James Clarke Object Oriented Programming Continuous Assessment Project C20375736

NOTE: Project requires the module 'simplejson' be installed to run.
This can be installed using the command 'pip install simplejson' in the python terminal.
If there are any problems pertaining to this please contact me.
Email: C20375736@mytudublin.ie

Global Dictionaries

These are dictionaries that are loaded from files when the program starts, and are saved to files when the program is exited, that contain each instance of their respective class, with a unique key for each instance.

CustomersDict

Stores every 'Customer' object. The 'username' attribute of the customer class is used for each key in the dictionary. Each item in this dictionary is saved to the file 'customers.txt' when the program is ended, and loaded from the same file when the program is booted.

AccountDict

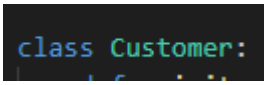
Stores every 'Account' object (including subclasses). The 'uniqueID' attribute of the customer class is used for each key in the dictionary. Each item in this dictionary is saved to the file 'accounts.txt' when the program is ended, and loaded from the same file when the program is booted.

TransactionDict

Stores every 'Transaction' object. The 'uniqueID' attribute of the Transaction class is used for each key in the dictionary. Each item in this dictionary is saved to the file 'accountTransactions.txt' when the program is ended, and loaded from the same file when the program is booted.

Project Classes

'Customer' Class



This is a class that denotes one user of the system. Each acts as a "profile" for one user, a user must create a customer class before they can use any other features of the application. The user can then create multiple bank accounts that will be tied to their 'Customer' class

Attributes	Description
username	A unique username that is set by the user on creation of their "user". This is used as unique ID for each customer
pin	A unique 4 digit string that is used to verify

	the user
name	The user's name
age	The user's age

Methods	**Description**
getAccountDict(self)	Will return a "sub dictionary" of the dictionary 'AccountDict', containing just the accounts the belong to the customer
menu(self)	Will display a menu that allows the user to run one of the following methods: ● selectAccount() ● createNewAccount() ● logout
selectAccount()	Displays a list of the user's accounts where the user can then select one to enter the account menu of. If the user has no accounts an error message will be displayed.
createNewAccount()	The user can create a new account of either Savings Account or Checking Account type, and give it a name
logout()	Returns the user to the login screen

'Account' Class

This class represents one Bank Account that can be created by the user. Multiple actions can be carried out with an account, such as deposit money, withdraw money, transfer money, etc.

Attributes	**Description**
uniqueID	This is a uniqueID that gets generated upon the accounts creation that is used as a unique key within the AccountDict dictionary. When an account object is created, the ID of each existing account within the 'AccountDict' is checked to ensure that the ID is unique.
ownerUsername	The 'username' attribute of the Customer object that created this account object

balance	The money balance stored in the account
name	A name assigned to the account by the user to make identifying account easier. The name is ensured to be unique within the scope of accounts created by the account owner, but not necessarily unique to all accounts

Methods	**Description**
getOwnerObject(self, customerDict)	Uses the ‘ownerUsername’ attribute of the account to return the Customer object that created the account. (Note: the customer object could not be directly stored within the object as this would then be too messy to store in a txt file)
viableCheck(self)	Checks if the age of the owner customer matches the allowed age to create the account and returns True if viable and False if not.
delete(self)	Will remove the dictionary entry containing this object, essentially deleting the object as it will not be saved when the program closes
recordTransaction(self, description)	Will create a new ‘Transaction’ object with the description that is passed in, and store it in the ‘TransactionDict’ dictionary
checkIfEnoughFunds(self, amount)	Checks if the passed in amount subtracted from the account.balance value will be less than the minimum balance allowed by the account. Returns True if allowed and False if not.
menu(self, previousMenu)	Display a persistent menu that allows the

	user to choose which to run of the following methods: ● viewBalance() ● withdraw() ● deposit() ● transfer() ● showTransactions() ● deleteConfirmation() ● goToPreviousMenu()
viewBalance(self)	Prints the current balance of the account
withdraw(self)	If they have sufficient funds, user withdraws a selected amount from account. A transaction log is created.
deposit(self)	The user enters an amount to be deposited, which is then added to their account. A transaction log is created.
transfer(self)	The user enters the unique ID number of the account they wish to transfer funds to. If the account exists, they will then be prompted to enter the amount they wish to transfer. If they have sufficient funds for their desired transaction, the amount will be removed from the host host account balance and added to the recipient account balance. Two transaction logs will be created, one for the account sending the money and one for the account receiving the money.
showTransactions(self)	Prints each transaction related to to the account object
deleteConfirmation(self)	Will ask the user for confirmation before deleting the account and then exit to the previous menu the user was on..
goToPreviousMenu(self)	Will access the menu of the ManageAccountMenu object passed into it

‘SavingAccount’ Subclass
This is a subclass of ‘Account’.
Unique Traits:

- Must be 14 years old or older to create
- Can only make on transfer/withdrawal per year

Unique Class Attributes	**Description**
minAge	Will always equal 14. Used to check what the minimum age is for account creation.

