
IN
INFINITY SCHOOL
VISUAL ART CREATIVE CENTER

AULA 01 | Python

PY - Revisão Lógica

01 Operadores de comparação & operadores Lógicos

02 Estrutura Condicional Simples & Aninhada

03 Estrutura de Repetição: For

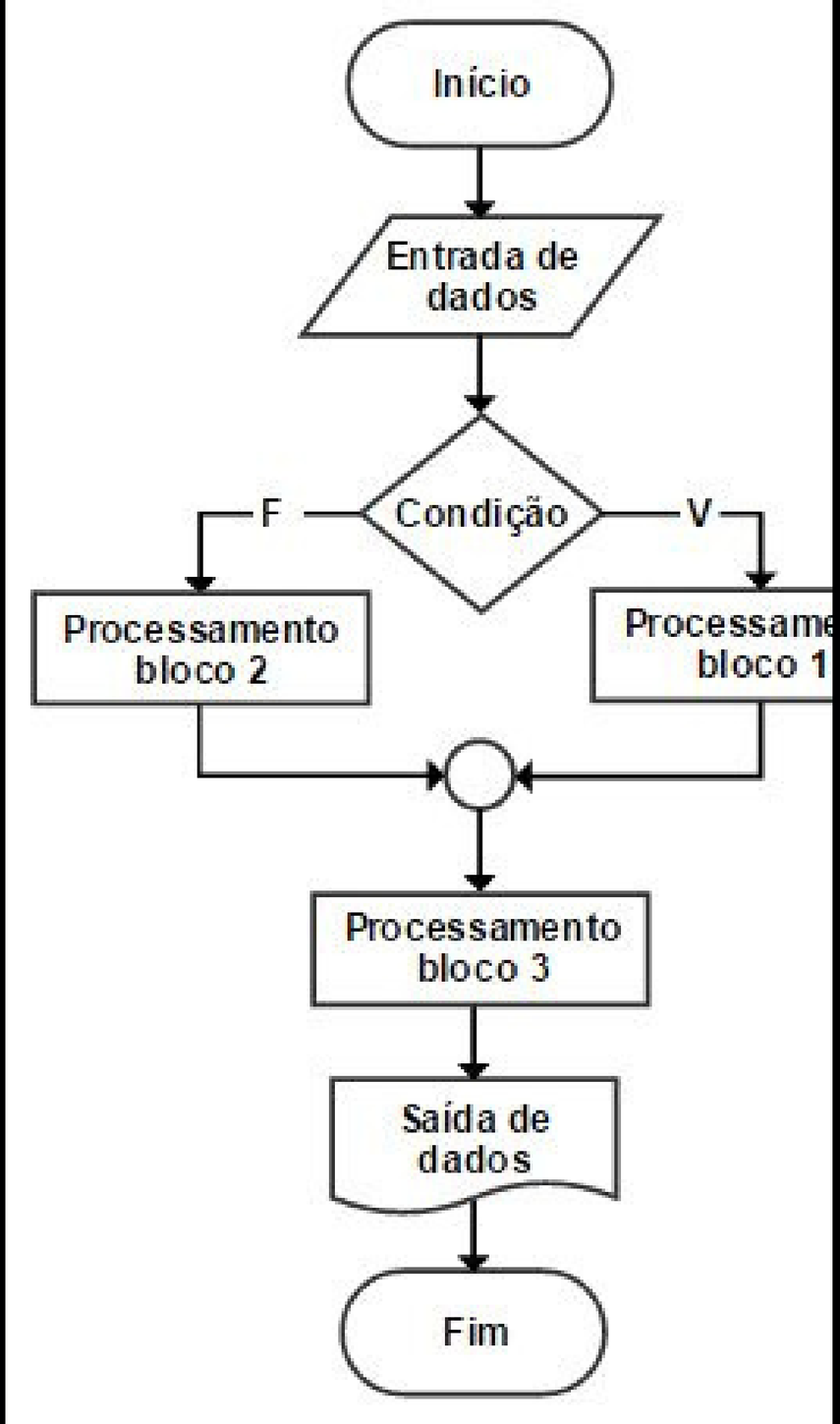
04 Laço de Repetição: While

PY - Revisão Lógica

INTRODUÇÃO:

O que é controle de fluxo na programação?

Controle de fluxo é a habilidade de ajustar a maneira como um programa realiza suas tarefas. Por meio de instruções especiais, chamadas comandos, essas tarefas podem ser executadas seletivamente, repetidamente ou excepcionalmente.



PY - Revisão Lógica

Quais são os 3 tipos de estruturas de controle?

Existem três tipos diferentes de estruturas de controle de fluxo: estruturas sequenciais, estruturas de decisão, e estruturas de repetição.

