DATA STRUCTURE REPORT

Problem Definition:

The main purpose of the code is to design a fully integrated banking system that handles a multitude of functions. Throughout the program, the user will be able to:

- Sign up and sign in for an account
- Store his personal information
- Withdraw and deposit money
- Modify personal information
- Print the transaction history

The code is designed with security in mind, for example, the user cannot modify his password unless he/she types the original password first. This ensures that only the original user can modify the security key. The number of bank accounts that the program can hold is limited to 100 accounts.

ADTs selected:

- Linked list
- Array-Based list

Why the selected ADT:

The Array will be used to store the information of multiple accounts as objects, along with their designated information. A sign-in option is implemented in the program where the user chooses an already created account. Once the user picks the desired account, they will be granted access, and thus allowed to modify the following information: name, password, and balance. The reason behind choosing an array-based list is that it is going to be easier for the program to select the account and adding several accounts will be easier. Due to the nature of arrays, the program has a finite number of user of which it can hold (100 users).

The Linked list will be used to store the transaction history of the actions that the user took regarding the balance, during the program runtime. The linked list is picked for this project because it can store an undetermined number of transactions in chronological order.

The Main Design

Classes

We have one class in our project and it is called "account". This class stores the account information, displays it, modifies it, has functions that withdraw and deposit from the account balance, and a function that displays all the transactions done.

Functions of each class

The Functions of the classes are:

Account Class	What the function does	Group Member
bool operator==(const account_functions other_account);	Assignment operator for the class	Mostafa Mohamed
void set_name(string n);	Takes name as an argument and sets it in the class	Mostafa Mohamed
string get_name() const;	Returns the name	Mohamed Gohary
void set_password(string p);	Takes password as an argument and sets it in the class	Mohamed Gohary
string get_password() const;	Returns the password	Mohamed Absawy
account_functions(double b, string p, string n);	Constructor sets class members	Mohamed Absawy
account_functions();	Default constructor	Mostafa Mohamed
void rand_id();	Generates random id	Mostafa Mohamed
int get_id() const;	Returns id	Mohamed Absawy
void set_balance(double b);	Sets Balance	Mostafa Mohamed
double get_balance() const;	Returns Balance	Mohamed Absawy
void display_balance();	Displays the Balance in a cout function	Mohamed Absawy
void update_name();	Changes the users name	Mohamed Gohary
void update_password();	Changes the users password	Mohamed Gohary
void withdraw();	Subtracts an amount of from balance and displays it	Mohamed Gohary
void deposit();	Adds an amount of from balance and displays it	Mohamed Absawy
void account_info();	Displays all account info	Mostafa Mohamed
void trans_his();	Displays all the actions done to the account	ALL

Array List Class	What the function does	Group Member
const array_list<type>& operator=(const array_list<type>& other_list);	Assignment operator to copy the info to another list	Mostafa Mohamed
bool is_empty() const;	Checks if the list is empty	Mohamed Absawy
bool is_full() const;	Checks if the list is full	Mohamed Gohary
int list_size() const;	Returns length	Mohamed Gohary
int max_array_size() const;	Returns max_size	Mohamed Absawy
void print() const;	Displays list	Mostafa Mohamed
bool is_item_at_equal(int location, const type& item) const;	Checks the value of an item at a certain position	Mohamed Gohary
void insert_at(int location, const type& insert_item);	Inserts item at a certain position	Mostafa Mohamed
void insert_end(const type& insert_item);	Inserts item at the last position	Mohamed Absawy
void remove_at(int location);	Removes item at a certain position	Mohamed Gohary
void retrieve_at(int location, bool& inserted, type& ret_item) const;	Returns item from a certain position	Mostafa Mohamed
void replace_at(int location, const type& rep_item);	Replaces item at a certain position	Mohamed Absawy
void clear_list();	Clears list	Mohamed Gohary
int search(const type& item) const;	Searches for an item	Mohamed Gohary
array_list(int size);	Constructor	Mostafa Mohamed
array_list(const array_list<type>& other_list);	Copy Constructor	Mohamed Absawy
~array_list();	Destructor	Mostafa Mohamed

