* Vinicius Gribel Person

MAIN

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using ContaBancaria;

namespace ContaBancaria

{

class Program

{

static public void InvalidInt()// function to show a message error

{

Console.WriteLine("Please, correctly insert the fields");

Console.WriteLine("Press any key to continue");

Console.ReadKey();

Console.Clear();

}

static public int Menu()

{ // menu

while (true)

{ // while to make infinity menu in case of error

try

{

Console.WriteLine("\nChose a option:" +

"\n[1] - Deposit" +

"\n[2] - WithDraw" +

"\n[3] - Show data of account" +

"\n[4] - Change a limit of WithDraw" +

"\n[5] - Exit" +

"\nPlease select a option");

int opc = int.Parse(Console.ReadLine());

if(opc >= 1 || opc <= 3)

{

return opc;

}

else

{

Console.WriteLine("\nPlease, select a option between 1 and 3");

Console.WriteLine("Press any key to continue");

Console.ReadKey();

Console.Clear();

continue;

}

}

catch (FormatException)

{

InvalidInt();

continue;

}

catch(Exception e)

{

Console.WriteLine(e);

}

}

}

static void Main(string[] args)

{

CurrentAccount account = new CurrentAccount();

while (true)

{// while to make infinity menu in case of error

try

{

Console.Write("Input your number of account: ");

account.NumberAccount1 = int.Parse(Console.ReadLine());

Console.Write("Name: ");

account.Name1 = Console.ReadLine();

Console.Write("Begin Balance: $");

account.Balance1 = double.Parse(Console.ReadLine());

Console.Write("Limit of With Draw: $");

account.LimitWithDraw1 = double.Parse(Console.ReadLine());

Console.WriteLine("\n\nPress any key to continue");

Console.ReadKey();

Console.Clear();

break;

}

catch (FormatException)

{

InvalidInt();

continue;

}

catch(Exception e)

{

Console.WriteLine(e);

continue;

}

}

while (true)

{// while to make infinity menu

int menu = Menu();

switch (menu)

{

case 1: // deposit switch

{

account.Deposit();

break;

}

case 2: // withDraw Switch

{

account.WithDraw();

break;

}

case 3: // show data switch

{

account.ShowData();

break;

}

case 4: // change limit switch

{

account.ChangeLimit();

break;

}

case 5: // exit switch

{

Console.WriteLine("\nLeaving the program");

System.Environment.Exit(0);

break;

}

}

}

}

}

}

Class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using ContaBancaria;

namespace ContaBancaria

{

class CurrentAccount

{

private string Name;

private int NumberAccount;

private double Balance;

private double LimitWithDraw;

// gets e seets

public string Name1 { get => Name; set => Name = value; }

public int NumberAccount1 { get => NumberAccount; set => NumberAccount = value; }

public double Balance1 { get => Balance; set => Balance = value; }

public double LimitWithDraw1 { get => LimitWithDraw; set => LimitWithDraw = value; }

static public void InvalidInt()

{// function of modulation

Console.WriteLine("Please, correctly insert the fields");

Console.WriteLine("Press any key to continue");

Console.ReadKey();

Console.Clear();

}

public void Deposit()

{

try

{

Console.Write("\nInput a value what do you want to deposit : $");

Balance1 += double.Parse(Console.ReadLine());

Console.WriteLine($"Deposit done, your new balance ${Balance1}");

Console.WriteLine("Press any key to continue");

Console.ReadKey();

Console.Clear();

}

catch (FormatException)

{

InvalidInt();

}

catch(Exception e)

{

Console.WriteLine(e);

}

}

public void WithDraw()

{ // function fro withdraw

try

{

Console.Write("\nInput a value what do you want to WithDraw");

double value = double.Parse(Console.ReadLine());

//condition

if (LimitWithDraw >= value && Balance1-value >= 0 && value > 0 )

{

Balance1 -= value;

Console.WriteLine($"\nWithDraw Complete, your new balance ${Balance1}");

Console.WriteLine("Press any key to continue");

Console.ReadKey();

Console.Clear();

}

else if(Balance1 - value >= 0)

{

Console.WriteLine("\nYou are unable to make this action, your balance it will be 0");

Console.WriteLine("Press any key to continue");

Console.ReadKey();

Console.Clear();

}

else if(value > LimitWithDraw1)

{

Console.WriteLine($"\nSorry, you are unable to make this action the amount you want with draw is more bigger than your Limit Of WithDraw -> ${LimitWithDraw1}");

Console.WriteLine("Press any key to continue");

Console.ReadKey();

Console.Clear();

}

else

{

Console.WriteLine($"\nSorry, you cannot complete this withdraw, please check all informations\nAttencion your Limit for WithDraw in {LimitWithDraw1}");

Console.WriteLine("Press any key to continue");

Console.ReadKey();

Console.Clear();

}

}

catch (FormatException)

{

InvalidInt();

}

catch(Exception e)

{

Console.WriteLine(e.Message);

}

}

public void ShowData()

{// show datas

Console.WriteLine("\tDatas fo your account");

Console.WriteLine($"\nNumber: {NumberAccount1}");

Console.WriteLine($"Balance: {Balance1}");

Console.WriteLine($"Limit of With Draw: {LimitWithDraw1}");

Console.WriteLine("\n\nPress any key to continue");

Console.ReadKey();

Console.Clear();

}

public void ChangeLimit()

{ // change limits

try

{

Console.Write("\nWhat is your new balance? $");

double NewBalance = double.Parse(Console.ReadLine());

Console.WriteLine($"\n your new limit is Limit of With Draw: {LimitWithDraw1}");

Console.WriteLine("Press any key to continue");

Console.ReadKey();

Console.Clear();

}

catch (FormatException)

{

InvalidInt();

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

}

}