an another.

Then we have 2 sorted arrays, all we want to do is to compare the elements from the left array with the elements from the right one and increase the counter of the arrays that contain the smaller ones , so if one array is finished we don't have to complete the comparison and we just want to fill the array with the rest elements.

We repeat these steps until the left index is not smaller than the right index.

Search algorithm (Linear Search)

1-declares an integer variable i to use as a loop counter.

2- starts a for loop that iterates from i = 0 to i < account counter. Where account counter the number of accounts in the bank.

3- in the loop, the strcmp function is used to compare the string account_numberin (Account numbers entered by the user) with the account number of index "i" in the acc array.

4- If strcmp function returns zero, the value of i is returned, which represents the index of the matching account.

5- if the account_numberin not found and the loop end, the variable i will be equal –1.

6- Finally, the function returns the value of i, which will be -1 if account_numberin not found.

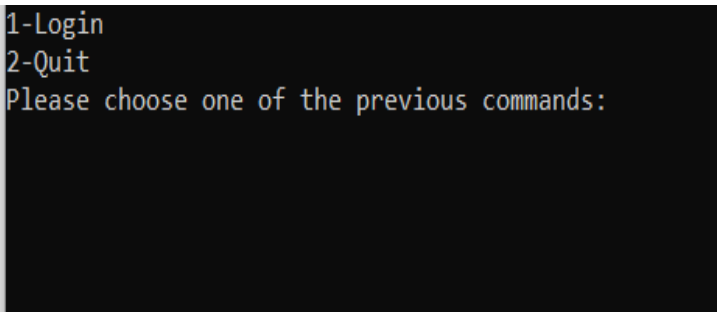
User Manual

introduction

This comprehensive guide is designed to provide you with all the necessary information and instructions to effectively use our banking system. This manual will walk you through the various features and functionalities of our banking system.

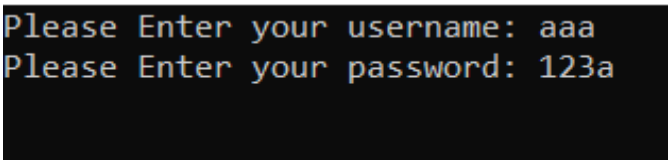
1-Mini menu

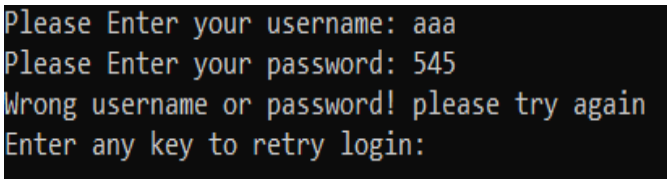
As soon as you open the program, the mini menu shows up, you can choose to login or quit the program.



2-Login

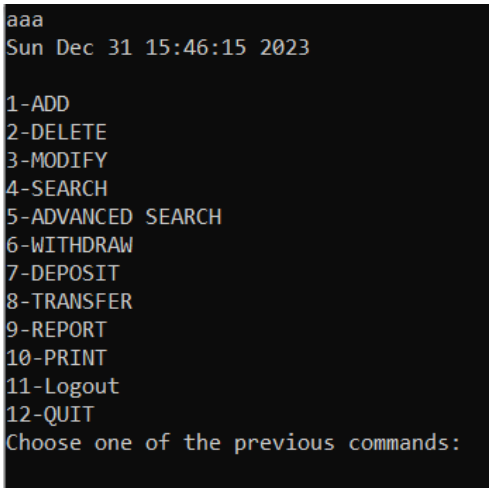
as you enter login, you are required to write your username and password if any errors happened you can write any key to re login.





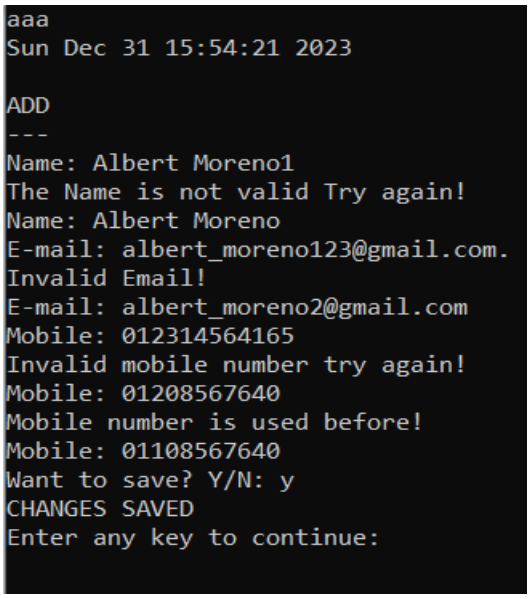
3-Menu

as you login correctly, the menu shows up and you have to choose what you want to do or quit you can add a new account, delete an existing account, modify data for an existing account , search by account number or by name ,withdraw money, deposit, transfer to another account, print a report for the last 5 transactions or print the data of the accounts or to Logout.



