


1 Comece a programar ou gere código com IA.

1 Crie uma pasta chamada oo em sua workspace e crie um arquivo chamado teste_conta.py

```
1 import os
2
3 os.makedirs('oo', exist_ok=True)
4
5 file_path = 'oo/teste_conta.py'
6
7 conteudo = """
8 class Conta:
9     def __init__(self, titular, saldo=0):
10         self.titular = titular
11         self.saldo = saldo
12
13     def depositar(self, valor):
14         self.saldo += valor
15
16     def sacar(self, valor):
17         if valor <= self.saldo:
18             self.saldo -= valor
19             return True
20         return False
21
22     def consultar_saldo(self):
23         return self.saldo
24 """
25
26 with open(file_path, 'w') as f:
27     f.write(conteudo)
28
29 print(f"Arquivo criado em: {file_path}")
30
```

 Arquivo criado em: oo/teste_conta.py

2. Crie a função chamada cria_conta(), que recebe como argumento numero, titular, saldo e limite

```
1 def cria_conta(numero, titular, saldo, limite):
2     conta = {
3         'numero': numero,
4         'titular': titular,
5         'saldo': saldo,
6         'limite': limite
7     }
8     return conta
9
```

3. Dentro de cria_conta(), crie uma variável do tipo dicionário chamada conta com as chaves recebendo os valores dos parâmetros (numero, titular, saldo e limite), e ao final, retorne a conta

```
1 def cria_conta(numero, titular, saldo, limite):
2     conta = {"numero": numero, "titular": titular, "saldo": saldo, "limite": limite}
3     return conta
4
```

4. Crie uma função chamada deposita() no mesmo arquivo teste_conta.py que recebe como argumento uma conta e um valor. Dentro da função, adicione o valor ao saldo da conta

```
1 def cria_conta(numero, titular, saldo, limite):
2     conta = {"numero": numero, "titular": titular, "saldo": saldo, "limite": limite}
3     return conta
4
5 def deposita(conta, valor):
6     conta['saldo'] += valor
7
```

5. Crie outra função chamada saca() que recebe como argumento uma conta e um valor. Dentro da função, subtraia o valor do saldo da conta

```
1 def cria_conta(numero, titular, saldo, limite):
2     conta = {"numero": numero, "titular": titular, "saldo": saldo, "limite": limite}
3     return conta
```

```

4
5 def deposita(conta, valor):
6     conta['saldo'] += valor
7
8 def saca(conta, valor):
9     conta['saldo'] -= valor
10

```

6.E por fim, crie uma função chamada extrato(), que recebe como argumento uma conta e imprime o numero e o saldo

```

1 def cria_conta(numero, titular, saldo, limite):
2     conta = {"numero": numero, "titular": titular, "saldo": saldo, "limite": limite}
3     return conta
4
5 def deposita(conta, valor):
6     conta['saldo'] += valor
7
8 def saca(conta, valor):
9     conta['saldo'] -= valor
10
11 def extrato(conta):
12     print("numero: {} \nsaldo: {}".format(conta['numero'], conta['saldo']))
13

```

7.Navegue até a pasta oo , digite os comandos no arquivo teste_conta.py e teste as funcionalidades

```

1 import os
2 import sys
3
4 os.makedirs('oo', exist_ok=True)
5
6 codigo = """
7 def cria_conta(numero, titular, saldo, limite):
8     conta = {"numero": numero, "titular": titular, "saldo": saldo, "limite": limite}
9     return conta
10
11 def deposita(conta, valor):
12     conta['saldo'] += valor
13
14 def saca(conta, valor):
15     conta['saldo'] -= valor
16
17 def extrato(conta):
18     print("numero: {} \nsaldo: {}".format(conta['numero'], conta['saldo']))
19 """
20
21 with open('oo/teste_conta.py', 'w') as f:
22     f.write(codigo)
23
24 sys.path.append('./oo')
25
26 import teste_conta
27
28 conta = teste_conta.cria_conta('123-7', 'João', 500.0, 1000.0)
29 teste_conta.deposita(conta, 50.0)
30 teste_conta.extrato(conta)
31
32 teste_conta.saca(conta, 20.0)
33 teste_conta.extrato(conta)
34

```

```

↩ numero: 123-7
saldo: 550.0
numero: 123-7
saldo: 530.0

```

8.(Opcional) Acrescente uma documentação para o seu módulo teste_conta.py e utilize a função help() para testá-la.

```

1 import os
2 import sys
3
4 os.makedirs('oo', exist_ok=True)
5
6 codigo = '''
7 """
8 Módulo teste_conta.py
9
10 Este módulo contém funções para criar e manipular contas bancárias simples,

