24/05/2022

# Assessment Task 3:

Python Project 2

Maxordia Financial.

Oisin Aeonn
S3952320

**Table of Contents:**

**Introduction:**

The scope of this project is to explore and continue learning how to program Python 3. In this project we will implement what we've learnt so far about the more complex functions of Python 3 and use them to create a basic banking system. To complete this project, we will have to use and differentiate between different types of data present in our code such as Integers, Floats and Strings. We will also need to understand and implement how to correctly store data / code in lists, classes, and functions. Math functions will also deeply aid us as well as sequence, selection and iteration. This project will allow us to again put into practice how to test and debug Python.

We have been employed by a bank by the name of 'Maxordia Financial' as a software developer. We have been given the task of creating a new software application for processing and accessing user's bank accounts. This can be seen in the coding Appendices B, D and G.

The programming language we are going to use to achieve these requirements is Python 3. We are using Python because it is a fast object-orientated programming language that is interpreted, interactive, with very high-level dynamic data types making it easy to learn. It will make satisfying our User Requirements much easier, as all types of data can be used seamlessly without consequence. It is also the programming language we are learning in this course: Diploma of Information Technology.

3 characteristics of Python you should consider are:

Sequence: Python is a sequence-based language meaning it is interpreted through every line of code in order: from top to bottom. This means in practice when writing our code, we need to state our variables before using them, and write out our code in the order in which we want it executed.

Selection: Selection is a conditional statement that is used to decide when an action should be performed depending on if the condition is met. Selection is evaluated as a Boolean value: being either True or False. In Python we use if, elif and else statements to write our code. We can use these to ensure that all our options can be fulfilled given to us in the User Requirements.

Iteration: Iteration is used to save-time it allows for the repetition of code. In Python we use for and while statements to conduct Iteration. In Python we use def, for and while statements to write our code. Iteration can be used to restart our program entirely which will satisfy one of our User Requirements.

In this particular project I decided to import a re function to search our text documents for correctly corresponding data as well as storing data / code within classes as instructed by the User Requirements and UML Diagram.

The algorithm we will use is denoted in the UML Diagram.

## User Requirements:

## NOTE IGNORE THESE IT JUST SAYS IN THE REPORT TO SHOW THEM.

menu() function in Bank class
1. Display main menu for user to select an option from 1 to 5.
2.1 If user selects option 1 in the main menu:
2.1.1 Prompt user to enter checking account number.
2.1.2 Store checking account number
2.1.3 If user enters non-numerical (letters or special characters) values for checking account number:
2.1.3.1 Display error message
2.1.3.2 Go back to step 1
2.1.4 Search for checking account number in CAccounts[] list (run getCAccount() function)
2.1.5 If index returned of checking account number is not -1:
2.1.5.1 Display submenu for checking account selected (run checkingMenu() function)
2.1.6 If index returned of checking account number is -1:
2.1.6.1 Display error message
2.1.6.2 Go back to step 1
2.2 If user selects option 2 in the main menu:
2.2.1 Prompt user to enter saving account number.
2.2.2 Store saving account number
2.2.3 If user enters non-numerical (letters or special characters) values for saving account number:
2.2.3.1 Display error message
2.2.3.2 Go back to step 1
2.2.4 Search for saving account number in SAccounts[] list (run getSAccount() function)
2.2.5 If index returned of saving account number is not -1:
2.2.5.1 Display submenu for saving account selected (run savingMenu() function)
2.2.6 If index returned of saving account number is -1:
2.2.6.1 Display error message
2.2.6.2 Go back to step 1
2.3 If user selects option 3 in the main menu:

2.3.1 Display objects of CAccount class from CAccounts[] list
2.3.2 Go back to step 1
2.4 If user selects option 4 in the main menu:
2.4.1 Display objects of SAccount class from SAccounts[] list
2.4.2 Go back to step 1
2.5 If user selects option 5 in the main menu:
2.5.1 Display exit message.
2.5.2 Go to step 2
2.6 If user enters numbers less than 1, numbers greater than 5 and any non-numerical (letters or special characters) values for the main menu option:
2.6.1 Display an error message
2.6.2 Go back to step 1
2. End

checkingMenu() function in Bank class
1. Access selected object of CAccount class from CAccounts[] list
2. Display submenu for user to select an option from 1 to 5
2.1 If user selects option 1 in the submenu:
2.1.1 Prompt user to enter deposit amount
2.1.2 Store deposit amount
2.1.3 If user entered positive whole number or positive decimal number for deposit amount:
2.1.3.1 Perform deposit transaction on selected object of CAccount class (run deposit() function from CAccount class)
2.1.3.2 Reassign selected object of CAccount class to the CAccounts[] list
2.1.3.3 Open BankingReceipt.txt file
2.1.3.4 Insert account number of selected object of CAccount class into BankingReceipt.txt file
2.1.3.5 Insert deposit amount entered by user into BankingReceipt.txt file

