Centro Universitario FEI

QuemPoupaTem

CAMILA LAURINDO REIS - RA: 22222037-8
MATHEUS MATOS DE OLIVEIRA – RA:

Banco QuemPoupaTem

Introdução

O objetivo do projeto “QuemPoupaTem” é estabelecer um ambiente bancário interativo e eficiente, no qual um “funcionário” desempenha um papel central na execução das operações financeiras. O sistema contempla diversas funcionalidades essenciais, como a capacidade de listar clientes, realizar transferências entre contas, fornecer extratos detalhados das transações, entre outros recursos relevantes. Para garantir um acesso simplificado e intuitivo, foi desenvolvido um menu inicial que direciona os usuários para as diferentes funcionalidades disponíveis. Além disso, todas as informações pertinentes a esse ambiente bancário são armazenadas de forma segura em um arquivo dedicado.

1 Funcionalidade

Cabeçalho

O código primeiro importa as bibliotecas necessárias, incluindo **re, random e json**, para realizar as operações necessárias. Em seguida, utiliza a biblioteca **datetime** para obter a data e hora atuais. Os dados dos clientes são carregados de um arquivo de texto no formato **JSON**, permitindo acessar as informações armazenadas. A lista de clientes é então impressa na tela para visualização.

Além disso, uma função é definida no código para atualizar os dados dos clientes no arquivo. Essa função garante que as alterações realizadas no sistema sejam refletidas corretamente no arquivo, mantendo os dados atualizados.

Após carregar os dados dos clientes e exibir a lista, a data e hora atuais são formatadas no formato desejado utilizando o método strftime(). Em seguida, a data formatada é impressa, permitindo visualizar a data e hora da consulta.

Essas etapas do código possibilitam listar os clientes registrados no sistema, apresentando a data e hora em que a consulta foi realizada.

```
from re import search
from random import *
import json
from datetime import datetime

clientes = []
with open("bancodedados.txt", "r") as arquivo:
    clientes = json.load(arquivo)
print(clientes)

def reloadclientes():
    with open("bancodedados.txt", "w") as arquivo:
        json.dump(clientes, arquivo)

datadehoje = datetime.now()
databr = (" %s / %s / %s  %s:%.2s" % (datadehoje.day,  datadehoje.month,
                                      datadehoje.year, datadehoje.hour, datadehoje.minute))
print(databr)
```

Variavéis

Para realizar as operações desejadas no sistema, são utilizadas várias variáveis com informações relevantes, indentificadas em cada comentário:

```
### declarando variaveis ###
operacao = 0 ## informa a operacao desejada
nome = 0 #nome do cliente
cpf = 0 #cpf do cliente
conta = 0 # tipo de conta do cliente
valor_inicial= 0 #qual o valor inicial do cliente
senha= 0 #senha do cliente
valor_deb = 0 #valor para debitar
valor_depo= 0#valor para depositar
cpf2= 0 #cpf do destinatario
valor_t=0 #valor da transferencia
apagar_cliente = 0 # usado para função apagar
valortotal = 0 # valor total em conta
tarifa = 0 # tarifa, quando não houver é 0
valormenosvalor = 0
```

Essas variáveis e listas são utilizadas para controlar e armazenar as informações relevantes durante a execução das operações do sistema bancário.

Menu

A função opcoes() exibe o menu de operações disponíveis no banco. Ao ser chamada, a função imprime uma mensagem de boas-vindas e exibe as operações possíveis no sistema:

```
### abaixo o banco pede as opções necessarias do menu ###
def opcoes():
    print("Bem Vindo ao Menu do Banco QuemPoupaTem")
    print("A seguir as operações possiveis em nosso APP")
    print("1. Novo cliente")
    print("2. Apaga cliente")
    print("3. Listar clientes")
    print("4. Débito")
    print("5. Depósito")
    print("6. Extrato")
    print("7. Transferência entre contas")
    print("8. Operação livre")
    print("9. Sair")
    global operacao
    operacao = int(input("Digite a opção desejada: "))
```

Em seguida, solicita ao usuário que digite a opção desejada, e essa entrada é armazenada na variável operacao após ser convertida para um número inteiro.

