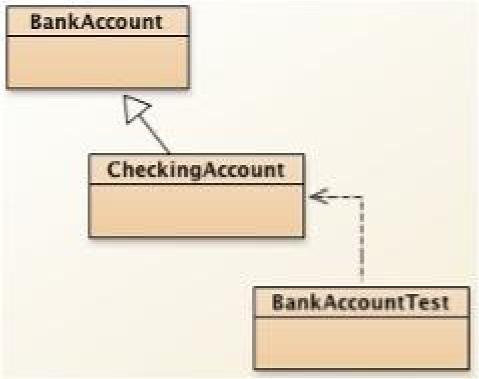
L06 - Bank Account Lab

In this lab, we'll implement three classes: `BankAccount`, `CheckingAccount`, and `BankAccountTest`. As we write these classes, we will be taking advantage of Java's inheritance and polymorphism capabilities to make our job easier.

BankAccount Class

This is the parent class. The `CheckingAccount` will be derived from this class.



Member variables

`double balance`

Constructors

`public BankAccount()`

- creates a `BankAccount` with a zero balance

`public BankAccount(double initialBalance)`

- creates a BankAccount with a value of initial balance

Methods

`public void deposit(double amount)`

- increases the balance by amount

`public void withdraw(double amount)`

- decreases the balance by amount

`public double getBalance()`

- returns the current balance

BankAccount.java

Download this file from Drive, it’s in L06F.zip and fill in the blanks with the missing code.

```
/**
Parent BankAccount class
@author __________
@version __________
*/
public class BankAccount {
  __________ double balance;

  public BankAccount(){
    balance = 0;
  }

  public BankAccount(double initialBalance) {
    balance = __________;
  }

  public __________ deposit(double amount) {
    balance = __________ + amount;
  }



  public void withdraw(double amount) {
    balance = __________ - amount;
  }

  public double getBalance() {
    return __________;
  }
}
```

CheckingAccount Class

This class extends the BankAccount class.

Member variables

`int transactionCount`

- the number of transactions accumulated over a month

Constructors

`public CheckingAccount()`

- creates a checking account with a zero balance and sets transactionCount to zero.

`public CheckingAccount(double initialBalance)`

- creates a checking account and with a balance of initialBalance and sets transactionCount to zero.

Methods

`public void deposit(double amount)`

- increases the balance by amount and increases the transactionCount by 1.

`public void withdraw(double amount)`

- decreases the balance by amount and increases the transactionCount by 1.

`public void deductFees()`

- deducts the accumulated transaction fees and resets the transactionCount to 0. If there are more than three transactions, then the fee is equal to 2.00*(transactionCount – 3). Note that the first three transactions are free and that the fee per transaction is $2.00.

CheckingAccount.java

Download this file from Drive, it’s in L06F.zip and fill in the blanks with the missing code.

```
/**
Child CheckingAccount class
@author __________
@version __________
*/
public class CheckingAccount __________ BankAccount {
  __________ int transactionCount;
  public CheckingAccount() {
    transactionCount = 0;
  }

  public CheckingAccount(double initialBalance) {
    super(__________);
    //use the super class constructor that's already coded
    transactionCount = 0;
  }

  public __________ deposit(double amount) {
    super.deposit(amount);
    transactionCount = __________ + 1;
  }

  public void withdraw(double amount) {
    __________.withdraw(amount);
    transactionCount = __________ + 1;
  }

  public void deductFees() {
    if (transactionCount > 3) {
      double fee = 2.0*(transactionCount – 3);
      __________.withdraw(fee);
    }
    transactionCount = 0;
  }
}
```

Testing Our Classes

To test our two classes, we'll write a BankAccountTest class that performs the following:

- Instantiates a checking account with an initial balance specified by the user
- Presents a menu with the following options:
  - D - Deposit money into the checking account (prompt the user for amount)
  - W - Withdraw money from the checking account (prompt user for amount)
  - P - Process end-of-month activities (withdraw fees for the checking account)
  - S - Show the current checking account balance

    - E - Exit the program

BankAccountTest.java

Download this file from Drive, it’s in L06F.zip and fill in the blanks with the missing code.

```
import java.util.*;
/**
BankAccount test class
@author __________
@version __________
*/
public class BankAccountTest {
  public static void main(String[] args) {
    Scanner myScanner = new Scanner(System.in);
    System.out.print("Enter initial checking account balance: ");
    double amount = Double.parseDouble(myScanner.nextLine());
    CheckingAccount myChecking = new CheckingAccount(amount);

    System.out.println("D - Deposit into checking");
    System.out.println("W - Withdraw from checking");
    System.out.println("P - End of month processing");
    System.out.println("S - Show account balance");
    System.out.println("E - Exit the program");

    boolean done = false;

    while (!done) {
      System.out.print("> ");
      String choice = myScanner.nextLine();

      if (choice.equalsIgnoreCase("D")) {
        System.out.print("Enter amount to deposit: ");
        amount = Double.parseDouble(myScanner.nextLine());
        myChecking.deposit(amount);
      }
      else if (choice.equalsIgnoreCase("W")) {
        System.out.print("Enter amount to withdraw: ");
        amount = Double.parseDouble(myScanner.nextLine());
        myChecking.withdraw(amount);
      }
      else if (choice.equalsIgnoreCase("P"))
        myChecking.deductFees();
      else if (choice.equalsIgnoreCase("S"))
        System.out.println("Checking balance: " + myChecking.getBalance());
      else if (choice.equalsIgnoreCase("E"))
        done = true;
      else
        System.out.println("Invalid menu choice - please re-enter");
    }

    System.out.println("Goodbye!");
  }
}
```

Revision #1
Created 24 April 2025 22:10:53 by Brandon Duke
Updated 24 April 2025 22:11:59 by Brandon Duke