2.1.3.6 Insert balance of selected object of CAccount class into BankingReceipt.txt file
2.1.3.7 Close the BankingReceipt.txt file
2.1.3.8 Go back to step 2
2.1.4 If user entered negative whole number, negative decimal number and any non-numerical (letters or special characters) values for the deposit amount:
2.1.4.1 Display an error message
2.1.4.2 Go back to step 2
2.2 If user selects option 2 in the submenu:
2.2.1 Prompt user to enter withdrawal amount
2.2.2 Store withdrawal amount
2.2.3 If user entered positive whole number or positive decimal number for withdrawal amount:

2.2.3.1 Perform withdraw transaction on selected object of CAccount class (run withdraw() function from CAccount class)
2.2.3.2 If Boolean value returned is True:
2.2.3.2.1 Reassign selected object of CAccount class to the CAccounts[] list
2.2.3.2.2 Open BankingReceipt.txt file
2.2.3.2.3 Insert account number of selected object of CAccount class into BankingReceipt.txt file
2.2.3.2.4 Insert withdrawal amount entered by user into BankingReceipt.txt file
2.2.3.2.5 Insert balance of selected object of CAccount class into BankingReceipt.txt file
2.2.3.2.6 Close the BankingReceipt.txt file
2.2.3.2.7 Go back to step 2
2.2.4 If user entered negative whole number, negative decimal number and any non-numerical (letters or special characters) values for the withdrawal amount:
2.2.4.1 Display an error message
2.2.4.2 Go back to step 2
2.3 If user selects option 3 in the submenu:

2.3.1 Display balance of selected object of CAccount class
2.3.2 Open BankingReceipt.txt file
2.3.3 Insert account number of selected object of CAccount class into BankingReceipt.txt file
2.3.4 Insert balance of selected object of CAccount class into BankingReceipt.txt file
2.3.5 Close the BankingReceipt.txt file
2.3.6 Go back to step 2
2.4 If user selects option 4 in the submenu:
2.4.1 Display selected object of CAccount class from CAccounts[] list
2.5 If user selects option 5 in the submenu:
2.5.1 Go to step 3
2.6 If user enters numbers less than 1, numbers greater than 5 and any non-numerical (letters or special characters) values for the submenu option:
2.6.1 Display an error message
2.6.2 Go back to step 2
3. Go back to main menu (run menu() function)

savingMenu() function in Bank class
1. Access selected object of SAccount class from SAccounts[] list
2. Display submenu for user to select an option from 1 to 5
2.1 If user selects option 1 in the submenu:
2.1.1 Prompt user to enter deposit amount
2.1.2 Store deposit amount
2.1.3 If user entered positive whole number or positive decimal number for deposit amount:
2.1.3.1 Perform deposit transaction on selected object of SAccount class (run deposit() function from SAccount class)
2.1.3.2 Reassign selected object of SAccount class to the SAccounts[] list

2.1.3.3 Open BankingReceipt.txt file
2.1.3.4 Insert account number of selected object of SAccount class into BankingReceipt.txt file
2.1.3.5 Insert deposit amount entered by user into BankingReceipt.txt file
2.1.3.6 Insert balance of selected object of SAccount class into BankingReceipt.txt file
2.1.3.7 Close the BankingReceipt.txt file
2.1.3.8 Go back to step 2
2.1.4 If user entered negative whole number, negative decimal number and any non-numerical (letters or special characters) values for the deposit amount:
2.1.4.1 Display an error message
2.1.4.2 Go back to step 2
2.2 If user selects option 2 in the submenu:
2.2.1 Prompt user to enter withdrawal amount
2.2.2 Store withdrawal amount
2.2.3 If user entered positive whole number or positive decimal number for withdrawal amount:
2.2.3.1 Perform withdraw transaction on selected object of SAccount class (run withdraw() function from SAccount class)
2.2.3.2 If Boolean value returned is True:
2.2.3.2.1 Reassign selected object of SAccount class to the SAccounts[] list
2.2.3.2.2 Open BankingReceipt.txt file
2.2.3.2.3 Insert account number of selected object of SAccount class into BankingReceipt.txt file
2.2.3.2.4 Insert withdrawal amount entered by user into BankingReceipt.txt file
2.2.3.2.5 Insert balance of selected object of SAccount class into BankingReceipt.txt file

2.2.3.2.6 Close the BankingReceipt.txt file
2.2.3.2.7 Go back to step 2

2.2.4 If user entered negative whole number, negative decimal number and any non-numerical (letters or special characters) values for the withdrawal amount:
2.2.4.1 Display an error message
2.2.4.2 Go back to step 2
2.3 If user selects option 3 in the submenu:
2.3.1 Display balance of selected object of SAccount class
2.3.2 Open BankingReceipt.txt file
2.3.3 Insert account number of selected object of SAccount class into BankingReceipt.txt file
2.3.4 Insert balance of selected object of SAccount class into BankingReceipt.txt file
2.3.5 Close the BankingReceipt.txt file
2.3.6 Go back to step 2
2.4 If user selects option 4 in the submenu:
2.4.1 Display selected object of SAccount class from SAccounts[] list
2.5 If user selects option 5 in the submenu:
2.5.1 Go to step 3
2.6 If user enters numbers less than 1, numbers greater than 5 and any non-numerical (letters or special characters) values for the submenu option:
2.6.1 Display an error message
2.6.2 Go back to step 2
2.7 Go back to main menu (run menu() function)

