Conceitos de python

A Linguagem Python foi concebida no fim dos anos 80. A primeira ideia de implementar o Python surgiu mais especificamente em 1982 enquanto Guido Van Rossum trabalhava no CWI (*Centrum Wiskunde & Informatica*, Centro de Matemática e Ciência da Computação) em Amsterdã, Holanda, no time de desenvolvimento da Linguagem ABC.

Posteriormente, em 1987, com o fim da linguagem ABC, Guido foi transferido para o grupo de trabalho Amoeba — um sistema operacional Microkernel liderado por Andrew Tanenbaum. Foi neste grupo que Guido percebeu a necessidade de uma linguagem para escrever programas intermediários, algo entre o C e o Shell Script.

Conceitos de python

Os objetivos do projeto da linguagem eram: produtividade e legibilidade. Em outras palavras, Python é uma linguagem que foi criada para produzir código bom e fácil de manter de maneira rápida. Entre as características da linguagem que ressaltam esses objetivos estão:

- baixo uso de caracteres especiais, o que torna a linguagem muito parecida com *pseudo-código executável*;
- o uso de identação para marcar blocos;
- quase nenhum uso de palavras-chave voltadas para a compilação;
- coletor de lixo para gerenciar automaticamente o uso da memória;
- etc.

Conceitos de python

Usando Visual Studio Code para programar em python com flask, plugins utilizados:

- Better Jinja (autor Samuel Colvin),
- Bootstrap 4,
- Font awesome 4 e 5 free & pro sn (autor Ashok Koyi),
- Python (autor Microsoft)

Conceitos de python

Python v3

We will use **conda** (to install *Python v3*), which is an open source package and environment management system that runs on **Linux**, **Windows** and **macOS**. A free minimal installer for *conda* is **Miniconda**, that includes only *conda* itself, *Python*, and a small number of other useful packages.
After **Miniconda is installed**, you can **use conda** to install any other packages and create environments. Example on how to create environment with *Python v3.6*:
conda create -n name-of-the-environment python=3.6
Example on how to list all existing environments:
conda env list
Example on how to activate environment:
conda activate name-of-the-environment

Conceitos de python

Para gerar um executável é necessário instalar pyinstaler:
C:\> pip install pyinstaller

Para criar o executável do programa com arquivos de dependências, execute:
C:\programa> pyinstaller arquivo.py

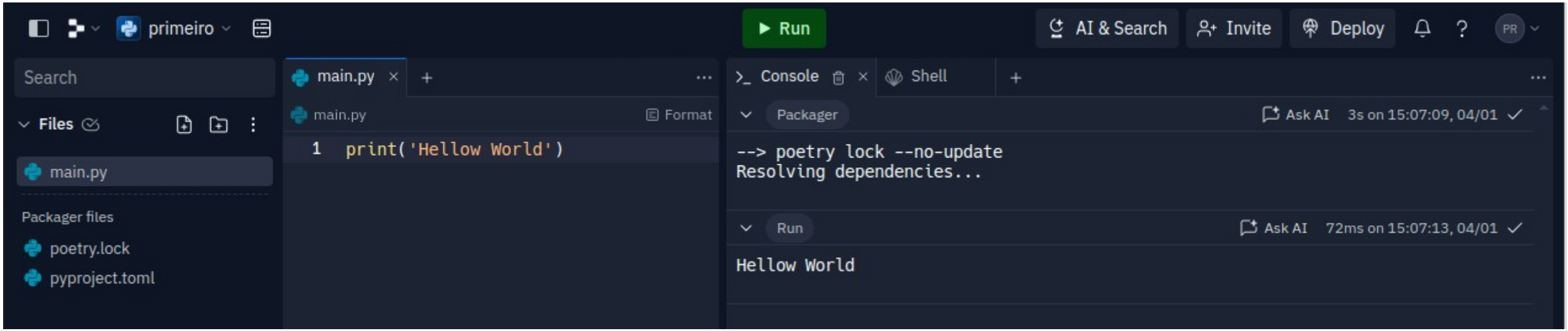
Criar somente o .exe
C:\programa> pyinstaller --onefile arquivo.py

