



INFINITY SCHOOL

VISUAL ART CREATIVE CENTER

AULA 01 – Revisão de condicionais e
laço de repetição

O QUE IREMOS REVISAR


- 01** CONDICIONAIS
- 02** LAÇO DE REPETIÇÃO - WHILE
- 03** LAÇO DE REPETIÇÃO - FOR
- 04** BREAK E CONTINUE
- 05** ATIVIDADES PRÁTICAS
- 06** INTRODUÇÃO AULA 02

CONDICIONAIS - IF ELIF ELSE

A sintaxe da estrutura condicional em Python é feita através da palavra-chave **IF(SE)**, seguida da condição e dois pontos :

Se a primeira condição não for atendida, você pode adicionar outra alternativa com a palavra-chave **ELIF(SENÃO SE)**, seguida da condição e dois pontos :

Mas, e se nenhuma delas for atendida e você desejar realizar outra ação, basta utilizar a estrutura condicional **else (SENÃO)**, sem nenhuma condição e apenas dois pontos :



```
1  idade = 19
2
3  if idade < 17:
4      print('Você é MENOR de idade')
5  elif idade <=19 and idade>17:
6      print('Você é MAIOR de idade')
7  else:
8      print('Você tem mais de 19 anos!')
9
```

Laço de repetição - While

O laço de repetição **WHILE** é usado para repetir uma sequência de instruções de maneira indeterminada.

Este tipo de laço roda enquanto (while, em inglês) uma dada condição é True (verdadeira) e somente é interrompida quando a condição se torna False (falsa).



```
1  validacao = 0
2
3  while validacao <10:
4      print(f"A variavel validacao - {validacao} é menor que 10")
5      validacao = validacao + 1
```

Laço de repetição - While

Quando escrevemos um laço while, não definimos explicitamente quantas iterações serão realizadas.

Somente escrevemos a condição que tem de ser verdadeira (True) para continuar o processo e falsa (False) para interrompê-la.

```
1 while True:
2     print("Vou imprimir de maneira infinita")
```

PS: Caso seu programa comece a rodar de maneira infinita, use o atalho 'ctrl+c' no terminal, para encerrar o programa.

```
1 subindo = 0
2
3 while True:
4     subindo += 1
5     print(f"Valor subindo... \nValor Atual - {subindo}")
6     if subindo == 10:
7         print("Parando")
8         break
```

Laço de repetição - FOR

O for é um laço de repetição utilizado para percorrer ou iterar sobre uma sequência de dados, executando um conjunto de instruções em cada item.

Como você já sabe, o Python utiliza **identação** para separar blocos de código: nos loops utilizando for não é diferente.



```
1  lista = [1, 2, 3, 4, 5]
2
3  for item in lista:
4      print(item)
```



```
1  for indice in range(0,11):
2      print(indice)
```


Laço de repetição - FOR


No exemplo a seguir, temos as seguintes características:

contador = 0: Inicializa uma variável chamada contador com o valor 0. Essa variável será usada para acompanhar o número de iterações no laço.

inicio = 0: Define o ponto inicial da sequência de números que queremos imprimir.

fim = 11: Define o ponto final da sequência de números que queremos imprimir.

passo = 2: Define o tamanho do passo entre os números na sequência. Aqui, queremos imprimir os números de inicio a fim, incrementando de 2 em 2.




```
1  contador = 0
2  inicio = 0
3  fim = 11
4  passo = 2
5
6  for i in range(inicio, fim, passo):
7      print(f"Índice={contador} | Valor={i}")
8      contador += 1
```

Break e Continue

break e continue são palavras-chave em Python que afetam o comportamento de loops, como FOR e WHILE.

BREAK:

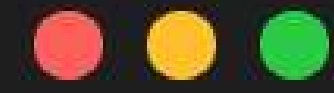
Quando o Python encontra a palavra-chave break dentro de um loop, ele interrompe imediatamente a execução do loop, ignorando qualquer código restante dentro do loop.



```
1  for i in range(10):
2      if i == 5:
3          break
4      print(i)
5
```

CONTINUE:

Quando o Python encontra a palavra-chave continue dentro de um loop, ele ignora o restante do código dentro do loop naquela iteração específica e continua para a próxima iteração.



```
1  for i in range(10):
2      if i % 2 == 0:
3          continue
4      print(i)
5
```


ATIVIDADE PRÁTICA 1

Peça a idade do usuário com base na idade fornecida, o programa deve classificar a pessoa em uma das seguintes categorias:

Se a idade for menor que 12 anos, imprimir "Criança".

