The American University in Cairo
Computer Science and Engineering Department

Fundamentals of Computing II
Spring 2020
CSCE110 03/04
Assignment 4

Notes:

- If any program is not compiling (has errors), you may lose marks.
- Submit your source, cpp, and header files along with screenshot(s) of your program running with different samples
- Make sure your program is user friendly, use cout before cin so the user knows what should be entered.
- Make sure you compress all your work (.cpp files and screenshots) in a zipped folder with your name, your id, and the assignment number. For example: Name-ID-Assignment1
- Upload your zipped folder to Blackboard.
- The Assignments should be your own original work. Collaboration with other students or copying answers from online sources is prohibited by the AUC academic integrity code. Any violation of academic integrity will result in receiving a Zero in the assignment or failing the course.

Problem Statement:

It is required to create a small Banking Management System that has at least (you may increase) three main classes:

A. Customer
B. Account
C. ATM Transactions

Let us go through each one of those classes:

Customer:

The Customer class has Customers' main information such as; but not limited to:

- Customer Unique Identification (Commonly known as Customer ID)
- Customer Full Name (Arabic and English)
- Customer Address
- Any other relevant information

Methods:

The Customer class has -but not limited to – the following methods:

METHOD NAME	DESCRIPTION

Search Customer	This method searches for a given Customer by name (whole or partial) , Customer ID, or by Account Number(s)
Add Customer(s)	Adding a given **customer or customers** to the customers' list with his/her accounts and his/her transactions
Delete Customer(s)	Delete a given **customer or customers** to the customers' list along with his/her accounts and his/her transactions

Pre-requisites and assumptions:

- It is a must to use Linked List as a Data Structure to store customers prior any operations undergoing them.
- It is up to you to have one customer's Linked List also including the Customer's accounts and cards or to have one Linked List for the customer and one Linked List for the accounts and one linked list for the cards. Implementing a doubly linked list is a plus for speeding up operations.
- For adding one or more customers based on the user's input, it is up to you to store them in a separate linked list, an array or a vector. Using a vector is a plus.
- It is up to you to implement four methods or to subdivide them into more smaller methods.
- It is expected to have the following on the class:
    - One or more constructors
    - Destructor
    - Overloaded Functions where applicable
    - Loading the customers in memory (if needed to be in a separate function)
- The more you have more fields/methods in the customer's class, the more you are eligible for a bonus
- If you opt to see how this is related to real life example and to enhance you output to get a bonus please look at :
    - https://docplayer.net/11528927-Personal-account-opening-form.html
    - or write Retail Account Opening form

Account:

The Account class has Customer's account(s) information such as -but not limited- to:

- The account number as a string of non repetitive and you can start from "100001"
- Type of account (Current, or Saving)
- Balance
- Card Number associated with the account. Each account may have one or more card numbers (make it as a constant as two cards) per account.

METHOD NAME	DESCRIPTION
Create Account	This method creates an account for a given Customer ID
Show Account(s)	This method shows all the account(s) information (eg. Balance) related to a given Customer ID

Deposit	This method deposits an amount of money from a given account using a card where the customer has executed an ATM transaction
Withdraw	This method withdraws an account of money from a given account using a card where the customer has executed an ATM transaction
Search for my accounts	This method searches for given account(s) that belong to a given Customer's ID and displays the result on the screen

Pre-requisites and assumptions:

- The above methods are the least to implemented where you are free to subdivide the above providing keeping the functionality.
- It is a must to use linked list as a Data Structure to store both Accounts and Cards. Alternatively, you can store one linked list for the accounts and One Linked List for a card.
- It is expected to have the following on the class:
    - One or more constructors
    - Destructor
    - Loading the accounts in memory (if needed to be in a separate function)
- It is a plus to separate the cards in a separate class called CARD.
- It is a plus to have a base class of account and having saving and current accounts as children classes.
- For simplicity, each account is related to one and only one card number.
- The customer can have two accounts only (Current and Saving)

ATM Transaction:

This class simulates the operation of deposit and withdrawal using the customer's card(s) from an ATM. It has -but not limited to- the following fields:

- Transaction Date
- Transaction Time
- ATM Location
- Transaction Amount
- Transaction used card number

METHOD NAME	DESCRIPTION
Deposit Money	Deposit an amount of money using a give card that is related to a given Customer ID that is associated with certain account.
Withdraw Monday	Withdraws a given amount of money if the account's balance permits so using a given card that is related to a given customer ID that is associated with certain account.

Pre-requisites and assumptions:

- The transactions for the customers should be stored in memory. You are not obliged to use linked list though using is a bonus. Alternatively, you can use arrays or vectors.
- It is expected to have the following on the class:

- One or more constructors
- Destructor

Delivery Guidelines:

- There are certain documentations that should be fulfilled upon the delivery of the assignment:
  - UML Class Diagram of all the classes with enough textual descriptions of both the class diagram, the classes, and the relationship between classes.
    UML Diagrams Documentation: https://tallyfy.com/uml-diagram/
  - An application (main function) to test all the public methods in your classes. The main function should be user selected. For example, the user should select if s/he wants to add a money, withdraw, show his/her balance etc.
  - Both the application and the classes methods should account for users' wrong inputs (Eg. Deducting -3000 L.E)
  - The source should be submitted well documented and adopting a good style of coding documentation found at:
    - https://users.ece.cmu.edu/~eno/coding/CppCodingStandard.html
    - https://isocpp.org/wiki/faq/coding-standards
    - http://web.mit.edu/6.s096/www/standards.html

- Use the following code for inheritance relationship at DropBox:
  \Dropbox\CSCE110-Students-Spring '21\Assignment 4