```

```

11 incluindo criação, depósito, saque e exibição de extrato.
12 """
13
14 def cria_conta(numero, titular, saldo, limite):
15     """
16     Cria uma conta bancária.
17
18     Args:
19         numero (str): Número da conta.
20         titular (str): Nome do titular da conta.
21         saldo (float): Saldo inicial da conta.
22         limite (float): Limite da conta.
23
24     Returns:
25         dict: Dicionário representando a conta.
26     """
27     conta = {"numero": numero, "titular": titular, "saldo": saldo, "limite": limite}
28     return conta
29
30 def deposita(conta, valor):
31     """
32     Deposita um valor na conta.
33
34     Args:
35         conta (dict): Conta onde o depósito será realizado.
36         valor (float): Valor a ser depositado.
37     """
38     conta['saldo'] += valor
39
40 def saca(conta, valor):
41     """
42     Realiza um saque na conta.
43
44     Args:
45         conta (dict): Conta onde o saque será realizado.
46         valor (float): Valor a ser sacado.
47     """
48     conta['saldo'] -= valor
49
50 def extrato(conta):
51     """
52     Imprime o número e saldo da conta.
53
54     Args:
55         conta (dict): Conta a ser exibida.
56     """
57     print("numero: {} \\\nsaldo: {}".format(conta['numero'], conta['saldo']))
58 '''
59
60 with open('oo/teste_conta.py', 'w') as f:
61     f.write(codigo)
62
63 sys.path.append('./oo')
64
65 import teste_conta
66
67 help(teste_conta)
68 help(teste_conta.cria_conta)
69 help(teste_conta.deposita)
70 help(teste_conta.saca)
71 help(teste_conta.extrato)
72

```

 Help on module teste_conta:

NAME

teste_conta

FUNCTIONS

cria_conta(numero, titular, saldo, limite)

deposita(conta, valor)

extrato(conta)

saca(conta, valor)

FILE

/content/oo/teste_conta.py

Help on function cria_conta in module teste_conta:

cria_conta(numero, titular, saldo, limite)

```

Help on function deposita in module teste_conta:

deposita(conta, valor)

Help on function saca in module teste_conta:

saca(conta, valor)

Help on function extrato in module teste_conta:

extrato(conta)


```

1.Crie um arquivo chamado conta.py na pasta oo criada no exercício anterior.

```

1 import os
2 import sys
3
4 os.makedirs('oo', exist_ok=True)
5
6 codigo_conta = '''
7 class Conta:
8
9     def __init__(self, numero, titular, saldo, limite):
10         self.numero = numero
11         self.titular = titular
12         self.saldo = saldo
13         self.limite = limite
14
15     def deposita(self, valor):
16         self.saldo += valor
17
18     def saca(self, valor):
19         if self.saldo < valor:
20             return False
21         else:
22             self.saldo -= valor
23             return True
24
25     def extrato(self):
26         print("numero: {} \\\nsaldo: {}".format(self.numero, self.saldo))
27
28     def transfere_para(self, destino, valor):
29         retirou = self.saca(valor)
30         if retirou == False:
31             return False
32         else:
33             destino.deposita(valor)
34             return True
35 '''
36
37 with open('oo/conta.py', 'w') as f:
38     f.write(codigo_conta)
39
40 codigo_teste = '''
41 from conta import Conta
42
43 conta1 = Conta('123-4', 'João', 120.0, 1000.0)
44 conta2 = Conta('567-8', 'Maria', 200.0, 1000.0)
45
46 conta1.deposita(50.0)
47 conta1.extrato()
48
49 conta1.saca(20.0)
50 conta1.extrato()
51
52 sucesso = conta1.transfere_para(conta2, 100.0)
53 print("Transferência realizada?", sucesso)
54
55 conta1.extrato()
56 conta2.extrato()
57 '''
58
59 with open('oo/conta_teste.py', 'w') as f:
60     f.write(codigo_teste)
61
62 sys.path.append('./oo')
63
64 get_ipython().system('python3 oo/conta_teste.py')
65

```

 numero: 123-4
saldo: 170.0
numero: 123-4
saldo: 150.0
Transferência realizada? True
numero: 123-4
saldo: 50.0
numero: 567-8
saldo: 300.0