Criar somente o .exe e permitir escrita
C:\programa> pyinstaller --onefile -w arquivo.py

Conceitos de python

Os arquivos em python devem ser salvos com a extensão .py

Por exemplo: main.py



Observação: Primeiro exemplo no ambiente web – replit.com

Conceitos de python

Variáveis

Assim como em outras linguagens, o Python pode manipular variáveis básicas como strings (palavras ou cadeias de caracteres), inteiros e reais (float). Para criá-las, basta utilizar um comando de atribuição, que define seu tipo e seu valor.

```
mensagem = 'Exemplo de mensagem!'
n = 25
pi = 3.1415926535899931
print(mensagem)
print (n)
print (pi)
```

Conceitos de python

Comando de entrada de informação

```
print("Acesso ao sistema \n")
print("Digite seu login e senha de acesso \n")
login = input("Login:")
print("Seu login é ",login)
senha = input("Senha:")
print("Sua senha e ",senha)
```

Conceitos de python

Comando de entrada de informação

```
print("Acesso ao sistema \n")
print("Digite seu login e senha de acesso \n")
login = input("Login:")
print("Seu login é %s" %login)
senha = input("Senha:")
print("Sua senha e %s" %senha)
```

Conceitos de python

Comando de entrada de informação

```
print("Acesso ao sistema \n")
print("Digite seu login e senha de acesso \n")
login = input("Login:")
senha = input("Senha:")
print("Login:%s, Senha:%s" %(login,senha))
```

Conceitos de python

Cálculo: adição de dois números inteiros

```
num1 = int(input("Digite a primeira nota: "))
num2 = int(input("Digite a segunda nota: "))
print(num1+num2)
```

Conceitos de python

Cálculo: adição de dois números reais

```
num1 = float(input("Digite a primeira nota: "))
num2 = float(input("Digite a segunda nota: "))
print(num1+num2)
```

Conceitos de python

Cálculos

Dessa forma, podemos observar que a:

- Adição utiliza o símbolo +
print(num1+num2)

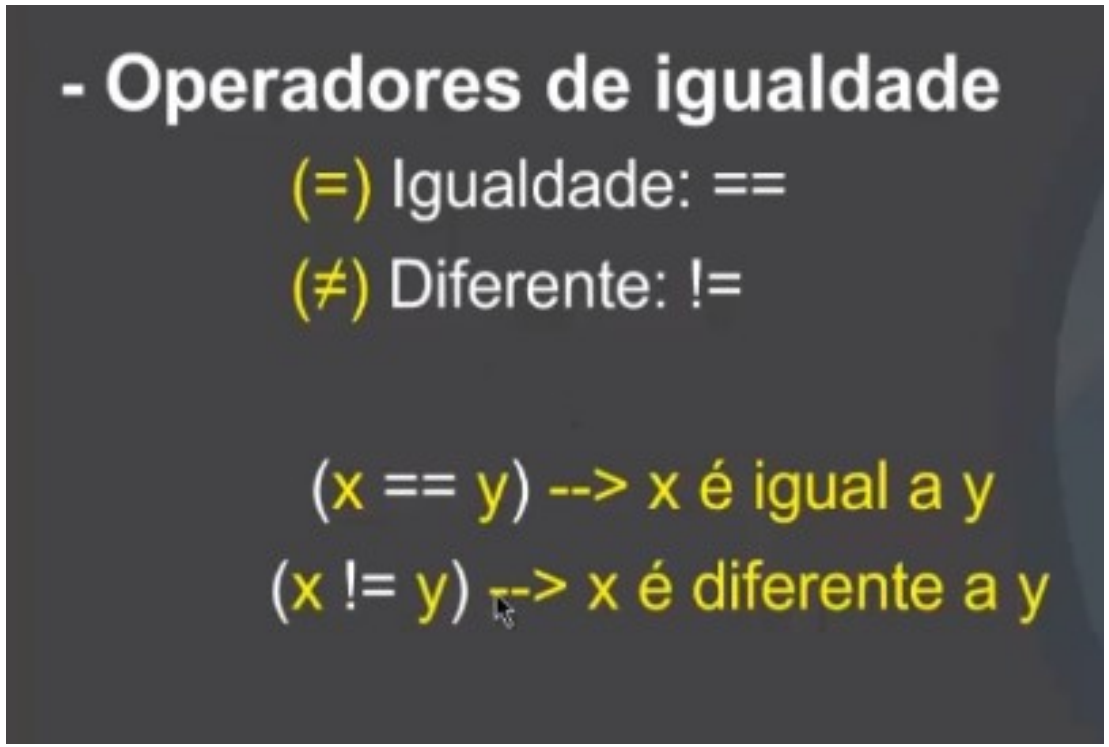
- Subtração utiliza o símbolo -
print(num1-num2)

