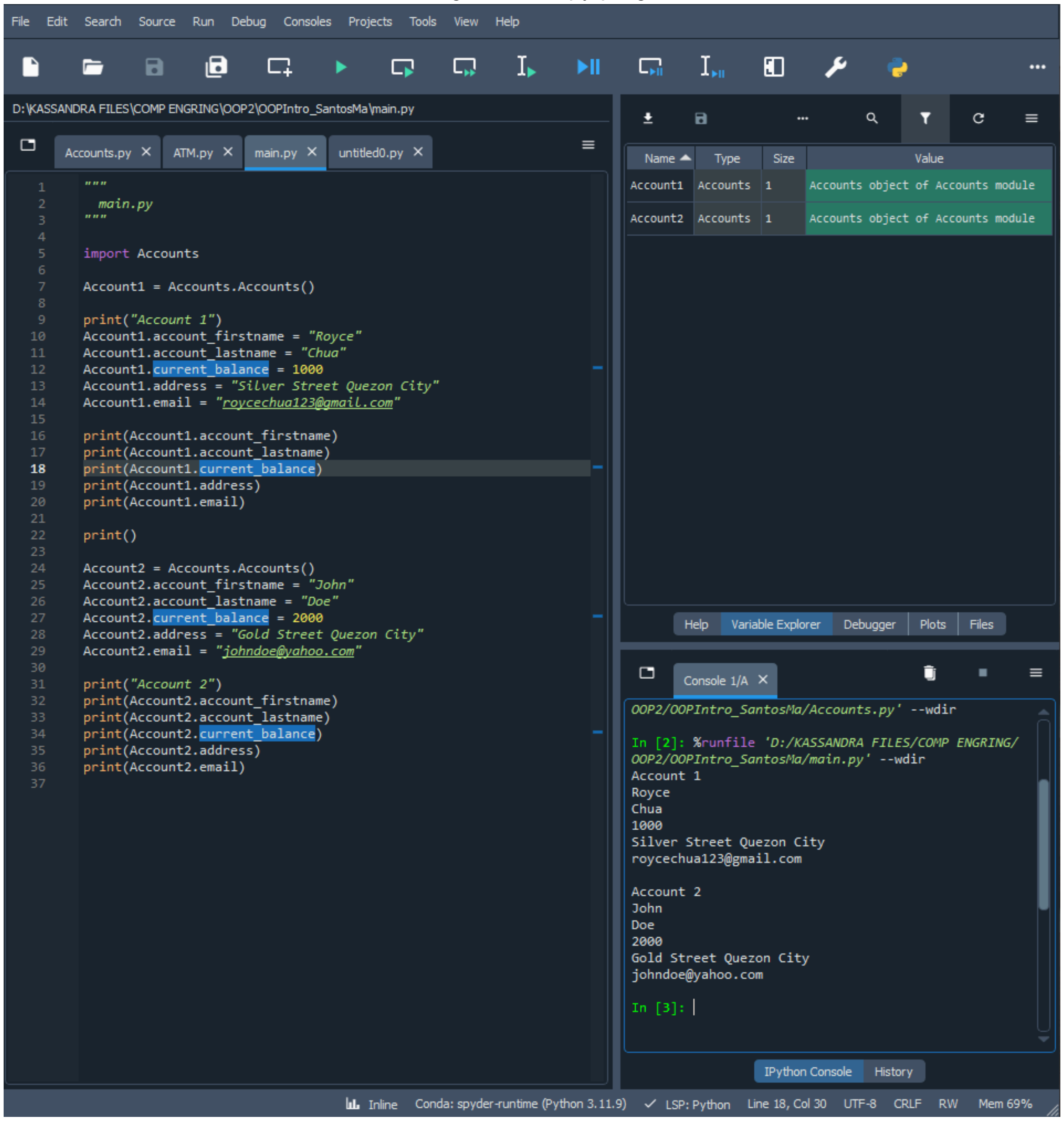
Laboratory Activity No. 1	
Introduction to Object-Oriented Programming	
Santos, Ma. Kassandra Nicole D.	09/14/2024
CPE009B	Ma'am Sayo