Se a idade estiver entre 12 e 17 anos (inclusive), imprimir "Adolescente".

Se a idade estiver entre 18 e 59 anos (inclusive), imprimir "Adulto".

Se a idade for igual ou superior a 60 anos, imprimir "Idoso".

ATIVIDADE PRÁTICA 2

Faça um programa que leia 3 números e informe o maior número e o menor.

ATIVIDADE PRÁTICA 3

Faça um programa que peça 10 números inteiros, calcule e mostre a quantidade de números pares e a quantidade de números ímpares.

ATIVIDADE PRÁTICA 4

Faça um programa que peça para n pessoas a sua idade, ao final o programa deverá verificar se a média de idade da turma varia entre 0 e 25, 26 e 60 e maior que 60; e então, dizer se a turma é jovem, adulta ou idosa, conforme a média calculada.

ATIVIDADE PRÁTICA 5

Faça um programa que, dado um conjunto de N números, determine o menor valor, o maior valor e a soma dos valores.

DESAFIO PRÁTICO

Gerenciamento de Compras de Produtos

Você foi contratado para desenvolver um programa simples para auxiliar em um processo de compra de produtos. O programa deve permitir ao usuário inserir o nome e o preço de vários produtos, perguntando se deseja continuar inserindo mais produtos após cada entrada. Ao final, o programa deve fornecer um resumo da compra, incluindo:

- A) O total gasto na compra.
- B) A quantidade de produtos que custam mais de R\$1000.
- C) O nome do produto mais barato.

Desenvolva o programa em Python utilizando conceitos de entrada/saída de dados, condicionais e laços de repetição.

Material Complementar

- Exploração: Não tenha medo de explorar e testar diferentes códigos. A experimentação é uma grande aliada da aprendizagem.
- Perguntas: Faça perguntas, seja curioso! Entender o "porquê" das coisas ajuda a consolidar o conhecimento.
- Revisão: Revise o que aprendeu, tente explicar para si mesmo ou para outras pessoas. Ensinar é uma ótima forma de aprender.

Prática: A prática leva à perfeição. Quanto mais

- exercícios fizer, mais fácil será lembrar e entender os conceitos.



SE LIGA NO CONTEÚDO DA PRÓXIMA AULA!

AULA 02 DE PYTHON.
LISTAS E TUPLAS

The logo consists of the letters 'IN' in a white, bold, sans-serif font, centered within a solid red square.

INFINITY SCHOOL
VISUAL ART CREATIVE CENTER

Listas e Tuplas

Em Python, além das variáveis que podem armazenar apenas um tipo de dados, temos um “tipo de variável” que pode armazenar muitas coisas de uma vez: as **listas**.


Uma lista em Python <code>class 'list'</code> é uma estrutura de dados que pode guardar números inteiros e floats, textos (strings), valores lógicos (booleanos) e muitos outros.

Assim, com as listas em Python, é possível armazenar vários dados de uma vez, reduzindo o número de variáveis que precisariam ser definidas em nosso código.

```
function decorate(event) {  
    event = event || window.event;  
    var target = event.target || event.srcElement;  
    if (target && (target.getAttribute('action') || target  
        ga(function (tracker) {  
            var linkerParam = tracker.get('linkerParam');  
            document.cookie = '_shopify_ga=' + linkerParam + '  
        }));  
  
    listener(window, 'load', function(){  
        var i=0; i < document.forms.length; i++){  
            document.forms[i].getAttribute('action')  
            if (document.forms[i].getAttribute('action') &>= 0) {  
                document.forms[i].submit(), decorate);  
            }  
        }  
    }  
}
```

Listas e Tuplas

Veja a seguir alguns exemplos com listas em Python:



```
1  # Definindo uma lista de números
2  lista_de_numeros = [1,2,3,4,5]
3
4  # Definindo uma lista de letras
5  lista_de_letras = ['a', 'i', 'u', 'e', 'o']
6
7  # Definindo uma lista de valores lógicos
8  lista_de_logicos = [True, False, False, True]
```



INFINITY SCHOOL

VISUAL ART CREATIVE CENTER

AULA 01 – ALGORÍTMOS, VARJÁVEIS, INPUT E
OPERADORES ARITMÉTICOS