C20394301 Artyom McNiff and Justine Langridge’s Object-Oriented Programming CA 2 Project – Bank Management System

# User Manual

The bank management system provides a variety of features that this user manual aims to explain. When the user first starts the program, they are greeted with a menu that allows them to login, register, or exit the program. The user is able to register a customer account with the bank to log into the system. The user registers using their name, age, and a password. Once they have registered, they can log into the management system. They log in by inputting their surname and password. By logging in, they are able to view the existing bank account they may have created previously, or they can create a new bank account.

The user has two options when creating a new bank account. They can either create a savings account or a checking account. They both possess similar functionality, but a savings account can only make withdrawals or transfers once a month, whereas a checking account can make as many withdrawals as they please. Checking accounts also possess the ability to have a negative balance to a specified credit limit.

Once the user has created a bank account, they can view a list of their accounts. To perform certain banking functions on a particular account, the user must type in the account’s number. Once they have done so, a number of options become available to them.

The user can perform the following actions on their bank account

1. Check its balance
2. Deposit funds
3. Transfer funds
4. Withdraw funds
5. View recent transactions
6. Use interest/Mortgage calculator
7. Manage cards (credit/debit)
8. Delete their bank account
9. Open another bank account from their list
10. Log out of their customer account

Once the user has finished doing business with their account, they can log out and exit the program, or choose another account of theirs to do business with. If the user has no further use for their account, they can delete it. Keep in mind that all funds must be taken out of their account before it can be deleted.

# Classes and Methods

## Account

Account is one of three main classes in the bank management system. It is the superclass of SavingsAccount and CheckingAccount. Its purpose is to represent a typical bank account. The parameters it contains are as follows:

* acc\_id: the account id of the bank account
* iban: the International Bank Account Number of the bank account
* funds: the amount of money available in the bank account
* cid: the customer id of the customer who owns the bank account
* transactions: the list of transactions the customer has made using the bank account

The account class also contains a number of methods which are used to implement certain banking functions.

### Account Methods

recent\_transactions: Stores the 5 most recent account transactions from the accountsTransactions.txt file.

checkbalance: Checks how much money is stored in the bank account

deposit: deposits a specified amount of money into the bank account

withdraw: Withdraws a user-specified amount of money from their account

interestcalc: Lets users calculate mortgage repayments similar to that of online mortgage calculators and also lets them calculate expected profit on their savings.

## SavingsAccount

SavingsAccount is one of two subclasses in the bank management system. It is a subclass of Account. Its purpose is to represent the savings account archetype. The parameters it contains are identical to Account, but with one addition, that being:

* timeframe: The time at which the last transaction was made

The SavingsAccount class inherits all of Account’s methods along with its own specific methods.

### SavingsAccount Methods

add\_acc: adds a new account to the accounts.txt file by appending the instance’s information to the file.

transfer: transfers funds from one bank account to another

withdraw: Overrides the method in Account. The SavingsAccount version checks to see whether or not a transfer/withdrawal has been made in the last 30 days

## CheckingAccount

CheckingAccount is the second of the two Account subclasses, the first being SavingsAccount. Its purpose as an object is to represent the savings account archetype. The parameters are identical to its parent class, with an addition of a credit limit:

* credit\_limit: The amount of credit the user is able to borrow from the bank

CheckingAccount inherits its parent’s methods along with one additional method.

### CheckingAccount Methods

add\_acc: similar to SavingsAccount’s version, but with “checking” as its account type over “savings”

transfer: see SavingsAccount method

## Customer

Customer is the second main class that the program uses. As an object, it is used to represent a customer who uses the banking system. The parameters are as follows:

* cust\_id: the customer id, a unique identifier used as a way to differentiate between the different customers using the banking system
* fname: the first name of the customer
* surname: the surname of the customer
* age: the age of the customer
* no\_of\_accounts: the number of account the customer has opened.

The Customer class has access to the following methods.

### Customer Methods

add\_cust: adds a new customer to the customers.txt file by using the information stored in a particular instance and appending it to the file.

update\_cust: updates the number of accounts a customer has whenever they open a new account

# Card

Card is the third and final main class of the program. It provides a template for cards to be used at a (fictional) ATM. The parameters are:

* card\_num: the 16-digit number of the card which acts as the cards id
* expr\_date: the date at which the card expires
* cvv: stands for card verification value, stops scams
* pin: Personal Identification Number for the card
* a\_id: account id that the card is linked to.

The Card class contains no special methods, it is however the parent class of two subclasses, CreditCard and DebitCard, which have their own methods

# CreditCard

CreditCard is one of two subclasses of the Card class. An instance of CreditCard can only be created if the account\_id linked to the card is a CheckingAccount class type. It has an extra parameter, “card\_type”, which is set to “credit” by default

CreditCard contains the following method

## CreditCard Methods

add\_card: writes a CreditCard instance to the cards.txt file

# DebitCard

DebitCard is the second subclass of the Card class. An instance of DebitCard can be created regardless of the account type. It has an extra parameter, “card\_type”, which is set to “debit” by default.

## DebitCard Methods

see “CreditCard Methods”

# Individual Contributions

Justine Langridge handled the implementation of the bank’s functionality. She coded the balance, withdraw, deposit, transfer, and transactions methods, as well as providing the base for the login system and the file layout. Artyom McNiff handled the construction of the classes as well as coding the functionality to create new bank accounts and access existing bank accounts. He also handled everything to do with the card classes.

# Difficulties and Challenges

The main challenge came from the reading and writing of different files into the program. Reading and appending information to the files was straightforward, but what was tedious was updating pre-existing information. For example, updating the number of accounts the customer has every time they opened a new one required a far more complex solution to write the change to the file than it would be if there was no requirement to write the data to a file.

Error checking was also a slight challenge due to the amount of user input required to use the program. It was time consuming and slightly challenging as the information on all 4 files had to be matched and cross checked.

Another difficulty would be the amount of work to be done on the assignment, but this was mitigated greatly by working with another person. Between us we were able to divide up the work and help each other out.

# Conclusion

Thank you for reading the report. We hope you have a Merry Christmas.