

Lógica de Programação

Aula 7 - 08/05/2023



https://bit.ly/23_1_cc_logica_n



O que temos para **hoje**?

- O que vimos até agora;
- Situação problema;
- Solução do problema;
- Estrutura while;
- Um novo tipo de variável;
- Estrutura for;
- Lista de exercícios;



O que vimos até agora



O que vimos até agora

- Entrada de dados;
- Declaração de variáveis;
- Operações matemáticas;
- Estruturas de sequência;
- Estruturas de seleção;
- Estruturas de repetição;
- Funções;
- Saída de dados;



Situação problema



Situação **problema**

- Faça um programa em Python que calcule a soma de todos os número de 0 até 100;
- Como vocês fariam?



Situação problema

- Faça um programa em Python que calcule a soma de todos os número de 0 até 100;
- Solução 1:

 slide7.py

```
1 soma = 0 + 1 + 2 + 3 + 4 + 5 + ... + 100
```



Situação **problema**

☉ Uma solução melhor?



Solução do problema



Solução do problema

- Como em situações do mundo real, em um programa, pode ser necessário repetir um trecho diversas vezes até que uma determinada condição seja satisfeita;
- Em programação, para casos como esse, são usadas estruturas conhecidas como iteração (não é interação!), repetição, laço ou loop;
- Python implementa duas estruturas de repetição: while e for.



Estrutura while



Estrutura **while**

```
1  while <condição (ou um conjunto delas) for verdadeira>:  
2      #Instruções a serem executadas  
3      #Instruções a serem executadas após o encerramento do loop
```



Estrutura **while**

- Voltando ao exemplo anterior...
- Faça um programa em Python que calcule a soma de todos os número de 0 até 100;



Estrutura **while**

🕒 Solução 2:

 slide14.py

```
1  soma = 0
2  numeros = 0
3
4  √ while numeros <= 100:
5      soma = soma + numeros
6      numeros = numeros + 1
7
8  print(soma)
```



Estrutura **while**

☉ CUIDADO!!!

 slide15.py

```
1   soma = 0
2   numeros = 0
3
4   while numeros <= 100:
5       soma = soma + numeros
6
7   print(soma)
```



Estrutura **while**

- Exemplo: faça um programa em Python que escreva na tela a frase “Olá, mundo!” 1000 vezes;



Estrutura **while**

 slide17.py

```
1   repeticoes = 1000
2
3   while repeticoes > 0:
4       print("Olá, mundo!")
5       repeticoes = repeticoes - 1
```



Estrutura **while**

- Exemplo: faça um programa que peça uma nota, entre zero e dez. Mostre uma mensagem caso o valor seja inválido e continue pedindo até que o usuário informe um valor válido;



Estrutura **while**

 slide19.py

```
1  nota = input("Digite uma nota: ")
2  nota = float(nota)
3
4  while nota < 0 or nota > 10:
5      print("Notá inválida!")
6      nota = input("Digite uma nota: ")
7      nota = float(nota)
8
9  print("Obrigado!")
```



Estrutura **while**

- Exemplo: faça um programa que leia um nome de usuário e a sua senha e não aceite a senha igual ao nome do usuário, mostrando uma mensagem de erro e voltando a pedir as informações;



Estrutura **while**

 slide21.py

```
1  usuario = input("Digite um usuário: ")
2  senha = input("Digite uma senha: ")
3
4  √ while usuario == senha:
5      print("Erro! Usuário e senha não podem ser iguais!")
6      usuario = input("Digite um usuário: ")
7      senha = input("Digite uma senha: ")
8
9  print("Obrigado!")
```



Um novo tipo de variável



Um novo tipo de **variável**

- ⦿ Até agora vimos que uma variável armazena apenas um valor, certo?
- ⦿ Mas é comum querermos armazenar vários valores em uma mesma variável;
- ⦿ Exemplo: armazenar as notas de uma turma;



Um novo tipo de **variável**

 slide24.py

```
1  nota1 = 5
2  nota2 = 10
3  nota3 = 7
4  nota4 = 5
```




Um novo tipo de **variável**

- Para esses casos (uma variável armazenando vários valores), utilizamos um tipo de variável chamado lista;
- Uma lista, ou vetor, ou array, é uma variável que armazena diversos valores, do mesmo tipo ou de diferentes tipos;



Um novo tipo de **variável**

 slide26.py

```
1  lista = [5,10,7,5]
2
3  print(lista)
```



Um novo tipo de **variável**

- Como fazer para acessar um valor de uma lista?
- Index!



Um novo tipo de **variável**

$L = [20, 'Jessa', 35.75, [30, 60, 90]]$

$\uparrow \quad \uparrow \quad \uparrow \quad \uparrow$

$L[0] \quad L[1] \quad L[2] \quad L[3]$



Um novo tipo de **variável**

- Escrever o segundo item da lista anterior;



Um novo tipo de **variável**

 slide30.py

```
1  L = [20, 'Jessa', 35.75, [30,60,90]]  
2  
3  print(L[1])
```



Um novo tipo de **variável**

- O que se pode fazer com uma lista?



