SSN College of Engineering

Department of Information Technology

UIT2312 – Programming and Design Patterns Laboratory

2023 – 2024

Mid-Semester Assessment

November 22, 2023

Time: 10:05 am to 12:35 pm **Marks:** 100

Read the following instructions carefully:

1. Demonstrate the OOPs paradigm to solve the problem.
2. Draw the class diagram for the given problem in the paper and submit during the end of the session.
3. Upload all the python files and the digital diary with relevant output/ error screenshots in the LMS page.
4. Marks split-up:

Design (class diagram, class relation diagram, description of member variables & methods)	Coding (Coding standards, Coding process)	Execution and Testing (test-driven coding, Quality of test cases, Correctness of results)	Viva-Voce	Total
15	45	30	10	100

Design and implement a comprehensive Online Banking System using Python. Apply object-oriented principles such as classes, inheritance, and polymorphism to model different entities like Account, SavingsAccount, CheckingAccount, and their relationships. Address a specific challenge related to the banking domain, such as handling different types of transactions with unique attributes. Utilize serialization techniques to persistently store and retrieve account and transaction data, ensuring data integrity across sessions. Implement string operations and regular expressions for efficient data manipulation, facilitating tasks like searching, filtering, and categorizing transactions based on their attributes. Organize your code into modular packages, including classes for Transactions, Customer, and a central Banking System.

List of tasks to be accomplished

Classes, Inheritance, and Polymorphism:

- Define a base class Account with attributes like account number, balance, and methods like display_info.

- Utilize inheritance to create subclasses like SavingsAccount and CheckingAccount, inheriting from the Account class.
- Implement polymorphism by overriding methods like display_info in the subclasses to provide specific information.

Transaction Handling Challenge:

- Create a scenario where different types of transactions (e.g., Deposit, Withdrawal) have unique attributes.
- Implement classes for these transactions and show how to handle them within the overall system.

Serialization:

- Implement serialization for storing and retrieving account and transaction data.
- Use a module like pickle or json for serialization.
- Ensure that serialized data retains the object's structure and state.

String Operations and Regular Expressions:

- Implement string operations for tasks like searching and filtering transactions based on their attributes.
- Use string methods for operations like searching by transaction type or filtering by date.
- Apply regular expressions for more complex pattern matching tasks.

Packages:

- Organize code into modular packages, including classes for Transactions, Customer, and a central Banking System.

Rubrics for coding and testing	Remarks
Classes and objects created	
Demonstration of inheritance, abstract methods, RegEx, MRO	
Implementation of serializability	
Package creation and installation	
