*ProgramAsignment2*

**Source code**

*BankAccount.cs*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using static System.Console;

namespace ProgramAssignment2

{

internal class BankAccount

{

private string id; // only set using constructor, provide an accessor method

private string firstName; // need to provide accessor and mutator methods

private string lastName; // need to provide accessor and mutator methods

private double balance; // Only set using constructor, provide an accessor method

private int numTransactions; // Initialized to zero in constructor, provide an accessor method

// Please provide Accessors and/or Mutator methods as well as using at least one Property method (for the fields and rules above)

// Please a few constructors... default constructor as well as a few parameterized Constructors:

/\* Note: if balance is not specified in constructor, it should be initializer to zero

Note: All constructors shall set numTransactions to zero

Constructor example: id, First and Last name, & balance

\*/

public string Id

{

get

{

return id;

}

}

public string FirstName

{

get

{

return firstName;

}

set

{

firstName = value;

}

}

public string LastName

{

get

{

return lastName;

}

set

{

lastName = value;

}

}

public double Balance

{

get

{

return balance;

}

}

public double NumTransactions

{

get

{

return numTransactions;

}

}

public BankAccount()

{

this.numTransactions = 0;

}

public BankAccount(string ID, string fname, string lname, double Balance = 0)

{

this.id = ID;

this.firstName = fname;

this.lastName = lname;

this.balance = Balance;

this.numTransactions = 0;

}

// Additional methods

// Deposit - Adds amount to balance. Also counts as 1 transaction.

public void deposit(double amount) // Note verify amount is positive and between 1..1000 (inclusive)

{

if (amount >= Constants.MIN && amount <= Constants.MAX)

{

this.balance += amount;

this.numTransactions++;

}

else

WriteLine("Cannot be done!");

}

// Subtracts amount from balance if user has enough money. Counts as 1 transaction.

public void withdraw(double amount)

{

if (this.balance >= amount)

{

this.balance -= amount;

this.numTransactions++;

}

else

WriteLine("You have less money then you want withdraw :(");

}

public string ReturnFullName()

{

return (firstName + " " + lastName);

}

public string toString() // This will return a string containing the account info: F and L name/ID and balance and num transactions.

{

return "Account Number: " + id +

"\nName: " + ReturnFullName() +

"\nBalance: " + balance + "$" +

"\nNumber of transactions: " + numTransactions;

}

}

}

*BankAccountApp.cs*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using static System.Console;

namespace ProgramAssignment2

{

internal class BankAccountApp

{

static void Main(string[] args)

{

WriteLine("First account" + "\n");

BankAccount accountJohnC = new BankAccount("12345", "John", "C", 50.00);

// Creates an account with $50 initial deposit for John C. Note that numTransactions should be set to zero as well.

accountJohnC.deposit(10.00);

accountJohnC.deposit(50.00);

accountJohnC.deposit(10.00);

accountJohnC.deposit(70.00);

accountJohnC.withdraw(100.00);

accountJohnC.LastName = "Cool";

WriteLine(accountJohnC.toString() + "\n");

//Account info: John Cool, 12345 has a Balance = $90, with 5 transactions

WriteLine("Second account" + "\n");

BankAccount account2 = new BankAccount("54321", "Ivan", "Chukharev");

account2.deposit(5000);

WriteLine(account2.toString());

ReadKey();

}

}

}

*Constants.cs*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ProgramAssignment2

{

public class Constants

{

public const int MAX = 1000;

public const int MIN = 1;

}

}

**Outputs**

Text

Description automatically generated

Graphical user interface, application

Description automatically generated

Text

Description automatically generated with medium confidence