Um novo tipo de **variável**

slide32.py

```
1  print("Criar uma lista vazia: ")
2  L = []
3  print(L)
4  print("Adicionar um item ao final da lista:")
5  L.append(20)
6  L.append('Jessa')
7  L.append(35.75)
8  L.append([30,60,90])
9  L.append("Fabricio")
10 print(L)
11 print("Verificar o tamanho da lista:")
12 print(len(L))
13 print("Inserir um item em uma posição específica:")
14 L.insert(0,"Araujo")
15 print(L)
16 print("Remover um valor e excluir o primeiro a aparecer:")
17 L.remove("Araujo")
18 print(L)
19 print("Remover um valor em uma posição específica")
20 L.pop(len(L)-1)
21 print(L)
```




Um novo tipo de **variável**

- Exercício: escreva um programa em Python que leia 8 bits (um a um) e diga quantos 1s e 0s compõem o byte lido;
- Utilize uma lista para armazenar os bits;
- Utilize while para ler os bits e para contabilizar os valores de 1s e 0s;



Um novo tipo de **variável**

slide34.py

```
1  bits = []
2  n_bits = 8
3  contador = 0
4
5  while contador < n_bits:
6      bit = input("Digite um bit: ")
7      bit = int(bit)
8      bits.append(bit)
9      contador = contador + 1
10
11  index = 0
12  n0s = 0
13  n1s = 0
14  while index < n_bits:
15      if bits[index] == 0:
16          n0s = n0s + 1
17      else:
18          n1s = n1s + 1
19
20      index = index + 1
21
22  print(n0s)
23  print(n1s)
```



Um novo tipo de **variável**

- Gerando uma lista com valores sucessivos:

```
1  lista0_10 = list(range(10))  
2  
3  print(lista0_10)
```



Um novo tipo de **variável**

- O que é possível fazer com a função range:
- <https://docs.python.org/3/library/stdtypes.html?highlight=range#range>



Um novo tipo de **variável**

- Faça um programa em Python que calcule a soma de todos os número de 0 até 100;
- Utilize uma lista para armazenar o range dos números;
- Utilize while para calcular a soma;



Um novo tipo de **variável**

📄 slide38.py

```
1  numeros = list(range(101))
2  soma = 0
3  index = 0
4
5  while index < len(numeros):
6      soma = soma + numeros[index]
7      index = index + 1
8
9  print(soma)
```



Estrutura for



Estrutura **for**

```
1   for <variável> in range([maneira_1|maneira_2]):  
2       #Instruções a serem executadas  
3       #Instruções a serem executadas após o fim do loop
```




Estrutura **for**

- Faça um programa em Python que calcule a soma de todos os número de 0 até 100;
- Utilize uma lista para armazenar o range dos números;
- Utilize for para calcular a soma;



Estrutura **for**

 slide42.py

```
1  numeros = list(range(101))
2  soma = 0
3
4  for index in numeros:
5      soma = soma + numeros[index]
6
7  print(soma)
```



Estrutura **for**

- Exercício: escreva um programa em Python que leia 8 bits (um a um) e diga quantos 1s e 0s compõem o byte lido;
- Utilize uma lista para armazenar os bits;
- Utilize for para ler os bits e para contabilizar os valores de 1s e 0s;



Estrutura **for**

slide44.py

```
1  bits = []
2  n_bits = list(range(8))
3  n0s = 0
4  n1s = 0
5
6  for idx in n_bits:
7      bit = input("Digite um bit: ")
8      bit = int(bit)
9      bits.append(bit)
10
11  for bit in bits:
12      if bit == 0:
13          n0s = n0s + 1
14      else:
15          n1s = n1s + 1
16
17  print(n0s)
18  print(n1s)
```



Estrutura **for**

- Exercício: escreva um programa em Python que pergunte ao usuário 5 números e escreva o menor número entre eles;



Estrutura for

slide46.py

```
1  numeros = []
2  total_numeros = list(range(5))
3
4  for n in total_numeros:
5      num = input("Digite um número: ")
6      num = int(num)
7      numeros.append(num)
8
9  menor_numero = numeros[0]
10
11  for n in numeros:
12      if n < menor_numero:
13          menor_numero = n
14
15  print(numeros)
16  print(menor_numero)
```



Estrutura for

- Exercício: supondo que a população de um país A seja da ordem de 80000 habitantes com uma taxa anual de crescimento de 3% e que a população de B seja 200000 habitantes com uma taxa de crescimento de 1.5%;
- Faça um programa que calcule e escreva o número de anos necessários para que a população do país A ultrapasse ou iguale a população do país B, mantidas as taxas de crescimento;



Estrutura for

slide48.py

```
1  popA = 80000
2  taxA = 3/100
3
4  popB = 200000
5  taxB = 1.5/100
6
7  anos = 0
8
9  while popA <= popB:
10     aumentoA = popA * taxA
11     popA = popA + aumentoA
12
13     aumentoB = popB * taxB
14     popB = popB + aumentoB
15
16     anos = anos + 1
17
18  print(anos)
```




Lista de exercícios



Lista de **exercícios**

- Acessar o arquivo lista3.pdf no teams da disciplina;
- Resolver as questões;



Obrigado!

Alguma pergunta?

Contato:

☎ 040601692@prof.unama.br