Essa função permite ao usuário escolher qual operação deseja.

Novo cliente

A função novo_cliente() é responsável por cadastrar novos clientes no sistema. Ela solicita ao usuário que forneça o nome, CPF, tipo de conta, valor inicial e senha do cliente. Os dados informados são armazenados em uma lista e adicionados à lista de clientes existente. A função também exibe os dados cadastrados do cliente e redefine o valor total como zero.

A função reloadclientes() é responsável por recarregar os dados dos clientes no arquivo. Quando chamada, essa função reescreve totalmente o arquivo "bancodedados.txt" com os dados atualizados da lista de clientes.

```
### funcao para cadastrar novos clientes ###
def novo_cliente():
    ## comeca a pedir informacoes ##
    nome = input("Digite seu nome: ")
    cpf = (input("Digite seu CPF: "))
    conta = input("Sua conta vai ser comum ou plus?: ")
    valor_inicial= float(input("Valor inicial: "))
    senha= (input("Digite a sua senha: "))
    ## termina de pedir informacoes ##
    valortotal = valor_inicial
    cliente = [nome, cpf,conta,valor_inicial, senha, valortotal] #a lita esta gaurdando as
    #inofrmacoes coorrelacionando uma string com uma variavel
    clientes.append(cliente) #esta colocando o novo valor na lista
    reloadclientes()
    print(nome,cpf,conta,valortotal,senha)
```

Apaga cliente

A função apaga_cliente() é responsável por realizar a exclusão de um cliente do sistema.

A função percorre a lista de clientes utilizando um loop for com a variável x para acessar os índices e a variável a para acessar cada cliente individualmente.

Dentro do loop, é verificado se o CPF informado pelo usuário é igual ao CPF de um dos clientes na lista. Caso haja uma correspondência, o código exibe a lista de clientes, remove o cliente encontrado utilizando o método pop(x) e exibe a lista de clientes atualizada. Em seguida, a função é encerrada com o uso do comando return.

A função reloadclientes() é responsável por recarregar os dados dos clientes no arquivo. Quando chamada, essa função reescreve totalmente o arquivo "bancodedados.txt" com os dados atualizados da lista de clientes.

```
## funcao para apagar um cliente ##
def apaga_cliente():
    ## comeca a pedir informacoes ##
    print("Você escolheu a opção apagar cliente")
    cpf = input("Digite o CPF do cliente que deseja apagar: ") # pede o CPF para deletar do sistema
    for x in range(len(clientes)): # anda pelos clientes até com a range até o final da lista
        ## termina de pedir informacoes ##
        for a in clientes:
            if cpf == a[1]: #localiza o cpf e exlui do sistema
                print(clientes)
                clientes.pop(x)
                print(clientes)
                reloadclientes()
```

Listar clientes

A função listar_clientes() é responsável por exibir na tela todas as informações dos clientes cadastrados no sistema. Ao ser chamada, a função exibe a mensagem "Você escolheu a opção listar clientes" e, em seguida, percorre a lista de clientes usando um loop for. Para cada cliente, são exibidos os seguintes dados: nome, CPF, tipo de conta, valor inicial e senha.

```
### funcao que lista todos os clientes ###
def listar_clientes():
    print("Você escolheu a opção listar clientes")
    for x in clientes: #confere os clientes dentro da lista
            print("Nome:", x[0]) #as quatro linhas abaixo assim como essa imprime as inf
            print("CPF:", x[1])
            print("Conta:", x[2])
            print("Saldo Atual:", x[5])
            print("Senha:", x[4])
```

Debito

A função debito() é responsável por realizar uma operação de débito na conta de um cliente. O código percorre a lista de clientes e verifica se o CPF e a senha informados correspondem a um cliente com a conta do tipo "plus" ou "comum". Dependendo do tipo de conta, é aplicada uma taxa específica ao valor a ser debitado. O código também verifica se o saldo disponível é suficiente para realizar o débito. Se todas as condições forem atendidas, o débito é efetuado e o saldo atualizado. Caso contrário, são exibidas mensagens adequadas.