Unique Attributes	**Description**
type	The type of account it is. Will always be equal to ‘SavingsAccount’. This is used when loading the data from the file to distinguish account types
lastWithdrawMonth	The month the last withdrawal/transfer was made. This is used for the trait that savings accounts can only withdraw/transfer once every month

Unique Methods	**Description**
menu()	Allows user to access mostly the same functions as the account menu() method, except Withdraw() and Transfer() are grouped into one function ‘savingsWithdrawTransfer’, that can only be used once a month.
savingsWithdrawTransfer(se)	If the user can withdraw/transfer, a menu is

	displayed where the user can choose to run either the withdraw() or transfer() methods

‘CheckingAccount’ Subclass

Unique Class Attributes	**Description**
MinAge	Will always equal 18. Used for checking what age the user can create their account
minBalance	Will always be equal to -100. Will mean that the user can go into minus credit up to -100 euro.

Unique Instance Attributes	**Description**
type	Will always equal “CheckingAccount”.

‘Transaction’ Class

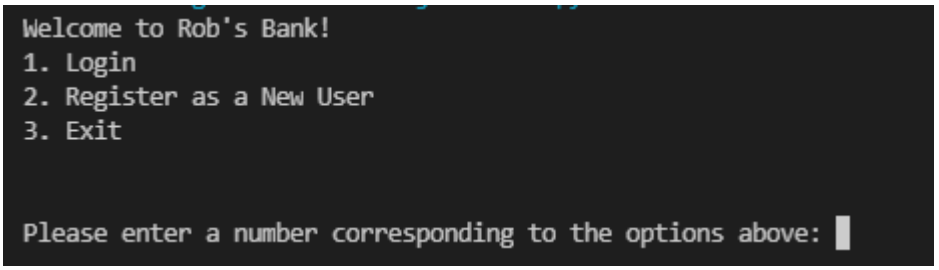
Attributes	**Description**
uniqueID	This is a uniqueID that gets generated upon the accounts creation that is used as a unique key within the ‘TransactionDict’ dictionary. When a Transaction object is created, the ID of each existing account within the ‘‘TransactionDict’ is checked to ensure that the ID is unique.
description	A textual log of the transaction
accountID	The ID of the account object that is the subject of the transaction

date	The current date and time

User Manual

Login Menu

When you first run the program, you are met with this menu:



As with all menus in this program, you enter the number corresponding to the option you wish to select. For example, if you wanted to 'Login', you would enter the number 1 and press enter.

Option 1: Login

Note: if you have not 'registered as a new user' you will not be able to proceed as you do not have login credentials.
You will be prompted to enter your username.
If you enter your username and it is correct, you will be prompted to enter your PIN number.
If you enter your correct PIN, you will be redirected to the **'Manage Accounts Menu.**

Option 2: Register as a New User

This will prompt you to enter the following details:

- a unique username (if the name is not unique you will be asked to enter it again)
- a 4 digit PIN number (you will be prompted to enter this number by number)
- Your first name
- Your second name
- Your age

After you enter all of these details correctly your account will be created and you will be redirected back to the **Login Menu,** where you can then use your username and pin to log in if you wish
Sample account creation:

```
Please enter a unique username
Please enter a unique name: James
Please enter a 4 digit PIN:
Enter digit 1
1
Enter digit 2
1
Enter digit 3
1
Enter digit 4
1
Please enter your first name: James
Please enter your last name: Clarke
Please enter your age: 19
Account Created Successfully!
```

Option 3: Exit
This will exit the program

<u>Manage Accounts Menu</u>
The Manage Accounts Menu looks like this:

```
Logged in as James Clarke
1. Select Account
2. Create New Account
3. Logout


Please enter a number corresponding to the options above:
```

Again, you enter a number and press enter to select a menu option

Option 1: Select Account
Note: If you have not created any accounts yet, a message will be displayed explaining that you have no accounts

If you do have accounts created, they will be listed like so:

```
Please input the number corresponding to the account you wish to use:
1 : MyCheckingAccount (CheckingAccount)
2 : MySavingsAccount (SavingsAccount)
3 : MyCheckingAccount2 (CheckingAccount)
4 : Go Back


Input number:
```

Entering the number next to one of your accounts will take you to the **<u>Account Menu</u>** for that account.

