Banking System using Python

The given Python code simulates a basic banking system where users can perform operations such as creating an account, logging in, depositing money, withdrawing money, transferring funds between accounts, and viewing transaction histories. The system is structured around two main classes: **BankAccount** and **BankingSystem.**

The **BankAccount** class represents an individual bank account and handles the core operations for managing an account's balance. It includes methods for depositing funds, withdrawing funds, transferring money to another account, and displaying transaction history. Each account keeps track of its own transaction history, such as deposits, withdrawals, and transfers. Additionally, the class ensures that only valid transactions (like positive deposits and sufficient funds for withdrawals) are processed.

The **BankingSystem** class acts as a management system for multiple bank accounts. It can create new accounts, log in users, and facilitate money transfers between different accounts. When a user log in, they can perform a series of operations such as depositing money, withdrawing, checking their balance, transferring money to another account, and viewing their transaction history. The code includes basic input validation and feedback, ensuring that users can only transfer money to existing accounts and preventing invalid operations like withdrawing more than the available balance.

The **main()** function acts as the entry point for the program, displaying a menu where users can create an account, log in, or exit the system. Upon logging in, users can perform various banking operations in a loop until they choose to log out. The program also supports the creation of multiple accounts and manages the data for all accounts in a centralized way using the **`BankingSystem`** class. This setup provides a simple but functional simulation of a banking system using core Python OOPs concepts like classes, methods, and loops.