- Multiplicação utiliza o símbolo *
print(num1*num2)

- Divisão utiliza o símbolo /
print(num1/num2)

Conceitos de python

Operadores relacionais



Conceitos de python

Operadores

```
# Exponent
print(5 ** 2)
# This is the same as 5 * 5

# Floor Division
print(5 // 2)

# Modulus
print(5 % 2)
```

```
25
2
1
```

Conceitos de python

Operador Lógico: AND

```
print(True and True)
print(True and False)
print(False and False)
```

```
True
False
False
```

a	b	a and b
True	True	True
True	False	False
False	True	False
False	False	False

Conceitos de python

Operador Lógico: OR

a	b	a or b
True	True	True
True	False	True
False	True	True
False	False	False

Conceitos de python

Operador Lógico: NOT

```
print(not True)    False
```

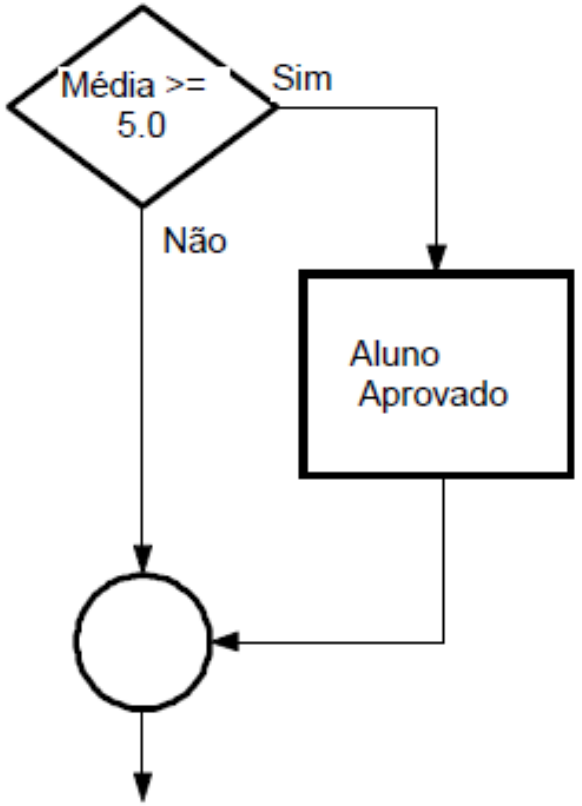
a	not a
True	False
False	True

Conceitos de python

Comando de decisão IF

if, if-else, if-elif-else

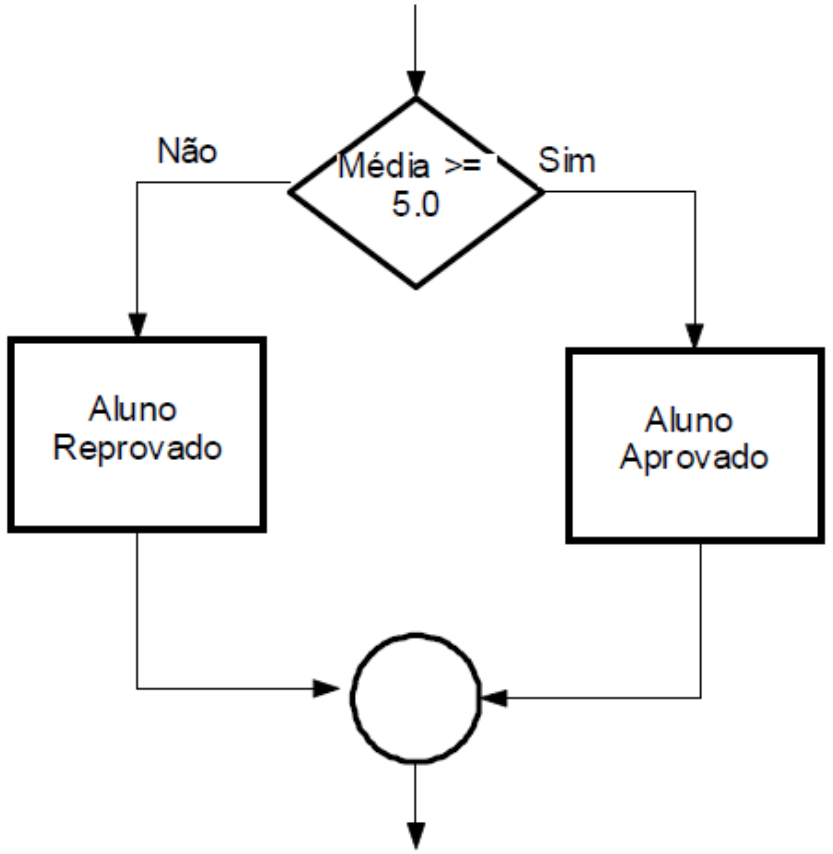
```
idade = 18
if idade < 20:
   print('Você é jovem!')
```



Conceitos de python

Comando de decisão if-else

```python
nota1 = float(input("Digite a primeira nota: "))
nota2 = float(input("Digite a segunda nota: "))
media = (nota1 + nota2) / 2
print("A média é: ", media)
if media >= 5.0:
    print("Aprovado")
else:
    print("Reprovado")
```