A função reloadclientes() é responsável por recarregar os dados dos clientes no arquivo. Quando chamada, essa função reescreve totalmente o arquivo "bancodedados.txt" com os dados atualizados da lista de clientes.

```
def debito():
    ## comeca a pedir informacoes ##
    print("Você escolheu a opção débito")
    cpf = input("Digite seu CPF: ") # dados do cleinte
    senha= input("Digite a sua senha: ")
    valordebitado = float(input("Qual valor deseja debitar?: "))
    ## termina de pedir informacoes ##
    for x in clientes: ## esse for entra na lista de todos clientes
        for j in x: ## esse for entra em casa cliente verificando os itens, quando encontra o que quere
            if (j == senha and cpf == x[1]) and x[2] == "plus": ## verificando as credenciais do client
                print("Encontramos o seu cadastro, cliente PLUS tem direito especial, somente 3 por cen
                tarifa = valordebitado*0.03 # calculando tarifa
                valorconfere = x[5] - (valordebitado+tarifa) # calculando quanto vai ficar o saldo para
                if valorconfere > -5000:
                    x[5] = valorconfere #alterando o valor para o que iria ficar descontando a tarifa e
                    print("Transação feita, seu saldo é de R$%f" % (x[5]))
                    reloadclientes()
                else:
                    print("saldo insuficiente deposite para continuar")
                ## abaixo é basicamente a mesma logica ##
            if (j == senha and cpf == x[1]) and x[2] == "comum":
                print("Encontramos o seu cadastro, cliente comum tem possui 5 por cento de taxa, vire p
                tarifa = valordebitado*0.05
                valordebitado = tarifa+valordebitado
                valorconfere = x[5] - (valordebitado+tarifa)
                if valorconfere > -1000:
                    x[5] = valorconfere
                    print("Transação feita, seu saldo é de R$%.2f" % (x[5]))
                    reloadclientes()
                else:
                    print("saldo insuficiente deposite para continuar")
```

Deposito

A função valor_cliente() é responsável por realizar uma operação de depósito na conta de um cliente. O código percorre a lista de clientes e verifica se o CPF informado corresponde a um cliente cadastrado. Se o CPF for válido, é solicitado o valor a ser depositado e exibida a mensagem de sucesso.

A função reloadclientes() é responsável por recarregar os dados dos clientes no arquivo. Quando chamada, essa função reescreve totalmente o arquivo "bancodedados.txt" com os dados atualizados da lista de clientes.

```python
#### funcao deposito bancario ####
def valor_cliente():
    print("Você escolheu deposito bancario")
    cpf = input("Digite seu CPF: ")
    for x in clientes:
        if x[1] == cpf:#confere se o cpf existe/esta correto
            valor_depo= float(input("Digite o valor a ser depositado: ")) ## valor a ser depositado
            annterior = x[5] ## declarando para informar na frase o saldo antigo
            x[5] += (valor_depo) # soma o valor depositado com saldo anterior
            reloadclientes() # recarrega a lista clientes
            print("O valor de R$", valor_depo, " foi depositado com sucesso. Saldo anterior: ", annterior,"saldo atual: ", x[5])
            return
    print("CPF inexistente") # caso nenhum cliente seja achado com cpf informado, for encerra com a informação
```