getCAccount() function in Bank class
1. Go through each object of CAccount class from the CAccounts[] list
1.1 If current object of CAccount class from the CAccounts[] list matches the checking account number entered by user:
1.1.1 Return index of current object of CAccount class from the CAccounts[] list
1.1.2 Go to step 3
2. Return index of -1
3. End

getSAccount() function in Bank class
1. Go through each object of SAccount class from the SAccounts[] list
1.1 If current object of SAccount class from the SAccounts[] list matches the saving account number entered by user:
1.1.1 Return index of current object of SAccount class from the SAccounts[] list
1.1.2 Go to step 3
2. Return index of -1
3. End

main() function in Bank class
1. Read the data from the CAccounts.txt file to create objects of CAccount class inside the CAccounts[] list (run loadCAccounts() function)
2. Read the data from the SAccounts.txt file to create objects of SAccount class inside the SAccounts[] list (run loadSAccounts () function)
3. Display the main menu (run menu() function)
4. End

loadCAccounts() function in Bank class
1. Open CAccounts.txt file
2. Read each line of data inside CAccounts.txt file
2.1 Remove the space at the end of the current line of data
2.2 Split the data between the semicolons ';' in the current line into separate fields
2.3 If the number of separate fields is 3 in the current line:
2.3.1 Create object of CAccount class using the 3 separate fields
2.3.2 Insert the object of CAccount class inside the CAccounts[] list
2.4 If the number of separate fields is 4 in the current line:
2.4.1 Create object of CAccount class using the 4 separate fields
2.4.2 Insert the object of CAccount class inside the CAccounts[] list
2.5 Close the CAccounts.txt file
3. End

loadSAccounts() function in Bank class
1. Open SAccounts.txt file
2. Read each line of data inside SAccounts.txt file
2.1 Remove the space at the end of the current line of data
2.2 Split the data between the semicolons ';' in the current line into separate fields
2.3 If the number of separate fields is 3 in the current line:

2.3.1 Create object of SAccount class using the 3 separate fields
2.3.2 Insert the object of SAccount class inside the SAccounts[] list
2.4 If the number of separate fields is 4 in the current line:
2.4.1 Create object of SAccount class using the 4 separate fields
2.4.2 Insert the object of SAccount class inside the SAccounts[] list
2.5 Close the SAccounts.txt file
3. End

deposit() function in CAccount class
1. Add deposit amount to balance
2. Round off balance to 2 decimal places
3. Display balance
4. Display confirmation message of successful deposit transaction
5. End

withdraw() function in CAccount class
1. If withdrawal amount is less than minimum amount:
1.1 Display minimum amount
1.2 Display error message
1.3 Display confirmation message of unsuccessful withdrawal transaction
1.4 Return Boolean value of False
1.5 Go to step 4
2. If withdrawal amount is greater than balance:
2.1 Display balance
2.2 Display error message
2.3 Display confirmation message of unsuccessful withdrawal transaction
2.4 Return Boolean value of False
2.5 Go to step 4
3. If withdrawal amount is less than or equal to balance and withdrawal amount is greater than or equal to minimum amount:
3.1 Subtract withdrawal amount from the balance
3.2 Round off balance to 2 decimal places
3.3 Display balance
3.4 Display confirmation message of successful withdrawal transaction
3.5 Return Boolean value of True
3.6 Go to step 4
4. End

deposit() function in SAccount class
1. Add deposit amount to balance
2. Round off balance to 2 decimal places
3. Display balance
4. Display confirmation message of successful deposit transaction
5. End

withdraw() function in SAccount class
1. If withdrawal amount is greater than maximum amount:
1.1 Display maximum amount
1.2 Display error message
1.3 Display confirmation message of unsuccessful withdrawal transaction
1.4 Return Boolean value of False
1.5 Go to step 4
2. If withdrawal amount is greater than balance:
3.7 Display balance
3.8 Display error message
3.9 Display confirmation message of unsuccessful withdrawal transaction
3.10 Return Boolean value of False
3.11 Go to step 4
3. If withdrawal amount is less than or equal to balance and withdrawal amount is less than or equal to maximum amount:
3.1 Subtract withdrawal amount from the balance
3.2 Round off balance to 2 decimal places
3.3 Display balance
3.4 Display confirmation message of successful withdrawal transaction
3.5 Return Boolean value of True
3.6 Go to step 4
4 End

