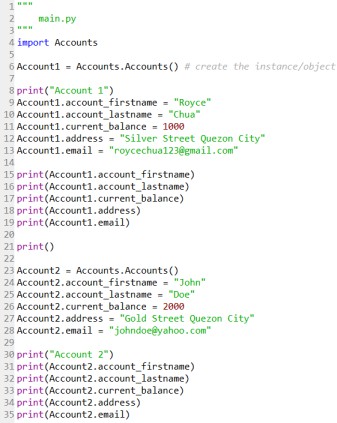
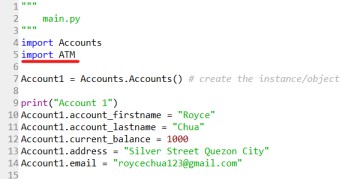
|  |  |
| --- | --- |
| **Laboratory Activity No. 3.1** | |
| **Introduction to Object-Oriented Programming** | |
| **Course Code:** CPE103 | **Program:** BSCPE |
| **Course Title:** Object-Oriented Programming | **Date Performed:** 01/29/2025 |
| **Section:** 1 – A | **Date Submitted:** 01/29/2025 |
| **Name:** Hermosura, Leigh B. | **Instructor:** Maria Rizette M. Sayo |
| **1. Objective(s):** | |
| This activity aims to familiarize students with the concepts of Object-Oriented Programming | |
| **2. Intended Learning Outcomes (ILOs):** | |
| The students should be able to:   * 1. Identify the possible attributes and methods of a given object   2. Create a class using the Python language   3. Create and modify the instances and the attributes in the instance. | |
| **3. Discussion:** | |
| Object-Oriented Programming (OOP) is an approach to programming that views the world and systems as consisting of objects that relate and interact with each other. This involves identifying the characteristics that describe the object which are known as the Attributes of the object. Furthermore, it also deals with identifying the possible capabilities or actions that an object is able to do which are called Methods.  An object is simply composed of Attributes and Methods wherein Attributes are variables that hold the information describing the object and Methods are functions which allow the object to perform its defined capabilities/actions. A UML Class Diagram is used to formally represent the collection of Attributes and Methods.  An example is given below considering a simple banking system.  Accounts ATM  + account\_number: int + serial\_number: int  + account\_firstname: string  + account\_lastname: string  + current\_balance: float  + address: string + deposit(account: Accounts, amount: int) + email: string + widthdraw(account: Accounts, amount: int) + update\_address(new\_address: string) + check\_currentbalance(account: Accounts) + update\_email(new\_email: string) + view\_transactionsummary() | |
| **4. Materials and Equipment:** | |

|  |
| --- |
| Desktop Computer with Anaconda Python/Python Colab Windows Operating System |
| **5. Procedure:** |
| **Creating Classes**   1. Create a folder named **OOPIntro\_LastName** 2. Create a Python file inside the **OOPIntro\_LastName** folder named **Accounts.py** and copy the code shown below: |

|  |
| --- |
| 1. Modify the Accounts.py and add ***self,*** before the new\_address and new\_email. 2. Create a new file named ATM.py and copy the code shown below:     **Creating Instances of Classes**   1. Create a new file named main.py and copy the code shown below: |

6.

Run the main.py program and observe the output. Observe the variables names account\_firstname, account\_lastname as well as other variables being used in the Account1 and Account2. 7. Modify the main.py program and add the code underlined in red.

8. Modify the main.py program and add the code below line 38.

|  |
| --- |
| 9. Run the main.py program.  **Create the Constructor in each Class**   1. Modify the Accounts.py with the following code:   Reminder: def init (): is also known as the constructor class   1. Modify the main.py and change the following codes with the red line. Do not remove the other codes in the program. |

|  |
| --- |
| 3. Run the main.py program again and run the output. |
| **6. Supplementary Activity:** |
| **Tasks**   1. Modify the ATM.py program and add the constructor function.   For number 1, please refer to this link <https://colab.research.google.com/drive/1kfxYo6KaHkFa3LPBM97alUg76PKwA2hG#scrollTo=clU9zlvdG_h2&line=10&uniqifier=1>   1. Modify the main.py program and initialize the ATM machine with any integer serial number combination and display the serial number at the end of the program.   For number 2, please refer to this link <https://colab.research.google.com/drive/1kfxYo6KaHkFa3LPBM97alUg76PKwA2hG#scrollTo=JD1dQT3oKv17&line=10&uniqifier=1>   1. Modify the ATM.py program and add the **view\_transactionsummary()** method. The method should display all the transaction made in the ATM object.   For number 3, please refer to this link <https://colab.research.google.com/drive/1kfxYo6KaHkFa3LPBM97alUg76PKwA2hG#scrollTo=odaQ703fP2Bs&line=4&uniqifier=1>  **Questions**   1. What is a class in Object-Oriented Programming?   A class offers a way to group data and functionality into a single unit. When a class is created, it defines a new object type, allowing multiple instances of that type to be created.   1. Why do you think classes are being implemented in certain programs while some are sequential(line-by-line)?   My opinion is that it’s a matter of where the programmer is comfortable with and how complex the program is. Classes are typically used in programs that require reusability and better structure while coding sequentially is often used on simpler tasks.   1. How is it that there are variables of the same name such account\_firstname and account\_lastname that exist but have different values? |

These variables share the same name because they represent the instance attributes of a certain object of the class. Each time an object is created within a class, a separate copy for these variables is created and stored inside that object.

|  |
| --- |
| 4. Explain the constructor functions role in initializing the attributes of the class? When does the Constructor function execute or when is the constructor function called?  The primary role of constructor functions is to initialize the attributes of the object with specific values. It ensures each object has its own independent data and allows for the customization of each instance. It’s called instantly upon the creation of an object which means it’s automatically runs without the need to call for it directly. |

|  |
| --- |
| 5. Explain the benefits of using Constructors over initializing the variables one by one in the main program?  It makes the block of code shorter and less of a hassle to store multiple attributes compared to listing them one by one and also improves the code’s readability by reducing redundancy. |
| **7. Conclusion:** |
| Programming tends to be more organized and reusable when classes are used, particularly for complex applications. In Object Oriented Programming (OOP), classes enable a more structural code compared to sequential programming which follows a line-by-line approach. The constructor function automatically initializes attributes when an object is created within a class. Lastly, instance attributes allow multiple variables that have same names but different values exist in one program. In short, the use of constructor functions and classes makes for a more maintainable, efficient and readable code which is essential for more complex and larger applications. |
| **8. Assessment Rubric:** |