

**Disciplina: POO**

**Professor: Genivan Silva**

**Alunos: Anderson Nascimento;**

**Francisca Natalia; Marcos Antonio**

# Atividade 02 – Herança e Polimorfismo

1. Crie uma classe chamada Conta Corrente, com os atributos agência, número da conta e saldo. Crie duas subclasses da classe Conta Corrente, uma chamada Conta Física e outra chamada Conta Jurídica. A Conta Física deve possuir, além dos atributos herdados, os atributos específicos dela que são CPF do cliente e renda mensal. Já a conta jurídica deve possuir os atributos CNPJ da empresa e capital investido.

class ContaCorrente:

    def \_\_init\_\_(*self*, *agencia*, *numConta*, *saldo*):

*self*.agencia = *agencia*

*self*.numConta = *numConta*

*self*.saldo = *saldo*

class ContaPJ(ContaCorrente):

    def \_\_init\_\_(*self*, *agencia*, *numConta*, *saldo*, *cnpj*, *capital*):

        super().\_\_init\_\_(*agencia*, *numConta*, *saldo*)

*self*.cnpj = *cnpj*

*self*.capital = *capital*

    def visualizar(*self*):

        print('===========================================================\n'

            f'Agencia:{*self*.agencia}\nNumero da Conta:{*self*.numConta}\n'

        f'Saldo:{*self*.saldo}\nCNPJ:{*self*.cnpj}\nCapital:{*self*.capital} \n'

        '===============================================================')

class ContaPF(ContaCorrente):

    def \_\_init\_\_(*self*, *agencia*, *numConta*, *saldo*, *cpf*, *renda*):

        super().\_\_init\_\_(*agencia*, *numConta*, *saldo*)

*self*.cpf = *cpf*

*self*.renda = *renda*

    def visualizar(*self*):

        print('===========================================================\n'

            f'Agencia:{*self*.agencia}\nNumero da Conta:{*self*.numConta}\n'

        f'Saldo:{*self*.saldo}\nCPF:{*self*.cpf}\nRenda:{*self*.renda}\n'

        '===============================================================')

pj = ContaPJ(1, '1111-1', 200000.00, '46.031.672/0001-65', 1000000.00)

pf = ContaPF(1, '45-1', 345.50, '684.276.460-20', 1100.00)

pj.visualizar()

pf.visualizar()

1. Na classe Conta Corrente implemente os métodos sacar e depositar. A operação depósito é a mesma para conta física e jurídica e ela aumenta o valor do saldo do cliente. Lembre como a operação de depósito é realizada no mundo real, além de informar o valor que está sendo depositado é necessário informar quem é o favorecido com o depósito, ou seja, é possível realizar um depósito em outra conta que não seja a que está realizando a operação. Já a operação de saque é diferente para as contas física e jurídica. Apesar da restrição óbvia da operação sacar (o valor do saque não pode ser maior que do saldo), na conta física, se a renda mensal do cliente for acima de 5000, ele pode sacar até 800 reais a mais quando o saldo fica em zero (como se fosse um empréstimo). Se a renda da pessoa física não for superior a 5000, o saque não deve ser autorizado quando não houver saldo suficiente.

1

Já para a operação de saque da conta jurídica, quando não há saldo suficiente na conta, o valor excedente pode ser descontado do capital investido, quando houver algum valor relativo a esse atributo.

**Exemplo**:

Conta Física – saldo da conta: 500 | Saque: 600 | **renda: 7000**

Status do saque: autorizado

Saldo da conta: -100

Conta Física – saldo: 100 | Saque: 300 | **renda: 3000**

Status do saque: não autorizado

Saldo da conta: não é modificado

Conta Jurídica – saldo: 500 | Saque: 600| **capital investido: 1000**

Status do saque: autorizado

Saldo da conta: 0

Saldo do capital investido: 900

class ContaCorrente:

def \_\_init\_\_(*self*, *agencia*, *numConta*, *saldo*):

*self*.agencia = *agencia*

*self*.numConta = *numConta*

*self*.saldo = *saldo*

def sacar(*self*):

pass

def depositar(*self*, *deposito*, *outraconta*=-1):

print('===========================================================\n')

if *outraconta* != -1:

*outraconta*.saldo += *deposito*

print(f'Valor depositado:{*deposito*}\n'

f'Saldo atual:{*outraconta*.saldo}\n')

else:

*self*.saldo += *deposito*

print(f'Valor depositado:{*deposito*}\n'

f'Saldo é atual:{*self*.saldo}\n')

class ContaPJ(ContaCorrente):

def \_\_init\_\_(*self*, *agencia*, *numConta*, *saldo*, *cnpj*, *capital*):

super().\_\_init\_\_(*agencia*, *numConta*, *saldo*)

*self*.cnpj = *cnpj*

*self*.capital = *capital*

def visualizar(*self*):

print('===============================================================\n'

f'Agencia:{*self*.agencia}\nNumero da Conta:{*self*.numConta}\n'

f'Saldo:{*self*.saldo}\nCNPJ:{*self*.cnpj}\nCapital:{*self*.capital} \n'

'===============================================================')

def sacar(*self*, *valor\_sacado*):

saldoaux = *self*.saldo

capaux = *self*.capital

status = ''

novosaldo = *self*.saldo - *valor\_sacado*

if novosaldo >=0:

*self*.saldo = novosaldo

status = 'Autorizado'

elif *self*.capital > *valor\_sacado* - *self*.saldo:

vd = *valor\_sacado*-*self*.saldo

*self*.saldo = 0

*self*.capital -= vd

status = 'Autorizado'

else:

status = 'Não Autorizado'

print('===========================================================\n'

'Tipo de conta: Conta Juridica\n'

f'Saldo da conta:{saldoaux}| Saque:{*valor\_sacado*}| Capital investido:{capaux}\n'

f'Status do saque: {status}\n'

f'Saldo da conta = {*self*.saldo}\n'

f'Saldo do Capital investido:{*self*.capital}\n')

class ContaPF(ContaCorrente):

def \_\_init\_\_(*self*, *agencia*, *numConta*, *saldo*, *cpf*, *renda*):

super().\_\_init\_\_(*agencia*, *numConta*, *saldo*)

*self*.cpf = *cpf*

*self*.renda = *renda*

def visualizar(*self*):

print('=============================================================== \n'

f'Agencia:{*self*.agencia}\nNumero da Conta:{*self*.numConta}\n'

f'Saldo:{*self*.saldo}\nCPF:{*self*.cpf}\nRenda:{*self*.renda}\n'

'===============================================================')

def sacar(*self*, *valor\_sacado*):

saldoaux = *self*.saldo

status = ''

novosaldo = *self*.saldo - *valor\_sacado*

if *self*.renda > 5000:

if novosaldo >= -800:

*self*.saldo = novosaldo

status = 'Autorizado'

else:

status = 'Não Autorizado'

else:

if novosaldo >= 0:

*self*.saldo = novosaldo

status = 'Autorizado'

else:

status = 'Não Autorizado'

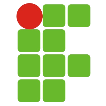
print('===========================================================\n'

'Tipo de conta: Conta Física\n'

f'Saldo da conta:{saldoaux}| Saque:{*valor\_sacado*}| Renda:{*self*.renda}\n'

f'Status do saque: {status}\n'

f'Saldo da conta = {*self*.saldo}\n')



**3.** Além das três classes listadas nas questões anteriores, no mesmo arquivo, crie instâncias para testar os métodos sacar e depositar. Efetue um saque e um depósito para cada tipo de conta e para cada situação:

1. Conta física: um depósito, um saque autorizado, um saque não autorizado
2. Conta jurídica: um depósito, um saque autorizado, um saque não autorizado

pf = ContaPF('0805', '012345', 500.00, 12345678912, 5500.00)

pf.visualizar()

pf.depositar(200.00)

pf.sacar(1400.00)

pf.sacar(200.00)

pj = ContaPJ('0805', '067890', 15000.00, 12341230001, 50000.00)

pj.depositar(500.00)

pj.sacar(40000.00)

pj.sacar(27000.00)

pj.visualizar()