**Task 1:**

See Appendix D called CAccount.py

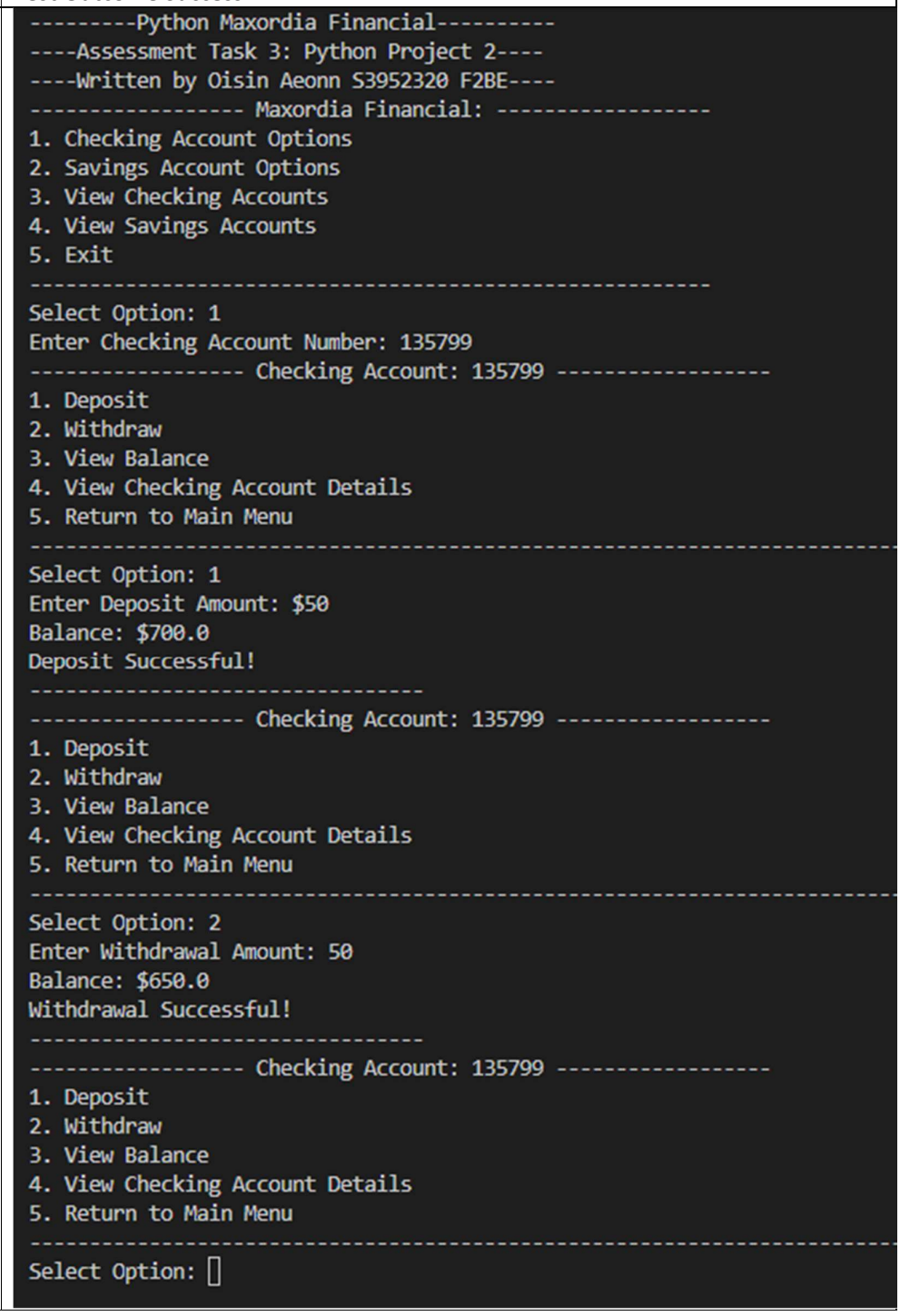
**Task 2:**

See Appendix G called SAccount.py

**Task 3:**

See Appendix B called BankApp.py

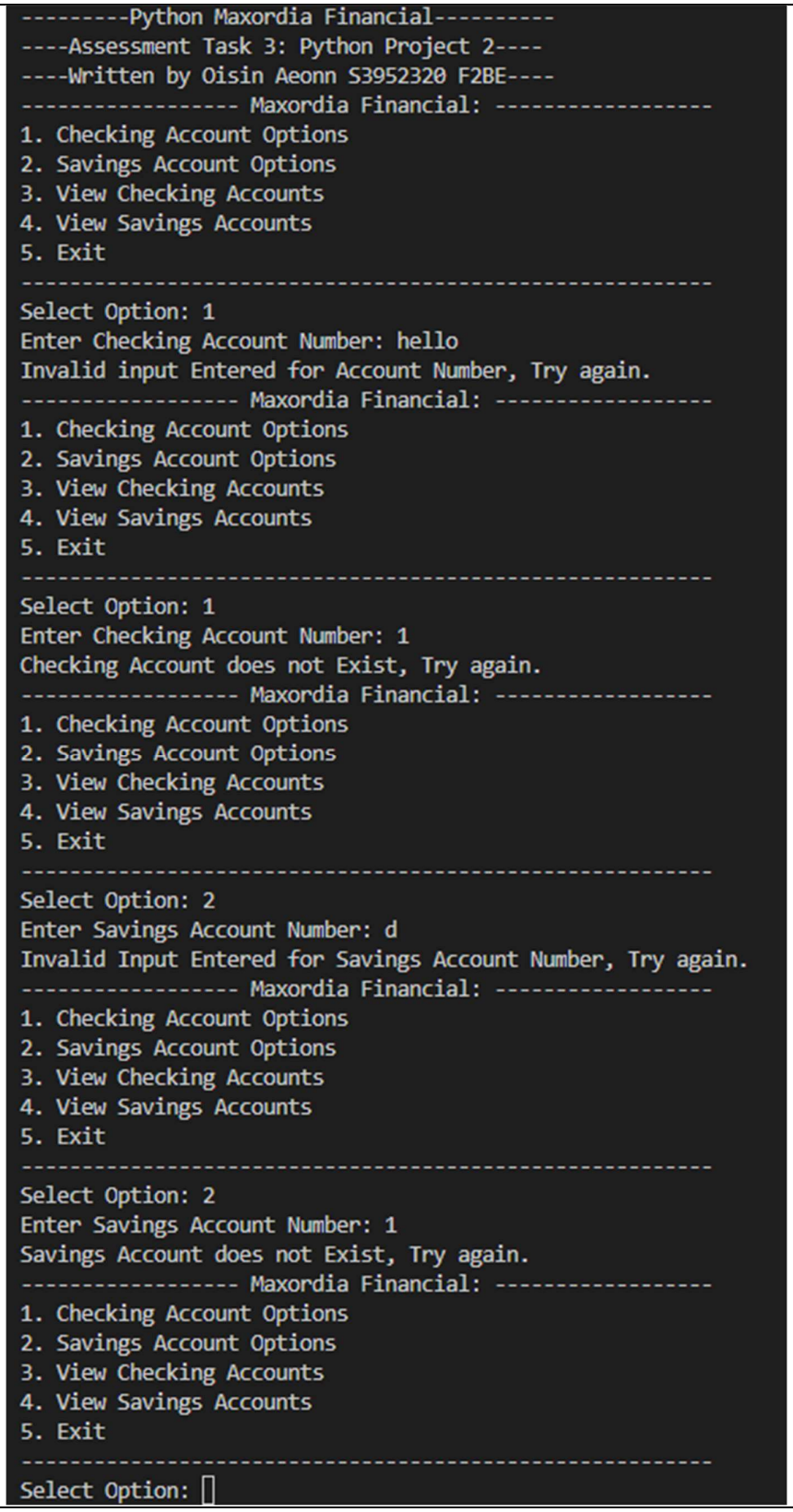
**Task 4a:**

| | Test Outcome Success |
|---|---|
| **Test 1 (allow authorised)** |  |

```
--------------------------------------------------------------------------------
Select Option: 5
------------------ Maxordia Financial: ------------------
1. Checking Account Options
2. Savings Account Options
3. View Checking Accounts
4. View Savings Accounts
5. Exit
------------------------------------------------------------
Select Option: 2
Enter Savings Account Number: 246800
------------------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 1
Enter Deposit Amount: $50
Balance: $650.0
Deposit Successful!
---------------------------------
------------------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 2
Enter Withdrawal Amount: 50
Balance: $600.0
Withdrawal Successful!
---------------------------------
------------------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option:
```

```
---------Python Maxordia Financial----------
----Assessment Task 3: Python Project 2----
----Written by Oisin Aeonn S3952320 F2BE----
------------------ Maxordia Financial: ------------------
1. Checking Account Options
2. Savings Account Options
3. View Checking Accounts
4. View Savings Accounts
5. Exit
-----------------------------------------------------------
Select Option: 1
Enter Checking Account Number: 135799
------------------ Checking Account: 135799 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Checking Account Details
5. Return to Main Menu
-------------------------------------------------------------------------
Select Option: 1
Enter Deposit Amount: $50.1
Balance: $700.1
Deposit Successful!
--------------------------------
------------------ Checking Account: 135799 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Checking Account Details
5. Return to Main Menu
-------------------------------------------------------------------------
Select Option: 2
Enter Withdrawal Amount: 50.1
Balance: $650.0
Withdrawal Successful!
--------------------------------
------------------ Checking Account: 135799 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Checking Account Details
5. Return to Main Menu
-------------------------------------------------------------------------
Select Option:
```

```
----------------- Maxordia Financial: ------------------
1. Checking Account Options
2. Savings Account Options
3. View Checking Accounts
4. View Savings Accounts
5. Exit
------------------------------------------------------
Select Option: 2
