Documentation Report for Bank Account Management System

Project Overview

The Bank Account Management System is a Java-based program that simulates banking operations for two types of accounts: Savings Account and Current Account. It provides functionalities such as depositing money, displaying balance, computing and depositing interest (for savings accounts), withdrawing money, and ensuring minimum balance compliance (for current accounts).

- **The program utilizes inheritance to model the relationship between generic accounts (Account) and their specific types (SavAcct and CurrAcct).**

Objectives

Model real-world banking features using object-oriented programming (OOP) concepts.

Use inheritance to create specialized classes for savings and current accounts.

Provide interactive functionality to handle customer transactions.

Implement error handling to ensure data validity and user-friendly error reporting.

Features

1. Savings Account

Compound interest computation and deposit.

Deposit and withdrawal capabilities.

No minimum balance requirement.

2. Current Account

Minimum balance enforcement with penalty for non-compliance.

Deposit and withdrawal capabilities.

No interest computation.

3. Generic Functionalities

Balance display.

Transaction logging through console messages.

Exception handling for invalid inputs or operations.

Class Structure and Methods

1. Class: Account

Purpose: Acts as a base class for all types of accounts. Stores common attributes and methods.

Attributes:

customerName: Name of the account holder.

accountNumber: Unique identifier for the account.

accountType: Type of the account (Savings/Current).

balance: Current account balance.

Methods:

initialize(): Initializes account details.

deposit(double amount): Adds money to the balance. Throws exception for invalid inputs.

displayBalance(): Displays the current balance.

withdraw(double amount): Deducts money from the balance. Handles insufficient funds.

2. Class: SavAcct

Purpose: Represents a savings account.

Attributes:

interestRate: Fixed annual interest rate (4%).

Methods:

computeAndDepositInterest(): Computes interest on the balance and adds it.

3. Class: CurrAcct

Purpose: Represents a current account.

Attributes:

minimumBalance: Required minimum balance (500).

penalty: Penalty for falling below minimum balance (50).

Methods:

checkMinimumBalance(): Checks if balance meets the minimum requirement and applies penalty if not.

4. Class: BankAccountManagement

Purpose: Entry point for the program and contains the main logic for user interaction.

Methods:

manageSavingsAccount(): Handles savings account operations via a menu-driven interface.

manageCurrentAccount(): Handles current account operations via a menu-driven interface.

Implementation Details

Key Concepts Used

1. Inheritance:

SavAcct and CurrAcct extend Account to inherit common attributes and methods.

2. Encapsulation:

Class attributes are protected via public methods.

3. Exception Handling:

Ensures user inputs and operations are valid.

Provides meaningful error messages for invalid cases.

Menu-Driven Interface

The program uses a menu system for each account type, allowing users to:

1. Deposit money.

2. Display the current balance.

3. Compute and deposit interest (savings account only).

4. Check minimum balance and apply penalties (current account only).

5. Withdraw money.

Input Validation

Deposit Amount: Must be positive.

Withdrawal Amount: Must be positive and less than or equal to the current balance.

Sample Interaction

Savings Account Menu:

Enter account type (Savings/Current): Savings
Enter customer name: John
Enter account number: 12345
Enter initial balance: 1000

1. Deposit
2. Display Balance
3. Compute Interest
4. Withdraw
5. Exit

Enter your choice: 1
Enter amount to deposit: 500
Deposit successful! New balance: 1500

Current Account Menu:

Enter account type (Savings/Current): Current
Enter customer name: Jane
Enter account number: 54321
Enter initial balance: 450

1. Deposit
2. Display Balance
3. Check Minimum Balance
4. Withdraw
5. Exit
Enter your choice: 3
Balance below minimum! Penalty of 50 imposed. New balance: 400

Exception Handling

Invalid Deposit or Withdrawal Amounts:

Throws IllegalArgumentException for non-positive amounts.

Insufficient Balance for Withdrawal:

Throws IllegalStateException.

General Input Errors:

Catches Exception for unexpected issues during input parsing.

Limitations

No persistence: Account data resets after program termination.

No multi-user support: Only one account can be managed per session.

Limited features: Real-world banking operations like account creation and deletion are not implemented.

Future Enhancements

1. Database Integration:

Store account data in a database for persistence and multi-user support.

2. GUI Implementation:

Replace the console-based interface with a graphical user interface.

3. Additional Account Types:

Add support for more account types (e.g., Fixed Deposit, Business Accounts).

4. Advanced Features:

Include functionality for cheque issuance and processing.

Conclusion

The Bank Account Management System successfully models basic banking operations for savings and current accounts. It demonstrates key object-oriented programming principles and highlights the importance of exception handling and input validation in software development.