5. PROCEDURE

Running the main.py program



D:\KASSANDRA FILES\COMP ENGRING\OOP2\OOPIntro_SantosMa\main.py

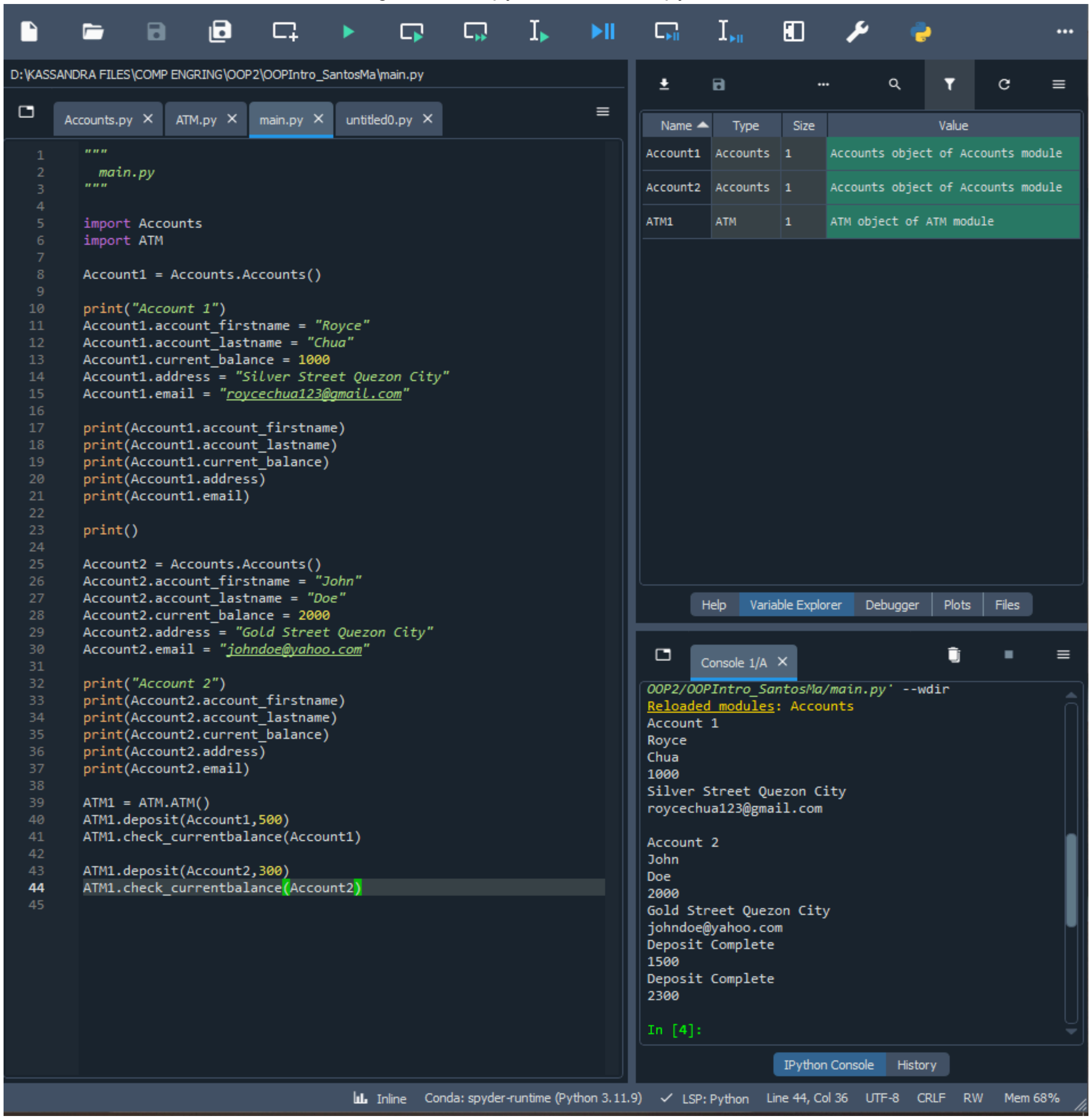
```python
"""
  main.py
"""

import Accounts

Account1 = Accounts.Accounts()

print("Account 1")
Account1.account_firstname = "Royce"
Account1.account_lastname = "Chua"
Account1.current_balance = 1000
Account1.address = "Silver Street Quezon City"
Account1.email = "roycechua123@gmail.com"

print(Account1.account_firstname)
print(Account1.account_lastname)
print(Account1.current_balance)
print(Account1.address)
print(Account1.email)

print()

Account2 = Accounts.Accounts()
Account2.account_firstname = "John"
Account2.account_lastname = "Doe"
Account2.current_balance = 2000
Account2.address = "Gold Street Quezon City"
Account2.email = "johndoe@yahoo.com"

print("Account 2")
print(Account2.account_firstname)
print(Account2.account_lastname)
print(Account2.current_balance)
print(Account2.address)
print(Account2.email)
```