Transferencia

A função transferencia() é responsável por realizar uma transferência entre contas de dois clientes. O código verifica se o CPF e a senha correspondem a um cliente remetente válido e se o valor da transferência é inferior ao saldo disponível. Em seguida, é solicitado o CPF do destinatário da transferência e verificado se o CPF é válido. Caso todas as condições sejam atendidas, o valor é transferido entre as contas dos clientes.

```python
### funcao transferencia ###
def transferencia():
    print("Você escolheu transferência") ## informa a pagina que entrou
    cpf = input("Digite seu CPF: ") #CPF DA PESSOA QUE IRÁ TRANSFERIR
    senha = input("Digite a sua senha: ") # SENHA DA PESSOA QUE IRÁ TRANSFERIR
    destinatario = input("Digite o CPF/CNPJ do destinatario: ") # CPF DA CONTA DE DESTINO
    valor_t = float(input("Digite o valor da transferência: ")) # VALOR DA TRANSFERENCIA
    for x in clientes: # FOR PARA INTERAR SOBRE OS CLIENTES
        if (x[4] == senha and cpf == x[1]): # CONFERINDO CADA SENHA E CPF, DANDO MATCH, PROSSEGUE
            novosaldo = x[5] - valor_t # JA DEFINIMOS O PROVAVEL NOVO SALDO
            if (x[2] == "comum")  and (novosaldo < -1000): #CONFERENCIA SE ESTÁ NO LIMITE DISPONIVEL
                print("Saldo insuficiente")
            elif (x[2] == "plus") and (novosaldo < -5000): #CONFERENCIA SE ESTÁ NO LIMITE DISPONIVEL
                print("Saldo insuficiente")
            elif (x[2] == "comum")  and (novosaldo > -1000): # VERIFICA QUE O LIMITE ESTÁ DISPONIVEL CONTA COMUM
                print("Credênciais corretas")
                for destino in clientes: # MAIS UMA INTERAÇÃO SOBRE OS CLIENTES PARA ACHAR O DESTINATARIO
                    if (destino[1] == destinatario): # CONFERINDO AS CREDENCIAIS
                        x[5] -= valor_t # RETIRANDO O SALDO DA CONTA DO REMETENTE
                        destino[5] += valor_t # DEPOSITANDO NA CONTA DO DESTINATARIO
                        reloadclientes() # SALVANDO
                    else:
                        print("Operação falha, tente novamente mais tarde") # ALGO DEU ERRADO, CPF, SENHA, VALOR
            elif (x[2] == "plus") and (novosaldo > -5000): # ## A MESMA SEQUÊNCIA DO CÓDIGO A CIMA, SÓ MUDA O TIPO DE CONTA "PLUS"
                print("Credênciais corretas")
                for destino in clientes:
                    if (destino[1] == destinatario):
                        novosaldo = destino[5] + valor_t
                        x[5] -= valor_t
                        destino[5] += valor_t
                        reloadclientes()
                    else:
                        print("Operação falha, tente novamente mais tarde")
            else:
                    print("Erro cadastral, entre em contato para atualização de cadastro")
```

Extrato

A função extrato() permite que os clientes acessem o extrato de suas contas. O usuário é solicitado a digitar seu CPF e senha. O código percorre uma lista chamada clientes em busca de um cliente correspondente ao CPF e senha fornecidos. Se um cliente válido for encontrado, suas informações, como nome, CPF e número da conta, são exibidas.

Em seguida, o código verifica a lista extratogeral em busca de transações associadas ao CPF do cliente encontrado. Cada transação contém informações como data, movimentação, tarifa e saldo. As transações são exibidas na tela, com o movimento financeiro formatado adequadamente (incluindo o sinal "+" ou "-") e o saldo do cliente exibido apenas uma vez, com base no valor armazenado na variável valorconfere.

Se nenhum cliente válido for encontrado ou se a senha estiver incorreta, é exibida a mensagem "Cliente não encontrado ou senha incorreta".

```
223
224 def extrato():
225     print("Você escolheu extrato")
226     cpf = input("Digite seu CPF: ")
227     senha = input("Digite a sua senha: ")
228     cliente_encontrado = False
229     for x in clientes:
230         if x[1] == cpf and x[4] == senha:
231             cliente_encontrado = True
232             valorconfere = x[5]
233             print("Cliente:", x[0])
234             print("CPF:", x[1])
235             print("Conta:", x[2])
236             break
237     if cliente_encontrado:
238         for j in extratogeral:
239             if cpf == j[1]:
240                 movimentacao = float(j[3].replace("+", "").replace("-", "").replace("R$", "")
241                 sinal = "+" if j[3].startswith("+") else "-"
242                 print("Data: ", j[0], end=" ")
```

```
PROBLEMS  OUTPUT  TERMINAL
TERMINAL
Digite seu CPF: 147258369
Digite a sua senha: 321
Cliente: joao
CPF: 147258369
Conta: plus
Data:   25 / 5 / 2023  14:29 Movimentação:  -2000000.0 Tarifa:  0  Saldo:  2000600.0

Data:   25 / 5 / 2023  15:59 Movimentação:  +600.0 Tarifa:  0  Saldo:  2000600.0
```

