Project 1 (Day 1): Basic Account Class

Description

Students will create a foundational `Account` class that represents a bank account. This class should include basic operations such as depositing money, withdrawing funds, and checking the account balance. The goal is to practice encapsulation and basic method writing.

Attributes

- **accountNumber** (String): A unique identifier for the account.
- **accountHolder** (String): The name (or ID) of the account holder.
- **balance** (double): The current monetary balance in the account.

Methods

1. **deposit(amount)**
    - **Purpose:** Add a specified amount to the balance.
    - **Parameters:** `amount` (double) – Must be a positive number.
    - **Return:** No value (void).
    - **Behavior:** Increase `balance` by `amount` if `amount > 0`.
2. **withdraw(amount)**
    - **Purpose:** Subtract a specified amount from the balance if sufficient funds exist.
    - **Parameters:** `amount` (double) – Must be positive and less than or equal to the current balance.
    - **Return:** Boolean – `true` if the withdrawal is successful, `false` otherwise.
    - **Behavior:** Decrease `balance` by `amount` if there is enough balance.
3. **getBalance()**
    - **Purpose:** Retrieve the current balance.
    - **Return:** The current `balance` (double).

Manual Test Cases (Using if/else Logic)

- **Test Case 1: Deposit Valid Amount**
    - **Action:** Create an account with an initial balance of 1000. Call `deposit(500)`.

    - **If/Else Check:**
        - **If** the new balance equals 1500, **then** print “Deposit successful.”
        - **Else** print “Error: Deposit did not update balance correctly.”

- **Test Case 2: Withdraw Valid Amount**
    - **Action:** Using the account with balance 1500, call `withdraw(300)`.
    - **If/Else Check:**
        - **If** `withdraw(300)` returns `true` and the new balance equals 1200, **then** print “Withdrawal successful.”
        - **Else** print “Error: Withdrawal failed or balance incorrect.”

- **Test Case 3: Withdraw Exceeding Balance**
    - **Action:** Attempt to withdraw 2000 from the account with balance 1200.
    - **If/Else Check:**
        - **If** `withdraw(2000)` returns `false`, **then** print “Properly handled insufficient funds.”
        - **Else** print “Error: Withdrawal allowed without sufficient funds.”

- **Test Case 4: Deposit Negative Amount**
    - **Action:** Attempt to deposit a negative value (e.g., `deposit(-100)`).
    - **If/Else Check:**
        - **If** the balance remains unchanged, **then** print “Negative deposit rejected.”
        - **Else** print “Error: Negative deposit affected the balance.”

Project 2 (Day 2): Extending Account – SavingsAccount & CurrentAccount

Description

Extend the basic `Account` class by creating two specialized account types:

- **SavingsAccount:** Includes an interest rate and a method to calculate interest.
- **CurrentAccount:** Includes an overdraft limit and an overridden withdrawal method that permits overdraft up to a limit.

Additional Attributes

- **For SavingsAccount:**

- **interestRate** (double): The interest rate (e.g., 0.05 for 5%).

- **For CurrentAccount:**
  - **overdraftLimit** (double): The extra amount allowed for withdrawal beyond the current balance.

Additional Methods

1. **SavingsAccount: calculateInterest()**
   - **Purpose:** Compute the interest based on the current balance.
   - **Return:** The interest amount (double) calculated as `balance * interestRate`.
2. **CurrentAccount: withdraw(amount)**
   - **Purpose:** Override the withdrawal to allow withdrawing more than the current balance up to `balance + overdraftLimit`.
   - **Return:** Boolean – `true` if the withdrawal is successful, `false` if the requested amount exceeds this limit.

Manual Test Cases (Using if/else Logic)

- **Test Case 1: SavingsAccount Interest Calculation**
  - **Action:** Create a SavingsAccount with an initial balance of 1000 and an interest rate of 0.05.
  - **If/Else Check:**
    - **If** calling `calculateInterest()` returns 50 (i.e., 1000 * 0.05), **then** print “Interest calculated correctly.”
    - **Else** print “Error: Incorrect interest calculation.”
- **Test Case 2: CurrentAccount Withdrawal Within Overdraft Limit**
  - **Action:** Create a CurrentAccount with a balance of 1000 and an overdraft limit of 500. Attempt to withdraw 1300.
  - **If/Else Check:**
    - **If** `withdraw(1300)` returns `true` and the new balance is (1000 - 1300) = -300, **then** print “Overdraft withdrawal successful.”
    - **Else** print “Error: Withdrawal within overdraft limit failed.”
- **Test Case 3: CurrentAccount Withdrawal Exceeding Limit**
  - **Action:** From the same account, attempt to withdraw an amount that exceeds (balance +

overdraftLimit), for example, 2000.

- **If/Else Check:**
  - **If** `withdraw(2000)` returns `false`, **then** print “Properly rejected excessive withdrawal.”
  - **Else** print “Error: Excessive withdrawal was allowed.”