Linked List Class	**What the function does**	**Group Member**
void copy_list(const linked_list<type>& other_list);	Copies a new list from an old one	Mostafa Mohamed
const linked_list<type>& operator=(const linked_list<type>& other_list);	Assignment operator	Mohamed Gohary
void initialize_list();	Initializes list	Mostafa Mohamed
type front() const;	Returns first element	Mohamed Absawy
type back() const;	Returns last element	Mohamed Gohary
linked_list();	Default constructor	Mohamed Absawy
void destroy_list();	Destroys list	Mostafa Mohamed
virtual void print(int transation_number) = 0;	Prints list with the appropriate transaction number	Mohamed Absawy
bool is_empty_list() const;	Checks if the list is empty	Mohamed Gohary
int length() const;	Returns count	Mostafa Mohamed
bool search(const type& search_item) const;	Searches for an item and returns a flag	Mohamed Absawy
virtual void insert_first(const type& new_item) = 0;	Inserts item at the first position	Mohamed Absawy
virtual void insert_last(const type& new_item) = 0;	Inserts item at the last position	Mohamed Gohary

virtual void delete_node(const type& delete_item) = 0;	Searches and deletes item	Mohamed Gohary
linked_list(const linked_list<type>& other_list);	Copy Constructor	Mostafa Mohamed
~linked_list();	Destructor	Mohamed Gohary

Unordered linked List Class	**What the function does**	**Group Member**
void insert_first(const type& new_item);	Inserts item at the first position	Mohamed Gohary
void insert_last(const type& new_item);	Inserts item at the last position	Mohamed Absawy
void delete_node(const type& delete_item);	Searches and deletes item	Mostafa Mohamed
void print(int transation_number);	Prints list with the appropriate transaction number	Mohamed Absawy

Functions outside the Classes

Functions	**What the function does**	**Group Member**
friend ostream& operator<<(ostream&, const account_fuctions& other_account);	Output Stream operator to print the name variable inside the class.	Mostafa Mohamed
void initialize_list(array_list<account_class>& accounts_list);	Adds several accounts to the array-list	Mohamed Gohary
void create_account(array_list<account_class>& accounts_list);	Creates and adds an account to list of accounts	Mostafa Mohamed
bool password_valid(const account_class);	Checks for password	Mohamed Absawy
void operations_on_account(account_functions);	Does the main operations on the account	Mohamed Absawy/ Mohamed Gohary

Full Program Description

First, the user is asked to choose an option of signing in or signing up. Depending on what is decided, the program does the following:

1- If the user picks signing up, it displays messages to ask him to enter his name, password, and his initial deposit. This info is stored in a new object of type "account". This account is added into the array list.

2- If the user chooses signing up, he is asked to enter his account name and password. The program then assigns the chosen account as the main account, to be able to perform operations. If the input information does not correlate with the information stored within the array list, then the program notifies the user that the account he is trying to access does not exist within the array.

After having selected an account or creating an account, the user is shown 7 operations that could be carried out, including:

1. Change Name
2. Change Password
3. Deposit
4. Withdraw
5. Display Balance
6. Transaction History
7. Display Account Info

Depending on the selected operation, the program carries out the proper commands. For example, if the user chooses to withdraw from his account, the program asks him the amount he wants to withdraw, then asks the user if he wants to do another operation or not. If the user enters a (y) or a (Y), the 7 prompts loop. If the user enters a (n) or a (N), the program ends.

The Role of Each Member

Mohamed Abdelmoneim Elgohary 210210: Unordered Linked List class, Array List Implementation, overloading functions, and the report.

Mostafa Mohamed Ahmed 212577: Unordered Linked List implementation, Array List class, and the account class with its implementation

Mohamed Ahmed Abdel Galeil Absawy 214618: Linked List class with its implementation, the account class, and overloading functions.