Doação

A função doacao() permite que os clientes façam doações para instituições. Ela recebe o CPF do cliente e a instituição desejada como entrada. Se o CPF for válido e a instituição existir, o cliente pode fazer a doação digitando o valor. O código verifica se o saldo do cliente é suficiente para a doação e exibe uma mensagem correspondente. Caso contrário, são exibidas mensagens de erro.

```python
#### funcao de doacao ####
instituicoes = ["AACD", "Teleton", "Criança esperança"]
def doacao(clientes):
    cpf = input("Digite seu CPF: ")
    instituicao = input("Escolha a instituição (AACD, Teleton, Criança esperança): ") #escolhe a instuicao
    for cliente in clientes:
        if cliente[1] == cpf:
            if instituicao in instituicoes:
                valor_doacao = float(input("Digite o valor: "))
                #confere o tipo de conta se esta no saldo e se esta no negativo ou nao e se
                if ((cliente[2] == "comum") and (cliente[5] - valor_doacao < -1000)) or ((cliente[2] == "plus") and (cliente[5] - valor_doacao < -5000)):
                    print("Saldo insuficiente.")
                else:
                    print("Doação para", instituicao, "feita com sucesso")
            else:
                print("Instituição não encontrada")
            return
    print("CPF inexistente")
```

While True

Por fim, o código utiliza um loop while True para criar um menu principal, onde o usuário pode escolher as operações desejadas. Dependendo da opção escolhida, a função correspondente é chamada. O loop continua em execução até que o usuário selecione a opção 9, que encerra o programa.

```python
#### laço de repetição do menu principal ####
while True:
    opcoes()
    print(operacao)
    if operacao == 1: ## if elif e else verifica a opção no menu
        novo_cliente()
    elif operacao == 2:
        apaga_cliente()
    elif operacao == 3:
        listar_clientes()
    elif operacao == 4:
        debito()
    elif operacao == 5:
        valor_cliente()
    elif operacao == 6:
        extrato()
    elif operacao == 7:
        transferencia()
    elif operacao == 8:
        doacao()
    elif operacao == 9: ## paralisa o app
        print("Que pena que escolheu sair, esperamos você em breve")
        break
    else:
        print("Operação invalida")
```

Menu

```
A seguir as operações possiveis em nosso APP
1. Novo cliente
2. Apaga cliente
3. Listar clientes
4. Débito
5. Depósito
6. Extrato
7. Transferência entre contas
8. Operação livre
9. Sair
Digite a opção desejada:
```

Novo cliente

```python
### funcao para cadastrar novos clientes ###
def novo_cliente():
    ## comeca a pedir informacoes ##
    nome = input("Digite seu nome: ")
    cpf = (input("Digite seu CPF: "))
    conta = input("Sua conta vai ser comum ou plus?: ")
    valor_inicial= float(input("Valor inicial: "))
    senha= (input("Digite a sua senha: "))
    ## termina de pedir informacoes ##
    valortotal = valor_inicial
    cliente = [nome, cpf,conta,valor_inicial, senha, valortotal] #a
    #inofrmacoes coorrelacionando uma string com uma variavel
    clientes.append(cliente) #esta colocando o novo valor na lista
    reloadclientes()
    print(nome,cpf,conta,valortotal,senha)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

Digite seu nome: Reis
Digite seu CPF: 52766770852
Sua conta vai ser comum ou plus?: plus
Valor inicial: 100000
Digite a sua senha: 321
Reis 52766770852 plus 100000.0 321
```