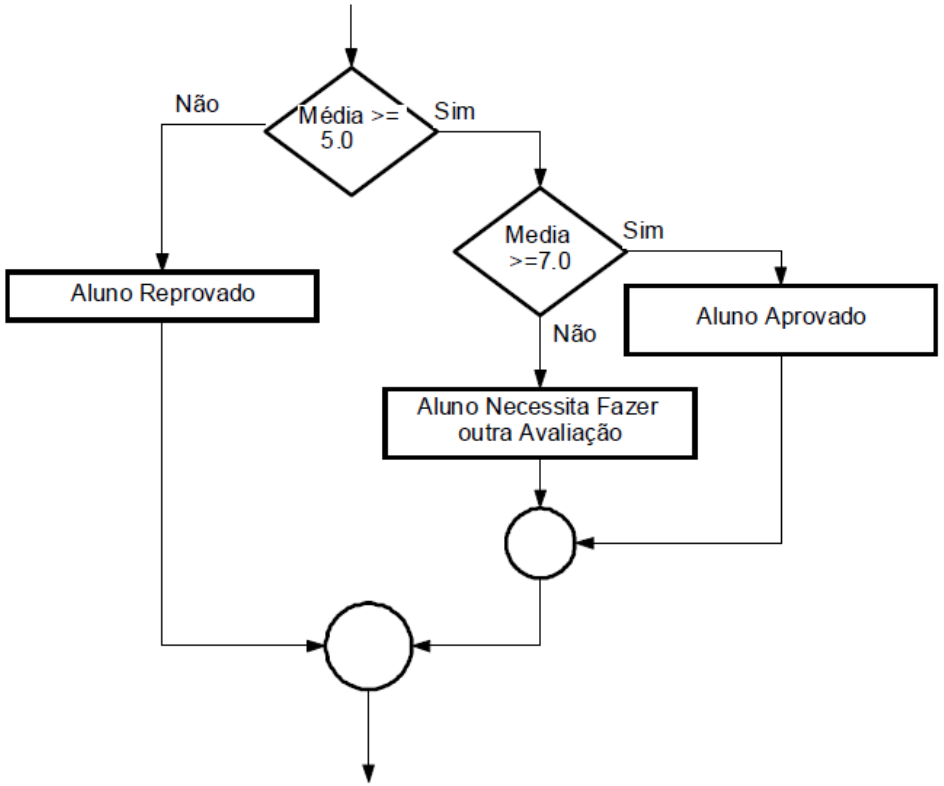


Conceitos de python

Comando de decisão if-else

```python
nota1 = float(input("Digite a primeira nota: "))
nota2 = float(input("Digite a segunda nota: "))
media = (nota1 + nota2) / 2
print("A média é: ", media)

if media >= 5.0:
    if media >= 7.0:
        print("Aprovado")
    else:
        print("Recuperação")
else:
    print("Reprovado")
```



Conceitos de python

Comando de decisão if-elif-else

```
idade = 18
if idade < 12:
                print('criança')
elif idade < 18:
                print('adolescente')
elif idade < 60:
                print(adulto')
else:
                print('idoso')
```

Conceitos de python

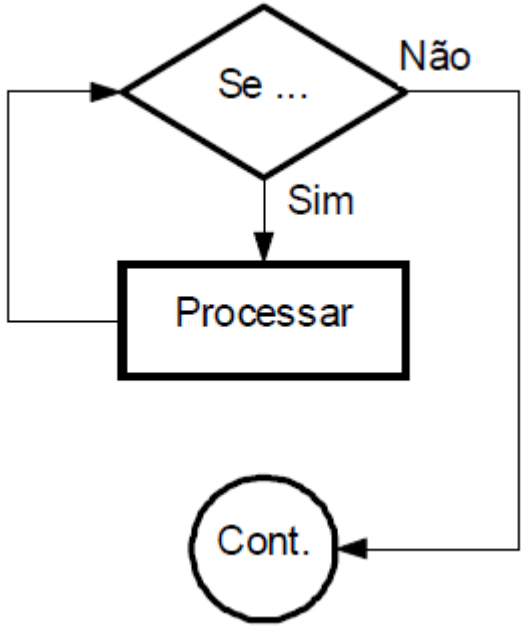
Exercício

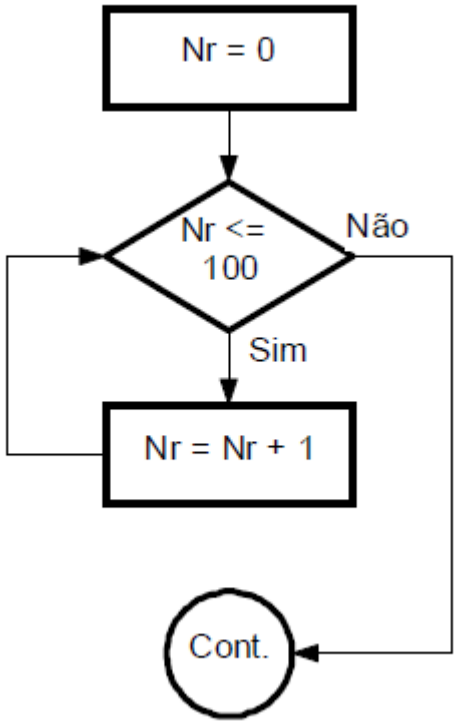
Usando o exemplo dado, complemente o código que avalia a média para aprovação
Considere aprovada a pessoa com média igual ou superior a 7.0
Considere reprovada a pessoa com média inferior a 5.0
Considere em recuperação a pessoa com média entre 5.0 e 6.9
A nota da recuperação deve ser informada e deve substituir a menor nota parcial anterior
A nova média deve verificar e informar se o aluno foi aprovado ou não