Enter Savings Account Number: 246800
----------------- Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
-------------------------------------------------------------------------
Select Option: 1
Enter Deposit Amount: $50.1
Balance: $650.1
Deposit Successful!
--------------------------------
----------------- Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
-------------------------------------------------------------------------
Select Option: 2
Enter Withdrawal Amount: 50.1
Balance: $600.0
Withdrawal Successful!
--------------------------------
----------------- Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
-------------------------------------------------------------------------
Select Option:
```

| | |
|---|---|
| **Test 2 (blocking unauthorised)** |  |

```
------------------------------------------------------------
Select Option: 1
Enter Checking Account Number: 135799
------------------ Checking Account: 135799 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Checking Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 1
Enter Deposit Amount: $d
Invalid Input Entered for Deposit Amount.
Deposit Failed, Try Again.
------------------ Checking Account: 135799 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Checking Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 1
Enter Deposit Amount: $-50
Invalid Input Entered for Deposit Amount.
Deposit Failed, Try Again.
------------------ Checking Account: 135799 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Checking Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 2
Enter Withdrawal Amount: 30
Minimum Amount: $50.0
Withdraw Amount is Less than the Minimum Amount
Withdraw Failed, Try Again.
--------------------------------
------------------ Checking Account: 135799 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Checking Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option:
```

```
------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: -50
Invalid Option Selected, Try again.
------------------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 2
Enter Withdrawal Amount: 1001
Balance: $600.0
Withdrawal Amount is Greater than the Balance
Withdrawal Failed, Try Again.
--------------------------------
------------------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 1
Enter Deposit Amount: $9999
Balance: $10599.0
Deposit Successful!
--------------------------------
------------------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 2
Enter Withdrawal Amount: 2000
Maximum Amount: $1000.0
Withdraw Amount is Greater than the Maximum Amount
Withdraw Failed, Try Again.
--------------------------------
------------------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option:
```

