Azamat Salamatov

nORTH AMERICAN UNIVERSITY  11/12/2022

project: banking Application

Contents

[Flowchart of the Program 1](file:///C:\Users\User\Desktop\flowchart.docx#_Toc119150701)

[Code: 2](#_Toc119150702)

[main.py 2](#_Toc119150703)

[login.py 4](#_Toc119150704)

[balance.py 4](#_Toc119150705)

[withdraw.py 5](#_Toc119150706)

[deposit.py 5](#_Toc119150707)

[send\_and\_receive\_money.py 6](#_Toc119150708)

# Flowchart of the Program

Azamat Salamatov

Project: Banking Application

Date: 11/12/2022

# Code:

Made in Visual Studio Code:

## main.py

# https://replit.com/@UnnecessaryAcco/PROJECT-AZAMAT-COMP1411-BANKING-APPLICATION#main.py

"""

Banking Application

- login

- withdraw

- deposit

- send money to another user

- receive money from another user

"""

#I have created other python files, now I am importing them

#Moreover, I import sleep function to make my function stop for several moments

from time import sleep

from login import login

from balance import balance

from withdraw import withdraw

from deposit import deposit

from send\_and\_receive\_money import send\_money

from send\_and\_receive\_money import receive\_money

#start main function, all other functions are inside this main

def main():

    #Set initial money

    total\_money = float(1000)

    #print intro sentence

    print("\n\*\*\*\*\*Banking Application\*\*\*\*\*\n#login: user\n#password: 1234")

    #sleep(i.e. stop, wait) for 1 sec

    sleep(1)

    #create a boolean variable is\_login\_true, it will run the login function (see: login.py file)

    is\_login\_true = login()

    #if the is\_login\_true variable is True, then the application will start,

    #otherwise, the function will break(i.e. end, stop)

    if is\_login\_true:

        sleep(1)

        print("\n\*\*\*\* Welcome to Banking Application! \*\*\*\*")

        #while loop will run(mainly, it will show the main page of the program),

        #until the user selects to exit the program

        while True:

            sleep(1)

            print("\n\nSelect Options ")

            print("1. Check Balance ")

            print("2. Cash Withdraw ")

            print("3. Make Deposit ")

            print("4. Send Money to Another User ")

            print("5. Receive Money from Another User ")

            print("6. Exit\n")

            #after showing the interface, now the user chooses one of the functions 1 to 6

            choice = input()

            #according to the choice of the user, the program will run different functions (see, other files for each function)

            if choice == "1":

                #program just print the return funtion of balance() function

                print(balance(total\_money))

            elif choice == "2":

                #here and in the followings, I set total\_money variable,

                #since all of these functions(2-5) will return total\_money amount

                total\_money = withdraw(total\_money)

            elif choice == "3":

                total\_money = deposit(total\_money)

            elif choice == "4":

                total\_money=send\_money(total\_money)

            elif choice == "5":

                total\_money=receive\_money(total\_money)

            elif choice == "6":

                #if the choice is "6", the program will end, break

                break

            else:

                #if the choice of function is not 1-6, the program will say, that the input is incorrect

                print("You entered invalid input! ")

    else:

        #if the username or the password is wrong, the program will say that

        print("\nYour username or password is wrong")

#now, call the main function, to start the program

main()

## login.py

#this function will check, whether the username and/or the password are correct

def login():

    #dictionary with key: username, and with value: password(string, since it is input)

    dict\_login = {

        "user": "1234"

    }

    #ask to input the username and the password, and strip them, if there are some extra spaces

    #the username and password are case-sensitive

    username = input("\nEnter your username: ").strip()

    password = input("\nEnter your password: ").strip()

    #check if the username exists in the keys of the dictionary

    if username in dict\_login.keys():

        #if the username is correct, then it will check if the password (value in dict) is correct (True or False)

        return dict\_login[username] == password

    else:

        #if the username is wrong, it will return False

        return False

## balance.py

# this function will return sting with the amount of money on the balance,

# it needs input, here total\_money

def balance(total\_money):

    return "Your balance is: ${}.".format(total\_money)

## withdraw.py

# this function will substract the amount of money, when withdrawn

# it needs input, here total\_money, and returns updated total\_money

def withdraw(total\_money):

    # ask the user the amount to withdraw

    amount = input("Enter amount to withdraw: $")

    # if the amount is number, "try:" will run

    try:

        # if the amount is available, if will run

        # convert amount(str) into float

        if float(amount) <= total\_money and float(amount) != 0:

            amount = float(amount)

            #now, the amount will substracted

            total\_money -= amount

            print("Get your money!\nThe left Amount is ${}.".format(total\_money))

        # if the amount is not available, elif will run

        elif float(amount) > total\_money or int(amount) == 0:

            print("Insufficient amount on account ")

            return total\_money

    # if the amount is not a number, "except:" will run

    except:

        print("Incorrect input! ")

        return total\_money

    return total\_money

## deposit.py

# this function will add money to the total\_amount

# it needs input total\_money, and return updated total\_money

def deposit(total\_money):

    print("Enter amount to deposit: ")

    # ask the amount (str for now)

    deposit\_amount = input("$")

    # if the amount is number, "try:" will run

    try:

        # check if the number is positive,

        # then the deposit will be added to the total amount

        # convert from str to float

        if float(deposit\_amount) > 0:

            deposit\_amount = float(deposit\_amount)

            total\_money += float(deposit\_amount)

            print("Operation is successful. Your new balance is: ${}".format(total\_money))

        # if the deposit is negative, the program will say this

        elif float(deposit\_amount) < 0:

            print("You cannot deposit negative number of money! ")

        # if the deposit is 0, the program will say this

        elif float(deposit\_amount) == 0:

            print("You cannot deposit $0")

    # if the amount is not a number, "except:" will run

    except:

        print("Incorrect input! ")

    return total\_money

## send\_and\_receive\_money.py

# these two functions will send money and get money from another user

# they need input and return updated total\_money

# create two list, one for number, second for the amount sent to another user

contacts\_of\_recipients = []

sent\_amount=[]

def send\_money(total\_money):

    # ask for amount to send

    amount = input("Enter amount to send: ")

    # if the amount is number, "try:" will run

    try:

        # check, if the amount to send is available

        # convert the string amount to float

        if float(amount) <= total\_money:

            amount = float(amount)

            # sent amount will be substracted

            total\_money -= amount

            # ask the phone number of the recipient, then save in the list

            number\_of\_recipient = input("Enter the phone number of the recipient: ")

            contacts\_of\_recipients.append(number\_of\_recipient)

            print("The money is sent successfully to number {}!\nThe new Balance is {}.".format(number\_of\_recipient, total\_money))

            # here I added the sent amount to the list,

            # then I will get the sum of this list,

            # that will be the total sent amount of money

            sent\_amount.append(amount)

        # if the amount is not available, the program will say this

        elif float(amount) > total\_money:

            print("Insufficient amount on account ")

            return total\_money

    # if the number is not a number, the program will say this

    except:

        print("Incorrect input! ")

        return total\_money

    return total\_money

# RECIEVE MONEY function

# I change the name of the list, to make it more clear

# then, i get the sum of the list of the sent amount

contacts\_of\_friends = contacts\_of\_recipients

float\_sent\_amount = sum(sent\_amount)

def receive\_money(total\_money):

    float\_sent\_amount = sum(sent\_amount)

    # ask the amount

    amount = input("Enter amount to receive: ")

    # ask the number of the friend to get his money

    number\_of\_friend = input("Enter the phone number of the friend to get his money: ")

    # 1. In order to get the money from another person, the user needs to send money to him first

    # 2. The user cannot get more money than he sent to another person

    # 3. Identity of another person is known only by phone number

    # if the input amount is number, try will run

    try:

        # first, we check, whether the user sent money to his friend

        if number\_of\_friend in contacts\_of\_friends:

            # then, we check if the amount to recieve from his friend is less than sent amount to his friend

            if float(amount) <= float\_sent\_amount:

                # then, the total\_money will be received

                total\_money += float(amount)

                # sent amount will be decreased, so that the user cannot get more money than he has sent, even in several steps

                float\_sent\_amount -= float(amount)

                print("The money is sent successfully to number {}!\nThe new Balance is {}.".format(number\_of\_friend, total\_money))

                return total\_money

            # if the amount is not available, the program will say it

            elif float(amount) > float\_sent\_amount:

                print("You cannot get more money, than you have sent your friend! ")

                return total\_money

        # if the user has not sent money, to another person, he cannot recieve his money

        elif number\_of\_friend not in contacts\_of\_friends:

            print("To get money of your friend, you need to send him money first! ")

            return total\_money

    # if the input number is not a number, except will run

    except:

        print("Incorrect input! ")

        return total\_money