Apaga cliente

```python
## funcao para apagar um cliente ##
def apaga_cliente():
    ## comeca a pedir informacoes ##
    print("Você escolheu a opção apagar cliente")
    cpf = input("Digite o CPF do cliente que deseja apagar: ") # pede o CPF para deletar do sistema
    for x in range(len(clientes)): # anda pelos clientes até com a range até o final da lista
        ## termina de pedir informacoes ##
        for a in clientes:
            if cpf == a[1]: #localiza o cpf e exlui do sistema
                print(clientes)
                clientes.pop(x)
                print(clientes)
                reloadclientes()
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

Você escolheu a opção apagar cliente
Digite o CPF do cliente que deseja apagar: 52766770852
[['Matheus', '123', 'plus', 100000.0, '1234', 100000.0], ['Telma', '440', 'comum', 300000.0, '1337', 300000.0], ['Reis
0000.0, '321', 100000.0]]
[['Telma', '440', 'comum', 300000.0, '1337', 300000.0], ['Reis', '52766770852', 'plus', 100000.0, '321', 100000.0]]
[['Telma', '440', 'comum', 300000.0, '1337', 300000.0], ['Reis', '52766770852', 'plus', 100000.0, '321', 100000.0]]
[['Telma', '440', 'comum', 300000.0, '1337', 300000.0]]
```

Listar clientes

```python
### funcao que lista todos os clientes ###
def listar_clientes():
    print("Você escolheu a opção listar clientes")
    for x in clientes: #confere os clientes dentro da lista
            print("Nome:", x[0]) #as quatro linhas abaixo ass
            print("CPF:", x[1])
            print("Conta:", x[2])
            print("Saldo Atual:", x[5])
            print("Senha:", x[4])
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
3
Você escolheu a opção listar clientes
Nome: Telma
CPF: 440
Conta: comum
Saldo Atual: 300000.0
Senha: 1337
```

Debito

```python
def debito():
    ## comeca a pedir informacoes ##
    print("Você escolheu a opção débito")
    cpf = input("Digite seu CPF: ") # dados do cleinte
    senha= input("Digite a sua senha: ")
    valordebitado = float(input("Qual valor deseja debitar?: "))
    ## termina de pedir informacoes ##
    for x in clientes: ## esse for entra na lista de todos clientes
        for j in x: ## esse for entra em casa cliente verificando os itens, quando
            if (j == senha and cpf == x[1]) and x[2] == "plus": ## verificando as
                print("Encontramos o seu cadastro, cliente PLUS tem direito especi
                tarifa = valordebitado*0.03 # calculando tarifa
                valorconfere = x[5] - (valordebitado+tarifa) # calculando quanto v
                if valorconfere > -5000:
                    x[5] = valorconfere #alterando o valor para o que iria ficar d
                    print("Transação feita, seu saldo é de R$%f" % (x[5]))
                    reloadclientes()
                else:
                    print("saldo insuficiente deposite para continuar")
                ## abaixo é basicamente a mesma logica ##
            if (j == senha and cpf == x[1]) and x[2] == "comum":
                print("Encontramos o seu cadastro, cliente comum tem possui 5 por
```

ROBLEMS OUTPUT TERMINAL

TERMINAL

```
Você escolheu a opção débito
Digite seu CPF: 52766770852
Digite a sua senha: 321
Qual valor deseja debitar?: 500
Encontramos o seu cadastro, cliente PLUS tem direito especial, somente 3 por cento de taxa
Transação feita, seu saldo é de R$1485.000000
```

Deposito

```python
def valor_cliente():
    print("Você escolheu deposito bancario")
    cpf = input("Digite seu CPF: ")
    for x in clientes:
        if x[1] == cpf:#confere se o cpf existe/esta correto
            valor_depo= float(input("Digite o valor a ser depositado: ")) ## valor a ser depos
            annterior = x[5] ## declarando para informar na frase o saldo antigo
            x[5] += (valor_depo) # soma o valor depositado com saldo anterior
            reloadclientes() # recarrega a lista clientes
            print("O valor de R$", valor_depo, " foi depositado com sucesso. Saldo anterior: "
            return
    print("CPF inexistente") # caso nenhum cliente seja achado com cpf informado, for encerra
```

```
ROBLEMS  OUTPUT  TERMINAL
TERMINAL
Digite a opção desejada: 5
5
Você escolheu deposito bancario
Digite seu CPF: 147258369
Digite o valor a ser depositado: 60
O valor de R$ 60.0  foi depositado com sucesso. Saldo anterior:  555555.0 saldo atual:  555615.0
```

