**FINANCIAL TRANSACTION ANALYSIS IN TPBANK**

**CSD201 - VUBC**

*In this exercise, you will create an object-oriented program for TPBank to analyze a series of financial transactions for a user's bank account. You will need to choose appropriate data structures, analyze the time complexities of your chosen operations, and incorporate a* ***User*** *class to represent the account holder.*

**Problem Statement:** You are working at TPBank and tasked with analyzing a series of financial transactions for a user's bank account. Each transaction is represented as an object with attributes for date, amount, and type (deposit or withdrawal). Your goal is to create an OOP program that efficiently processes and analyzes these transactions while also representing the user's account.

Create a ***User*** class with attributes such as ***name, account\_number, and balance***. Implement appropriate getters and setters for these attributes.

Create a ***Transaction*** class with attributes for ***date (as a string), amount (as a float), and type (as a string indicating 'deposit' or 'withdrawal')***. Implement appropriate getters and setters.

Initialize a list of 50 random ***Transaction*** objects. The transactions should have random dates, amounts ranging from -10,000 to 10,000, and random types (deposit or withdrawal).

Choose appropriate data structures to store and manage the list of transactions for the user's account. Consider factors like insertion, retrieval, and space complexity. Document your reasons for choosing these data structures.

Create a ***TransactionAnalyzer*** class that takes the list of transactions and the User object as input and provides the following methods:

* ***calculate\_balance\_history***: Calculate and return a list of account balances after each transaction.
* ***find\_min\_max\_balance***: Find and return the minimum and maximum account balances during the transaction history.
* ***find\_top\_deposits\_and\_withdrawals***: Find and return lists of the top three deposits and the top three withdrawals.
* ***find\_mean\_transaction***: Find and return the mean transaction amount for deposits and for withdrawals.
* ***find\_most\_frequent\_transactions***: Find and return the most frequent transaction amount(s).

Analyze and document the time complexities (Big O notation) of each operation in your program, such as insertion, retrieval, and sorting.

Additional Instructions:

* Implement the ***User***, ***Transaction***, and ***TransactionAnalyzer*** classes with appropriate methods and data structures.
* Use encapsulation and proper attribute access control in your classes.
* Test your program with different sets of random financial transactions to ensure correctness.