Conceitos de python

Estrutura de repetição: WHILE

while <condição_verdadeira>:
**   <executa o bloco>**





Loop infinito

```
nome = ‘inicio’
while nome:
    input(“Insira um nome: “)
```

Interrompendo loop infinito

```
nome = 'nome'
while nome:
  nome = input("Insira um nome: ")
  if (nome == 'sair'):
    print('Programa encerrado')
    break
```

```
contador = 0
while (contador < 5):
        print(contador)
        contador    = contador + 1
```

Conceitos de python

Estrutura de repetição: WHILE/ELSE

```
contador = 0
while (contador < 5):
        print(contador)
        contador = contador + 1
else:
        print("O loop while foi encerrado com sucesso!")
```

Conceitos de python

Estrutura de repetição: FOR

```
lojas = ['Rio de Janeiro', 'São Paulo', 'Belo Horizonte', 'Curitiba']

for loja in lojas:
    print(loja)
print('Acabou o FOR')
```

```
Rio de Janeiro
São Paulo
Belo Horizonte
Curitiba
Acabou o FOR
```

```
for i in range(4):
    print(i)
    print(lojas[i])
```

```
0
Rio de Janeiro
1
São Paulo
2
Belo Horizonte
3
Curitiba
```

FOR/ELSE

```
nomes = ['Ana', 'Maria', 'João', 'Pedro']
for n in nomes:
    print(n)
else:
    print("Nomes listados com sucesso")
```

Conceitos de python

Exercício

1. Utilize uma estrutura de repetição para mostrar seu nome colocando uma letra por linha
   nome = ‘Luiz’

Saída do programa:
L
u
i
z

2. Qual a diferença entre BREAK e CONTINUE? Escreva um código que exemplifique esse uso

Conceitos de python

Vetor

Exemplo: vetor1.py

```
vetor = [0, 0, 0]
#Repetição para obtenção de valores
for v in range(0, 3):
    vetor[v] = int(input(f'Digite um valor:[{v}]:'))
#linha de resultado
print(vetor)
```

Conceitos de python

Vetor

Exemplo: vetor2.py

```
vetor = [0, 0, 0]
#Repetição para obtenção de valores
for v in range(0, 3):
    vetor[v] = int(input(f'Digite um valor:[{v}]:'))
#linha de resultado
print('-=' * 30)
#Repetição para mostrar valores digitados
for v in range(0, 3):
    print(f'[{vetor[v]:^5}]', end='')
```

Conceitos de python

Matriz

Exemplo: matriz1.py

```
def main():
    a = [0, 1, 2, 3, 4]
    b = a[:]
    b[1] = 7
    print("a = ", a)
    print("b = ", b)
main()
```

Resultado na tela

```
a =  [0, 1, 2, 3, 4]
b =  [0, 7, 2, 3, 4]
```

Conceitos de python

Matriz

Exemplo: matriz2.py

```
matriz = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]
for l in range(0, 3):
    for c in range(0, 3):
        matriz[l][c] = int(input(f'Digite um valor:[{l},{c}]:'))
print(matriz)
```

Conceitos de python

Matriz

Exemplo: matriz3.py

```
matriz = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]
#Repetição para obtenção de valores
for l in range(0, 3):
    for c in range(0, 3):
        matriz[l][c] = int(input(f'Digite um valor:[{l},{c}]:'))
#linha de resultado
print('-=' * 30)
#Repetição para mostrar valores digitados
for l in range(0, 3):
    for c in range(0, 3):
        print(f'[{matriz[l][c]:^5}]', end='')
    print()
```

Conceitos de python

Exercícios

1. Faça um programa em python, que recebe uma distância entre zero até um metro. Faça três condicionais para essa distância, onde em uma distância ele acelera na outra distancia ele freia e a outra ele vira a esquerda.

2. Faça um programa em python, onde uma temperatura de até 30 graus celcius é informada. Conforme o valor recebido, informe se está frio (< 10), morno ou quente (>23). Como bônus, o tratamento de valores menores que 0 ou maiores que 30 apresentam uma mensagem informando o limite de parâmetros de entrada.