Transferencia

```python
def transferencia():
    print("Você escolheu transferência") ## informa a pagina que entrou
    cpf = input("Digite seu CPF: ") #CPF DA PESSOA QUE IRÁ TRANSFERIR
    senha = input("Digite a sua senha: ") # SENHA DA PESSOA QUE IRÁ TRANSFERIR
    destinatario = input("Digite o CPF/CNPJ do destinatario: ") # CPF DA CONTA D
    valor_t = float(input("Digite o valor da transferência: ")) # VALOR DA TRANS
    for x in clientes: # FOR PARA INTERAR SOBRE OS CLIENTES
        if (x[4] == senha and cpf == x[1]): # CONFERINDO CADA SENHA E CPF, DANDO
            novosaldo = x[5] - valor_t # JA DEFINIMOS O PROVAVEL NOVO SALDO
            if (x[2] == "comum")  and (novosaldo < -1000): #CONFERENCIA SE ESTÁ
                print("Saldo insuficiente")
            elif (x[2] == "plus") and (novosaldo < -5000): #CONFERENCIA SE ESTÁ
                print("Saldo insuficiente")
            elif (x[2] == "comum")  and (novosaldo > -1000): # VERIFICA QUE O LI
                print("Credênciais corretas")
                for destino in clientes: # MAIS UMA INTERAÇÃO SOBRE OS CLIENTES
```

```
ROBLEMS  OUTPUT  TERMINAL
TERMINAL
7
Você escolheu transferência
Digite seu CPF: 52766770852
Digite a sua senha: 321
Digite o CPF/CNPJ do destinatario: 440
Digite o valor da transferência: 200
Credênciais corretas
Transferencia feita com sucesso
```

Extrato

```
223
224  def extrato():
225      print("Você escolheu extrato")
226      cpf = input("Digite seu CPF: ")
227      senha = input("Digite a sua senha: ")
228      cliente_encontrado = False
229      for x in clientes:
230          if x[1] == cpf and x[4] == senha:
231              cliente_encontrado = True
232              valorconfere = x[5]
233              print("Cliente:", x[0])
234              print("CPF:", x[1])
235              print("Conta:", x[2])
236              break
237      if cliente_encontrado:
238          for j in extratogeral:
239              if cpf == j[1]:
240                  movimentacao = float(j[3].replace("+", "").replace("-", "").replace("R$", "")
241                  sinal = "+" if j[3].startswith("+") else "-"
242                  print("Data: ", j[0], end=" ")
```

```
PROBLEMS  OUTPUT  TERMINAL
TERMINAL
Digite seu CPF: 147258369
Digite a sua senha: 321
Cliente: joao
CPF: 147258369
Conta: plus
Data:   25 / 5 / 2023  14:29 Movimentação:  -2000000.0 Tarifa:  0  Saldo:  2000600.0

Data:   25 / 5 / 2023  15:59 Movimentação:  +600.0 Tarifa:  0  Saldo:  2000600.0
```

Doação

```
194  #### funcao de doacao ####
195  instituicoes = ["AACD", "Teleton", "Criança esperança"]
196  def doacao():
197      cpf = input("Digite seu CPF: ")
198      instituicao = input("Escolha a instituição (AACD, Teleton, Criança esperança): ") #escolh
199      for cliente in clientes:
200          if cliente[1] == cpf:
201              if instituicao in instituicoes:
202                  valor_doacao = float(input("Digite o valor: "))
203                  #confere o tipo de conta se esta no saldo e se esta no negativo ou nao e se
204                  if ((cliente[2] == "comum") and (cliente[5] - valor_doacao < -1000)) or ((cli
205                      print("Saldo insuficiente.")
206                  else:
207                      print("Doação para", instituicao, "feita com sucesso")
208              else:
209                  print("Instituição não encontrada")
210              return
211      print("CPF inexistente")
212
```

```
PROBLEMS  OUTPUT  TERMINAL
TERMINAL
Digite seu CPF: 52766770852
Escolha a instituição (AACD, Teleton, Criança esperança): AACD
Digite o valor: 500
Doação para AACD feita com sucesso
```