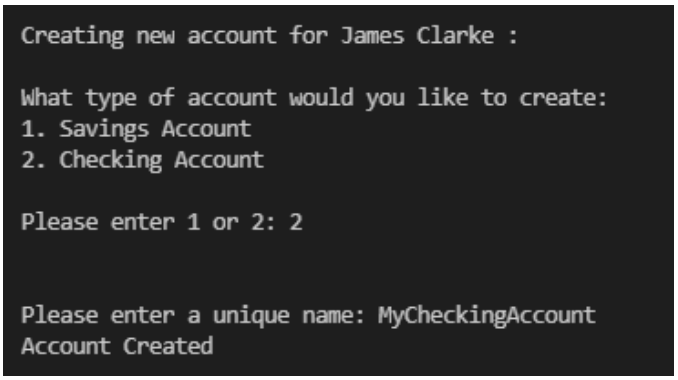
Selecting the number next to ‘Go Back’ will return you to the **Manage Accounts Menu**

Option 2: Create Account
You will be prompted to select an account type by entering a number.
You will then be prompted to enter a unique name for your account.

When you fill out those two criteria your account will be created and you be returned to the **Manage Accounts Menu**

NOTE: If you are under 18 you cannot create a Checking account and if you are under 14 you cannot create a Savings Account

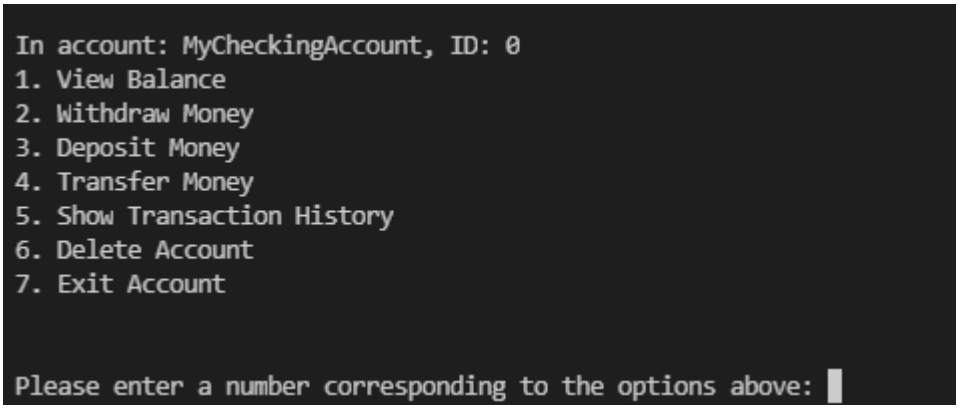
Sample Account Creation:



Option 3: Logout
This will return you to the **Login Menu**

Account Menu
This menu shows you all the functions pertaining to your account
Note: The menus for Checking Account and Savings Account are different

Checking Account Menu



Option: View Balance
This will display the current balance of the account:

```
Please enter a number corresponding to the options above: 1

Your current balance is €0
```

Option: Withdraw Money
This will prompt you to enter an amount to be withdrawn. The entered amount will be subtracted from your bank account
Note:

- Checking account balance minimum is -100 euro
- Savings account balance minimum is 0 euro

You cannot withdraw money past the values specified above.

```
Please enter a number corresponding to the options above: 2

Please enter an amount to withdraw: 15

€15 withdrawed from account ID 0 by Account Owner James. New Account Balance: €35
```

Option: Deposit Money
This will prompt you to enter an amount to deposit. The entered amount will be added to your bank account:

```
Please enter an amount to deposit: 20

€20 deposited into account ID 0 by Account Owner James. New Account Balance: €55
```

Option: Transfer Money
This will ask you to enter the ID of the account you wish to transfer money into (The account ID is displayed at the top of **Account Menu**).
If an account of the entered ID exists, you will then be prompted to enter the amount you wish to transfer.
The entered amount will be subtracted from your account and added to the recipient account.