Name	Type	Size	Value
Account1	Accounts	1	Accounts object of Accounts module
Account2	Accounts	1	Accounts object of Accounts module

```
OOP2/OOPIntro_SantosMa/Accounts.py' --wdir

In [2]: %runfile 'D:/KASSANDRA FILES/COMP ENGRING/
OOP2/OOPIntro_SantosMa/main.py' --wdir
Account 1
Royce
Chua
1000
Silver Street Quezon City
roycechua123@gmail.com

Account 2
John
Doe
2000
Gold Street Quezon City
johndoe@yahoo.com

In [3]:
```

File Edit Search Source Run Debug Consoles Projects Tools View Help
Accounts.py ATM.py main.py untitled0.py
Help Variable Explorer Debugger Plots Files
Console 1/A
IPython Console History
Inline Conda: spyder-runtime (Python 3.11.9) LSP: Python Line 18, Col 30 UTF-8 CRLF RW Mem 69%

Running the main.py with the ATM.py included



D:\KASSANDRA FILES\COMP ENGRING\OOP2\OOPIntro_SantosMa\main.py

Accounts.py ATM.py main.py untitled0.py

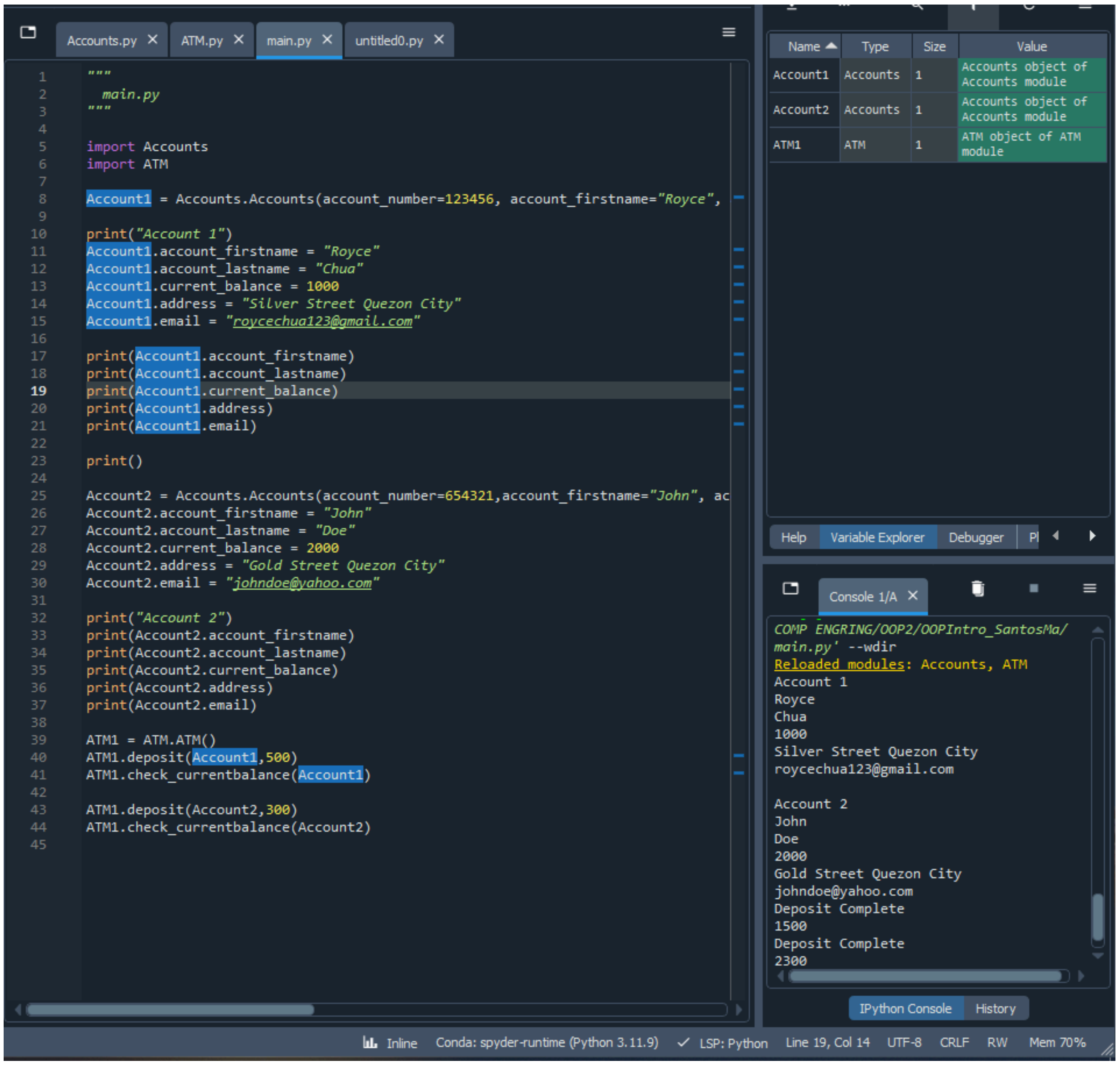
```python
"""
  main.py
"""

import Accounts
import ATM

Account1 = Accounts.Accounts()

print("Account 1")
Account1.account_firstname = "Royce"
Account1.account_lastname = "Chua"
Account1.current_balance = 1000
Account1.address = "Silver Street Quezon City"
Account1.email = "roycechua123@gmail.com"

print(Account1.account_firstname)
print(Account1.account_lastname)
print(Account1.current_balance)
print(Account1.address)
print(Account1.email)

print()

Account2 = Accounts.Accounts()
Account2.account_firstname = "John"
Account2.account_lastname = "Doe"
Account2.current_balance = 2000
Account2.address = "Gold Street Quezon City"
Account2.email = "johndoe@yahoo.com"

print("Account 2")
print(Account2.account_firstname)
print(Account2.account_lastname)
print(Account2.current_balance)
print(Account2.address)
print(Account2.email)

ATM1 = ATM.ATM()
ATM1.deposit(Account1,500)
ATM1.check_currentbalance(Account1)

ATM1.deposit(Account2,300)
ATM1.check_currentbalance(Account2)
```