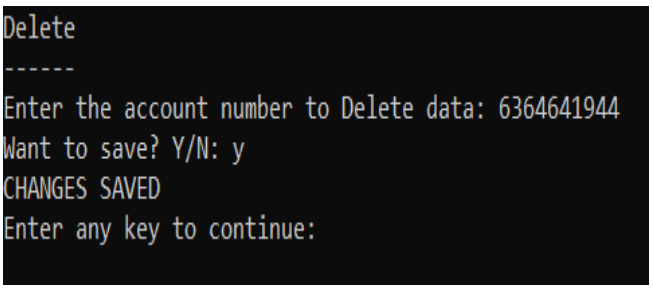
4-add

If you choose to add an account, you are required to enter your name (can't contain numbers and special characters) , E-mail address (you can't put an address that is already exist in the system) and your phone number, if you want to save, you can kindly confirm.



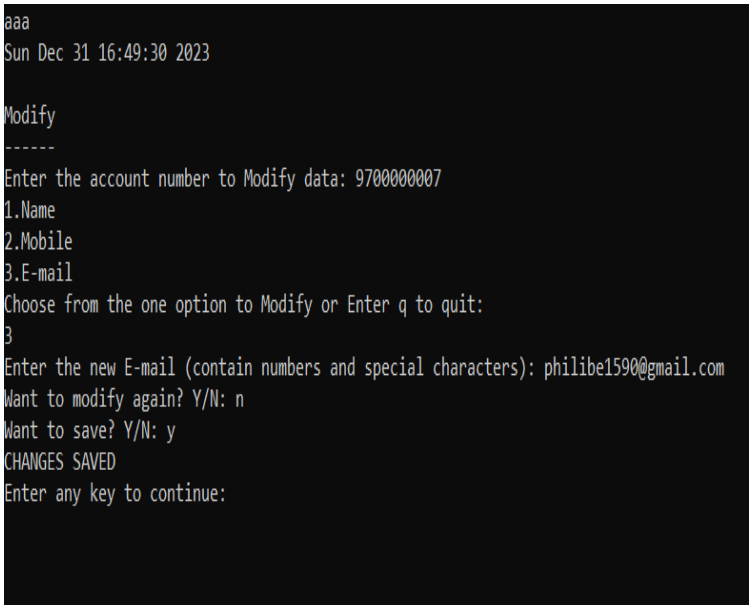
5-delete

To delete an account, you are required to enter an existing account number with 0 balance. You must confirm if you want to save The deletion or not.



6-modify

You can modify an existing account With restrictions in names and mobile Phones and e-mails like adding.
You must confirm if you want to save The modifications or not.



7-searching

If you want to search for an account you can Just write the account number and the all data related to this account will be printed.

```
aaa
Sun Dec 31 17:13:13 2023

Search
------
Enter account number: 9854652313
This account number is not found!
Enter any key to continue:
```

```
aaa
Sun Dec 31 17:09:53 2023

Search
------
Enter account number: 9700000009
Account number: 9700000009
Name: James Adams
E-mail: j.adams@gmail.com
Balance: 250.00
Mobile: 01009700009
Opened: May 2017

Enter any key to continue:
```

8-advanced search

You can also search by entering a name that registered in person and all accounts that contain that name will be printed.

```
aaa
Sun Dec 31 17:16:42 2023

Advanced Search
---------------
Enter Keyword: ahmed
Search Results:
no matches are found
Enter any key to continue:
```

```
aaa
Sun Dec 31 17:14:48 2023

Advanced Search
---------------
Enter Keyword: roberto
Search Results:
Account number: 9700000004
Name: Roberto Thomas
E-mail: rob.thomas@gmail.com
Balance: 400.50
Mobile: 01009700004
Opened: November 2015

Account number: 9700000001
Name: John Roberto
E-mail: j.roberto@outlook.com
Balance: 100.00
Mobile: 01009700001
Opened: December 2008

Enter any key to continue:
```