Note: Minimum value restrictions mentioned in **Option: Withdraw Money** Apply here too

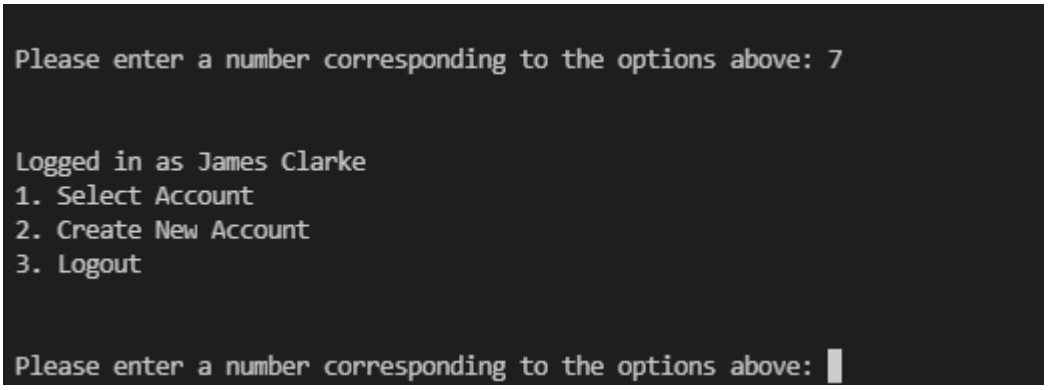
Option: Show Transaction History
This will print to the screen any transactions that involved the current account:

```
Showing Transaction History for account MyCheckingAccount:

[Transaction ID: 0] [Account ID: 0] 18/12/2021 23:21 : €50 deposited into account ID 0 by Account Owner James. New Account Balance: €50
[Transaction ID: 1] [Account ID: 0] 18/12/2021 23:23 : €15 withdrawed from account ID 0 by Account Owner James. New Account Balance: €35
[Transaction ID: 2] [Account ID: 0] 18/12/2021 23:24 : €20 deposited into account ID 0 by Account Owner James. New Account Balance: €55
[Transaction ID: 3] [Account ID: 0] 18/12/2021 23:25 : €30 transferred into Account ID 1. New Account Balance: €25
```

Option: Delete Account
This will first ask for your confirmation and then delete your account. You will be redirected to the **Account Management Menu:**

```
Please enter a number corresponding to the options above: 6


Are you sure you wish to delete the account 0?

1. Delete
2. Go back
Please enter a number corresponding to the options above: 1


MyCheckingAccount deleted!

Logged in as James Clarke
1. Select Account
2. Create New Account
3. Logout


Please enter a number corresponding to the options above:
```

Option 7: Exit Account
This will return your to the **Account Management Menu:**



Savings Account Withdraw and Transfer

When in the account menu of a savings account, in order to access the Withdraw and Transfer options mentioned above, you have to select the option **Withdraw or Transfer Money.**

NOTE: In a savings account you can only carry out 1 transfer or withdraw per month. The option will remain locked until the following month

```
2. Deposit Money
3. Show Transaction History
4. Withdraw or Transfer Money
5. Delete Account
6. Exit Account


Please enter a number corresponding to the options above: 4


Please select an option (NOTE: You can only perform one of these actions per month):

1. Withdraw
2. Transfer
Please enter a number corresponding to the options above: 1


Please enter an amount to withdraw: 5


€5 withdrawed from account ID 1 by Account Owner James. New Account Balance: €25


In SavingsAccount account: MySavingsAccount
1. View Balance
2. Deposit Money
3. Show Transaction History
4. Withdraw or Transfer Money (Already had one withdraw/transfer this month, cannot withdraw more)
5. Delete Account
6. Exit Account


Please enter a number corresponding to the options above:
```

Exiting

Exit the program through the ‘Exit’ option in the **Login Menu** to save all data, accounts and changes made.

Team Members Contributions

I, James Clarke, created everything in this project myself.

Difficulties and Challenges

The most difficult part I found was structuring the entire project. I had to go through much iteration and experimentation to find a layout that was easy to work with and also easily expandable. The final design that I settled on with the menus within the classes proved very dynamic and workable, as having a central menu to access all of the methods within a function proved very fitting for this particular project.

Storing and Loading the persistent data proved both interesting and challenging.
This involved both what data structures I would use to store the data and modify the data while the project was running, but also how I would format the data in the files, so that they would store in a txt format.
I settled on storing my data in json format in text files, where each line would contain a json string that would contain the information on one object. This is then loaded in and stored into a dictionary of objects on runtime.

Creativity

This project demanded many creative decisions from me.

- The decision to have a login menu where the user can create a user account was something I had to come up with
- The data being stored in json format was also something that I had to exercise some creative muscles figure out
- The decision to have a menu within an object was a creative one on my part, as it allows for the end user to access methods from directly within the object.
- The user is able to give their accounts names, adding a personal touch as well as making the accounts more idenitfiable

Further Development

To further develop this project, I would add another account subclass called ‘SterlingAccount’ that would be used for storing money in pound sterling. All the main account methods would remain the same, except the transfer money method would need to convert euro to sterling and vise versa.