NOTE I HAD SOME ISSUES WHERE I HAD ACCIDENTALLY INVERTED THE MINIMUM AND MAXIMUM AMOUNTS AND HAD APPLIED IT TO DEPOSITS ASWELL. I FIXED THEM BEFORE UNDERTAKING THESE TESTS SO THERE WAS NOTHING MAJOR TO MODIFY AFTER CONDUCTING THE TESTS.

**Task 4b:**

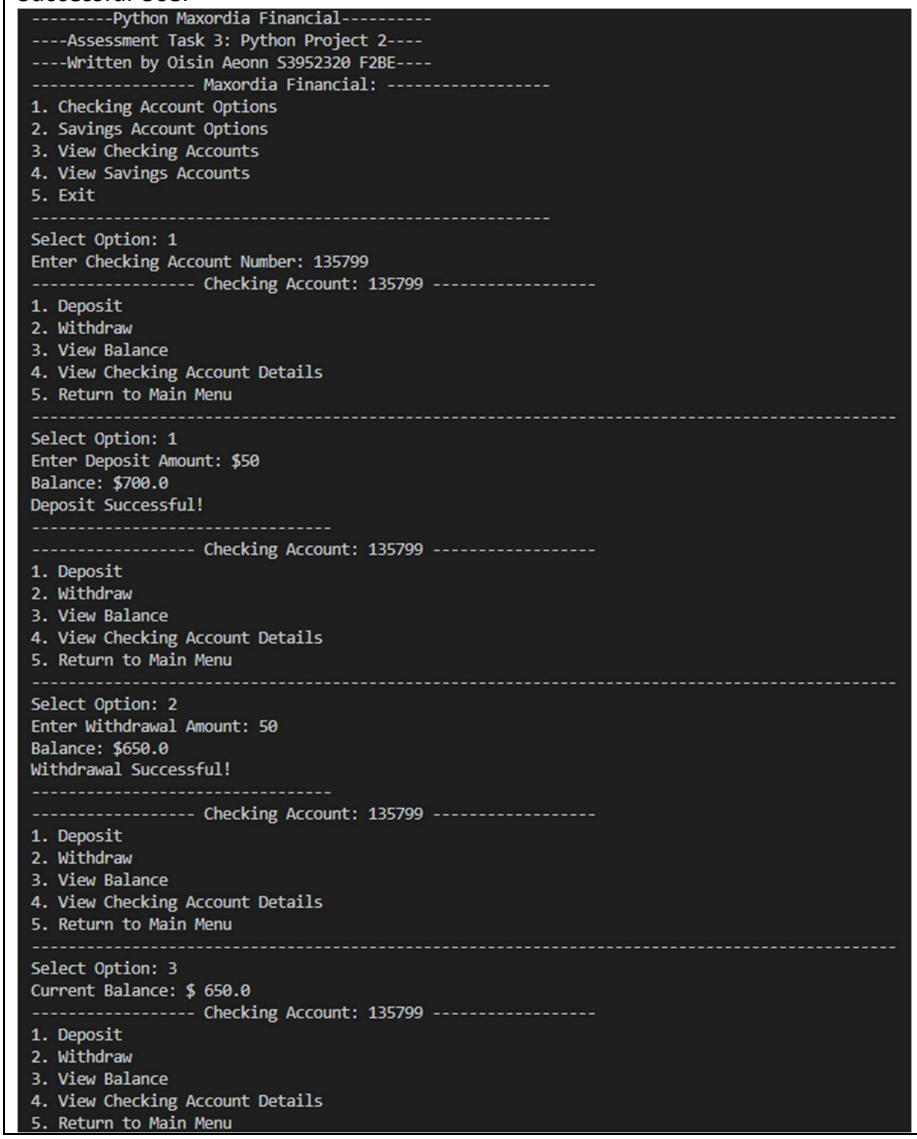
I swapped my final program with Wonbin Kim who found no issues with my code and vice versa.

**Task 4c: Successful Use of Program**

I ensured by running a second round of tests to ensure everything was working as intended and then finished cleaning up and commenting all of my code so that anyone else could look at it and understand what was going on.

After this I took more screenshots showing the successful running of my code.