Name	Type	Size	Value
Account1	Accounts	1	Accounts object of Accounts module
Account2	Accounts	1	Accounts object of Accounts module
ATM1	ATM	1	ATM object of ATM module

```
OOP2/OOPIntro_SantosMa/main.py' --wdir
Reloaded modules: Accounts
Account 1
Royce
Chua
1000
Silver Street Quezon City
roycechua123@gmail.com

Account 2
John
Doe
2000
Gold Street Quezon City
johndoe@yahoo.com
Deposit Complete
1500
Deposit Complete
2300

In [4]:
```

Inline Conda: spyder-runtime (Python 3.11.9) LSP: Python Line 44, Col 36 UTF-8 CRLF RW Mem 68%

Running Python after putting a constructor in each class



Accounts.py ATM.py main.py untitled0.py

```
"""
  main.py
"""

import Accounts
import ATM

Account1 = Accounts.Accounts(account_number=123456, account_firstname="Royce",

print("Account 1")
Account1.account_firstname = "Royce"
Account1.account_lastname = "Chua"
Account1.current_balance = 1000
Account1.address = "Silver Street Quezon City"
Account1.email = "roycechua123@gmail.com"

print(Account1.account_firstname)
print(Account1.account_lastname)
print(Account1.current_balance)
print(Account1.address)
print(Account1.email)

print()

Account2 = Accounts.Accounts(account_number=654321,account_firstname="John", ac
Account2.account_firstname = "John"
Account2.account_lastname = "Doe"
Account2.current_balance = 2000
Account2.address = "Gold Street Quezon City"
Account2.email = "johndoe@yahoo.com"

print("Account 2")
print(Account2.account_firstname)
print(Account2.account_lastname)
print(Account2.current_balance)
print(Account2.address)
print(Account2.email)

ATM1 = ATM.ATM()
ATM1.deposit(Account1,500)
ATM1.check_currentbalance(Account1)

ATM1.deposit(Account2,300)
ATM1.check_currentbalance(Account2)
```

Name	Type	Size	Value
Account1	Accounts	1	Accounts object of Accounts module
Account2	Accounts	1	Accounts object of Accounts module
ATM1	ATM	1	ATM object of ATM module