9-withdraw

You can withdraw money (but with limit 10000) and you absolutely should have sufficient balance All you should do to enter your account number

```
aaa
Sun Dec 31 17:20:06 2023

Withdraw
--------
Enter account number: 970000000
Enter balance to withdraw: 50
Withdrawing balance...

Want to save? Y/N: y
CHANGES SAVED
Enter any key to continue:
```

And the balance. And confirm

if you want to save or not.

```
aaa
Sun Dec 31 17:22:56 2023

Deposit
-------
Enter account number: 6597290464
Enter balance to deposit: 1000
Depositing balance...

Want to save? Y/N: y
CHANGES SAVED
Enter any key to continue:
```

10-deposit

You can withdraw money (but with limit 10000)

All you should do to enter your account number

And the balance. And confirm

if you want to save or not.

11-Transfer

You can transfer balance from an account to another. You should enter the account number that will transfer the money and the account number that will accept the money and the balance to be transferred. And confirm if you want to save or not.

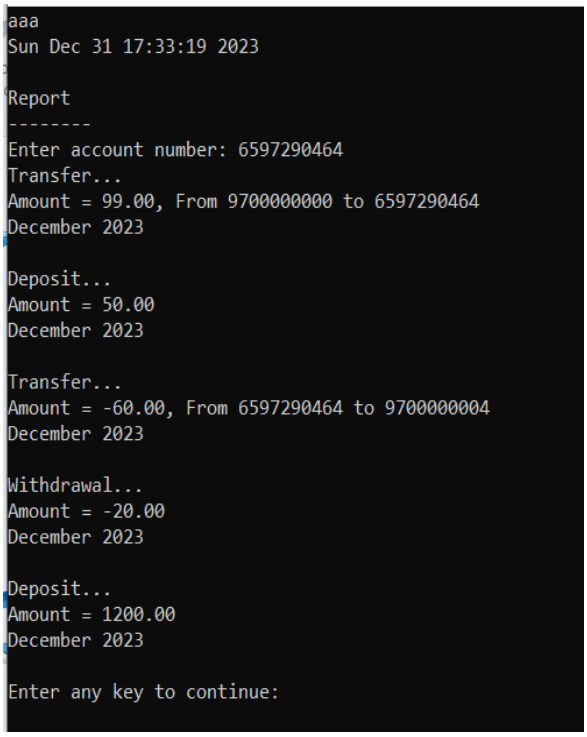
```
aaa
Sun Dec 31 17:26:34 2023

Transfer
--------
Enter account number to transfer from: 9700000000
Enter account number to transfer to: 6597290464
Enter balance to transfer: 99
Transferring balance...

Want to save? Y/N: y
CHANGES SAVED
Enter any key to continue:
```

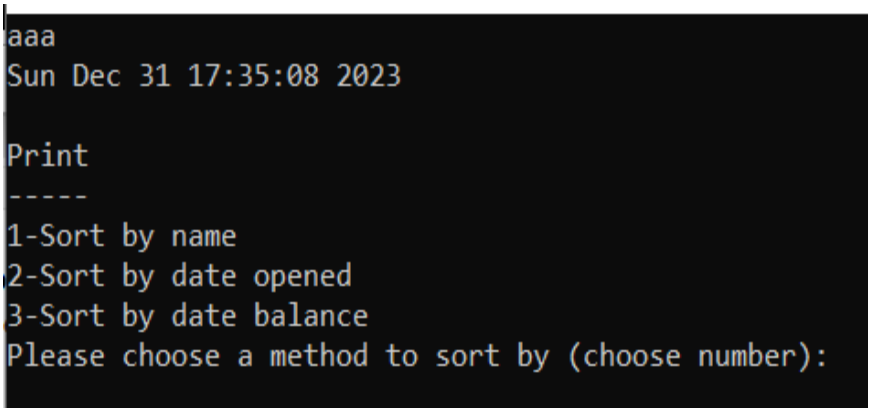
12-report

In the report you can print the last five transactions you just have to enter the account number.



13-print

you can print the data of all accounts sorted by name. (alphabetically),
date opened (oldest to newest) or balance (highest to lowest) by just choose from this menu.



Here is an example of sorting by balance.