Successful Use:



```
1. Deposit
2. Withdraw
3. View Balance
4. View Checking Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 4
----------------- Checking Account Details: -----------------
Account Number: 135799
Account Name: Vanessa
Balance: $650.0
Maximum Amount: $50.0
------------------ Checking Account: 135799 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Checking Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 5
------------------ Maxordia Financial: ------------------
1. Checking Account Options
2. Savings Account Options
3. View Checking Accounts
4. View Savings Accounts
5. Exit
------------------------------------------------------------
Select Option: 2
Enter Savings Account Number: 246800
------------------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 1
Enter Deposit Amount: $50
Balance: $650.0
Deposit Successful!
---------------------------------
------------------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 2
Enter Withdrawal Amount: 50
Balance: $600.0
Withdrawal Successful!
---------------------------------
```

```
------------------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 3
Current Balance: $ 600.0
------------------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 4
----------------- Savings Account: -----------------
Account Number: 246800
Account Name: Anthony
Balance: $600.0
Minimum Amount: $1000.0
------------------ Savings Account: 246800 ------------------
1. Deposit
2. Withdraw
3. View Balance
4. View Savings Account Details
5. Return to Main Menu
--------------------------------------------------------------------------------
Select Option: 5
------------------ Maxordia Financial: ------------------
1. Checking Account Options
2. Savings Account Options
3. View Checking Accounts
4. View Savings Accounts
5. Exit
------------------------------------------------------------
Select Option: 3
------------------ Checking Accounts: ------------------
Account Number: 135790
Account Name: Anna
Balance: $350.0
Maximum Amount: $50.0
**********
------------------------------------------------------------
Account Number: 135791
Account Name: Frank
Balance: $500.0
Maximum Amount: $100.0
**********
------------------------------------------------------------
```

```
------------------------------------------------------------
Account Number: 135792
Account Name: Mark
Balance: $700.0
Maximum Amount: $200.0
**********
------------------------------------------------------------
Account Number: 135793
Account Name: Matthew
Balance: $850.0
Maximum Amount: $400.0
**********
------------------------------------------------------------
Account Number: 135794
Account Name: Natasha
Balance: $1000.0
Maximum Amount: $100.0
**********
------------------------------------------------------------
Account Number: 135795
Account Name: Peter
Balance: $1200.0
Maximum Amount: $200.0
**********
------------------------------------------------------------
Account Number: 135796
Account Name: Ryan
Balance: $1400.0
Maximum Amount: $400.0
**********
------------------------------------------------------------
Account Number: 135797
Account Name: Stephanie
Balance: $950.0
Maximum Amount: $100.0
**********
------------------------------------------------------------
Account Number: 135798
Account Name: Tony
Balance: $1150.0
Maximum Amount: $200.0
**********
------------------------------------------------------------
Account Number: 135799
Account Name: Vanessa
Balance: $650.0
Maximum Amount: $50.0
**********
```

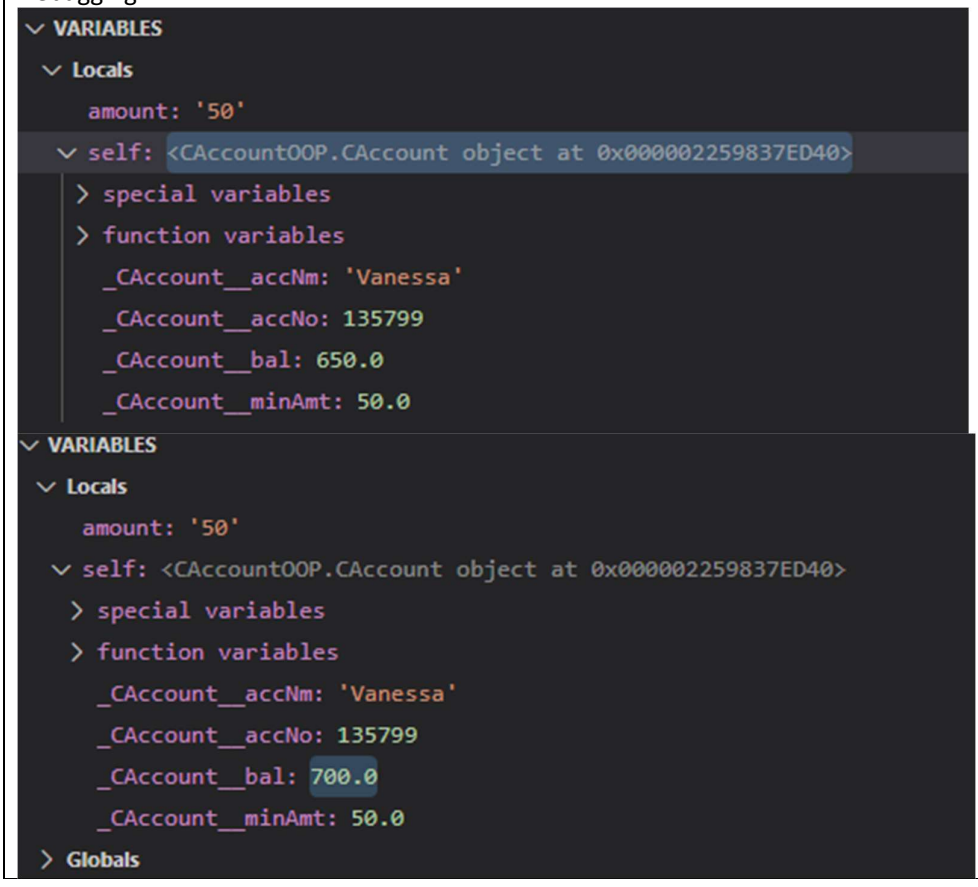
```
1. Checking Account Options
2. Savings Account Options
3. View Checking Accounts
4. View Savings Accounts
5. Exit
------------------------------------------------------------
Select Option: 4
----------------- Savings Accounts: -----------------
Account Number: 246800
Account Name: Anthony
Balance: $600.0
Minimum Amount: $1000.0
************************
Account Number: 246801
Account Name: Bruce
Balance: $800.0
Minimum Amount: $2000.0
************************
Account Number: 246802
Account Name: Diana
Balance: $950.0
Minimum Amount: $4000.0
************************
Account Number: 246803
Account Name: Kate
Balance: $450.0
Minimum Amount: $500.0
************************
Account Number: 246804
Account Name: Michael
Balance: $1100.0
Minimum Amount: $1000.0
************************
Account Number: 246805
Account Name: Nathan
Balance: $1300.0
Minimum Amount: $2000.0
************************
Account Number: 246806
Account Name: Paul
Balance: $1500.0
Minimum Amount: $4000.0
************************
Account Number: 246807
Account Name: Rebecca
Balance: $700.0
Minimum Amount: $500.0
************************
Account Number: 246808
Account Name: Robert
Balance: $1050.0
Minimum Amount: $1000.0
************************
```

```
Account Number: 246809
Account Name: Sarah
Balance: $1400.0
Minimum Amount: $2000.0
**********
---------- Maxordia Financial: ----------
1. Checking Account Options
2. Savings Account Options
3. View Checking Accounts
4. View Savings Accounts
5. Exit
--------------------------------------------------
Select Option: 5
Thank You for Selecting Maxordia Financial, Have a Good Day!
PS C:\Users\ODISH\OneDrive\Desktop\Python 2 of 3\S3952320 OISIN AEONN PYTHON PROJEC
```