Help Variable Explorer Debugger

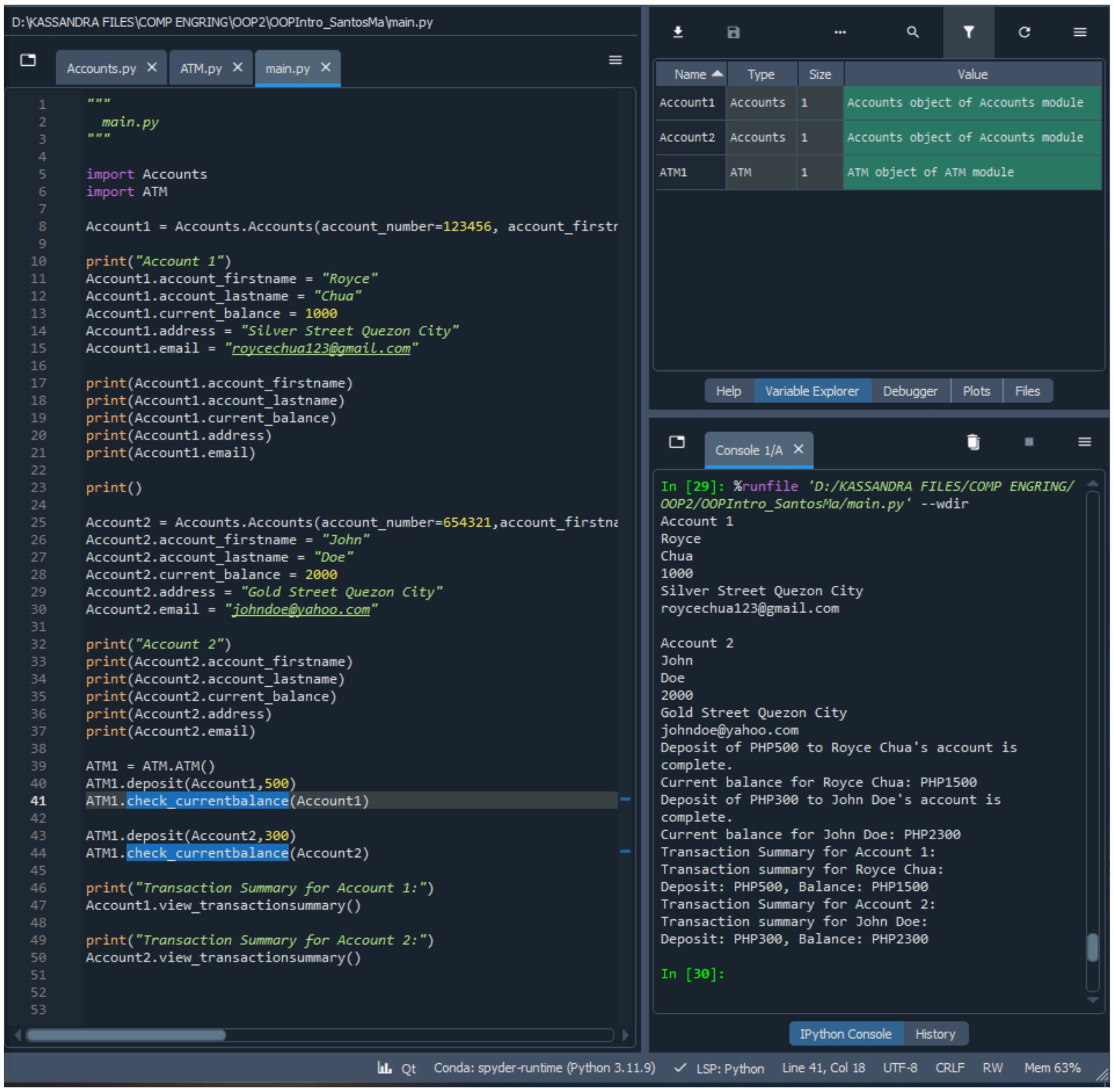
Console 1/A

```
COMP ENGRING/OOP2/OOPIntro_SantosMa/
main.py' --wdir
Reloaded modules: Accounts, ATM
Account 1
Royce
Chua
1000
Silver Street Quezon City
roycechua123@gmail.com

Account 2
John
Doe
2000
Gold Street Quezon City
johndoe@yahoo.com
Deposit Complete
1500
Deposit Complete
2300
```

IPython Console History

Inline Conda: spyder-runtime (Python 3.11.9) ✓ LSP: Python Line 19, Col 14 UTF-8 CRLF RW Mem 70%

Modifying the ATM.py program by adding view_transactionsummary()


D:\KASSANDRA FILES\COMP ENGRING\OOP2\OOPIntro_SantosMa\main.py

Accounts.py | ATM.py | main.py

```python
"""
  main.py
"""

import Accounts
import ATM

Account1 = Accounts.Accounts(account_number=123456, account_firstr

print("Account 1")
Account1.account_firstname = "Royce"
Account1.account_lastname = "Chua"
Account1.current_balance = 1000
Account1.address = "Silver Street Quezon City"
Account1.email = "roycechua123@gmail.com"

print(Account1.account_firstname)
print(Account1.account_lastname)
print(Account1.current_balance)
print(Account1.address)
print(Account1.email)

print()

Account2 = Accounts.Accounts(account_number=654321,account_firstna
Account2.account_firstname = "John"
Account2.account_lastname = "Doe"
Account2.current_balance = 2000
Account2.address = "Gold Street Quezon City"
Account2.email = "johndoe@yahoo.com"

print("Account 2")
print(Account2.account_firstname)
print(Account2.account_lastname)
print(Account2.current_balance)
print(Account2.address)
print(Account2.email)

ATM1 = ATM.ATM()
ATM1.deposit(Account1,500)
ATM1.check_currentbalance(Account1)

ATM1.deposit(Account2,300)
ATM1.check_currentbalance(Account2)

print("Transaction Summary for Account 1:")
Account1.view_transactionsummary()

print("Transaction Summary for Account 2:")
Account2.view_transactionsummary()
```

Name	Type	Size	Value
Account1	Accounts	1	Accounts object of Accounts module
Account2	Accounts	1	Accounts object of Accounts module
ATM1	ATM	1	ATM object of ATM module

Help | Variable Explorer | Debugger | Plots | Files

Console 1/A

```
In [29]: %runfile 'D:/KASSANDRA FILES/COMP ENGRING/
OOP2/OOPIntro_SantosMa/main.py' --wdir
Account 1
Royce
Chua
1000
Silver Street Quezon City
roycechua123@gmail.com

Account 2
John
Doe
2000
Gold Street Quezon City
johndoe@yahoo.com
Deposit of PHP500 to Royce Chua's account is
complete.
Current balance for Royce Chua: PHP1500
Deposit of PHP300 to John Doe's account is
complete.
Current balance for John Doe: PHP2300
Transaction Summary for Account 1:
Transaction summary for Royce Chua:
Deposit: PHP500, Balance: PHP1500
Transaction Summary for Account 2:
Transaction summary for John Doe:
Deposit: PHP300, Balance: PHP2300

In [30]:
```

IPython Console | History

Qt Conda: spyder-runtime (Python 3.11.9) ✓ LSP: Python Line 41, Col 18 UTF-8 CRLF RW Mem 63%

6. QUESTIONS

1. What is a class in Object-Oriented Programming?
    a. class in OOP is like a blueprint or guide for creating an object or an output, class can also be described as a constructor. A constructor in class would be the __init__ which initializes the object attributes and __str__ function when the class object is represented as a string.
2. Why do you think classes are being implemented in certain programs while some are sequential?
    a. Doing a sequential program can be tedious and time-consuming, sometimes if the code is too long, it would look complex and messy. Classes can help reduce time make the code more organized and maximize functionality, not only that, classes can be reusable as well.
3. How is it that there are variables of the same name such as account_firstname and account_lastname that exist but have different values

    a. Its because within the class Account (), there is a def update_firstname) and within that is the self.account_firstname = new_firstname. This function can help update or change the information within account1 without overlapping any other account objects.

4. Explain the constructor functions role in initializing the attributes of the class. When does the constructor function execute or when is the constrictor function called?
    a. As said before class in OOP can be described as a constructor as it is a blueprint on how to create an object. the constructor function normally named as __init__ short for initialization. This method helps to set up the initial state of the object by assigning attributes such as the first name, last name, and more.
5. Explain the benefits of using Constructors over initializing the variables one by one in the main program?
    a. The benefits of using constructors are to help initialize details within a class, keep the whole code clear and concise, and be easily manageable.

7. CONCLUSION

a. Class in OOP is a type of constructor that can be described as a blueprint of a code. Classes can help in having a more manageable and simpler code script. __init__ is a type of code to help initialize an object by putting an attribute in it. This practice helped me better understand how constructor works both in the world of python and as well as c++, I got to practice how to initialize attributes within a class which made me realize how easy it is to construct a class.