**Task 4d: Debug**

Debugging was not listed in this project, however I still decided to undergo some debugging.

Debugging:



**Task 5:**

**User Manual:**

Step 1: Run the program in a Python Compiler e.g. Visual Studio Code or similar.

Step 2: You will be prompted with a Menu with 5 different choices.

Step 3: Select a value between 1 – 5 based on the Menu Choices given.

**Choice 1 / 2:**

These options will bring a further submenu with 5 more options select 1 for depositing which will prompt you for an amount which will be added to your account, 2 for withdrawing which will ask you for an amount which will be deducted from your account, 3 which will show you just your balance, 4 for your entire account details or 5 to return to the previous menu.

**Choice 3 / 4:**

These options will display all known accounts present in their corresponding text files (3 for CAccounts or 4 for SAccounts).

**Choice 5:**

This option will exit the program, leaving you with a farewell message.

**Errors:**

If an invalid input containing anything, but a valid input it will say Invalid Input and give you another try. All inputs must be put in integer / float numbers such as 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9 with no space and a decimal where necessary.

There are also Minimum amounts (for Crediting Account) and Maximum amounts (for Savings Accounts) which upon being violated will prompt the user for another attempt. By default they are $50 minimum for Crediting Accounts and $500 maximum for Savings Accounts. Some users have a custom amount and it will be shown upon violation or if you go into the account information. To avoid these simply withdraw more / less depending on the situation.

If you are broke, you may run into something known as amount is greater than balance simply deposit more than specified amount of money to withdraw this amount.

If you observe / experience any bugs, errors or other problems please do not hesitate to contact the I.T. department within Maxordia Financial.

**Task 6:**

I did not include a separate file for User Requirements or Sign Off as I did not think it was required and would just add confusion.

**Conclusion:**

Hey Eddie, would be happy to come into class in week 17 or 18 if necessary or partake in a video conference. I will give you a short summary here again, but I think I understand all concepts here that were required to create this program. I am looking forward to further developing my coding skills, do you have any idea if we will be continuing Python next semester or swapping to a different language. I am oh so eager to challenge my old friend C++ again. I am sure you will be very busy with other people's projects, especially those who did not attain and reference a relative file location when referring to their text documents. I will be completing the quiz as we speak.

**Teacher Signature:**

| | |
|---|---|
| Name: | Eddie Vanda |
| Signature: | |

**Appendices:**

All files are encompassed within the S3952320 OISIN AEONN PYTHON PROJECT 2 folder, which will be zipped before submission.

Appendix A: __pycache__ - ignore this

Appendix B: BankApp.py – main file with the Bank class

Appendix C: BankingReceipt.txt – stores the output for most recent transaction

Appendix D: CAccountOOP.py – CAccount Class with deposit / withdraw functions

Appendix E: CAccounts.txt - Stores the valid SAccounts in a list

Appendix F: S3952320 OISIN AEONN PYTHON PROJECT 2 REPORT.docx – this file

Appendix F: S3952320 OISIN AEONN PYTHON PROJECT 2 REPORT.pdf – this file

Appendix G: SAccountsOOP.py – SAccount Class with deposit / withdraw functions

Appendix H: SAccounts.txt – Stores the valid SAccounts in a list

**References:**

Previous Python Project 1

Weekly Notes and Presentations

General Internet