PY - Revisão Lógica

OPERADORES DE COMPARAÇÃO

Os operadores de comparação são operadores que possuem o objetivo de analisar os valores de uma expressão e retornar um valor booleano, ou seja, verdadeiro ou falso.

```
1  # == IGUAL: Verifica se um valor é igual o outro.
2  print(5 == 5) # // True
3
4  # != DIFERENTE: Verifica se o valor é diferente do outro.
5  print(5 != 5) # // False
6
7  # > MAIOR QUE: Verifica se um valor é maior que o outro.
8  print(5 > 1) # // True
9
10 # < MENOR QUE: Verifica se o valor é menor que o outro.
11 print(5 < 1)# // False
12
13 # >= MAIOR IGUAL: Verifica se um valor é maior ou igual o outro.
14 print(5>=5) # // True
15
16 # <= MENOR IGUAL: Verifica se o valor é menor ou igual o outro.
17 print(5 <= 4) # // False
```

PY - Revisão Lógica

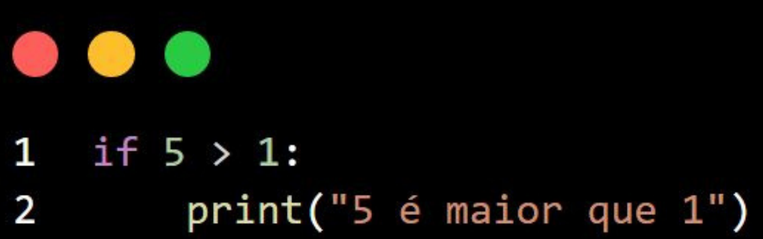
Condicionais - if

Uma estrutura condicional, como o próprio nome já diz é uma estrutura que vai analisar uma condição e baseado no resultado dessa condição é que vamos executar uma determinada ação.

A instrução IF, que, traduzindo para o português significa SE. Dessa forma, facilitamos o entendimento.

Um exemplo simples:
Se 5 for maior que 1, então, execute uma ação. Caso contrário não faça nada.

Então a instrução IF vai executar uma ação somente se a condição testada for verdadeira, nesse caso vai executar tudo que estiver dentro dela (com a indentação, espaçamento que temos abaixo.



```
1   if 5 > 1:
2       print("5 é maior que 1")
```

PY - Revisão Lógica

Estrutura condicional simples - if e else

```
1  if 5 > 1:
2      print("5 é maior que 1")
3  else:
4      print("5 não é maior que 1")
```

Aqui temos a instrução ELSE (que seria o senão). Primeiramente iremos testar a informação do IF e se ela não for verdadeira nós vamos para opção else.

Agora teremos 2 resultados para essa nossa comparação, ou seja, duas alternativas.

Funciona basicamente assim:

Se a primeira opção (condição) for verdadeira: Execute.

Senão, execute a segunda opção (condição).

PY - Revisão Lógica

```python
cor_semaforo = "amarelo"
if cor_semaforo == "verde":
    print("Acelerar!")
elif cor_semaforo == "amarelo":
    print("Atenção!")
else:
    print("Parar!")
```

Estrutura Condicional Aninhada
if elif else

Só que algumas vezes nós precisamos de mais de 2 resultados (condições), como um semáforo de trânsito.

Se a luz verde estiver acesa, podemos passar, luz amarela precisamos de atenção e no vermelho precisamos parar.

Ao invés de utilizar 3 vezes a instrução IF ou usar IF ELSE e depois outro IF, nós temos a instrução ELIF, que seria basicamente a junção de um ELSE + IF.

PY - Revisão Lógica

Operadores Lógicos

O operador lógico é quem indica ao sistema a maneira como se quer que uma premissa esteja em relação à outra dentro de uma estrutura condicional para que assim retorne verdadeiro ou falso.

Utilizamos esses operadores praticamente o tempo todo, principalmente para controle de fluxo de execução e tomadas de decisão.

Conheça agora alguns operadores lógicos.

and

p	q	p^q
V	V	V
V	F	F
F	V	F
F	F	F

or

p	q	p v q
V	V	V
V	F	V
F	V	V
F	F	F

not

p	~p
V	F
F	V

PY - Revisão Lógica

Execute esse código!

Um exemplo de como podemos aplicar um dos operadores lógicos é quando precisamos testar vários valores.

```python
n1 = 5
n2 = 3
n3 = 1
if n1 > n2 and n1 > n3:
    print("n1 é maior que todos!")
elif n2 > n1 and n2 > n3:
    print("n2 é maior que todos!")
else:
    print("n3 é maior que todos!")
```

Teste Tambem !

```python
# AND é verdadeiro somente se ambos os operadores forem verdadeiros.
print(5 > 3 and "cinco" == "cinco") # // True

# OR  o resultado será verdadeiro bastando apenas uma das comparações ser verdadeira.
print(5 == 5 or "cinco" != "5")# // True

# NOT é utilizado para inverter o resultado de uma determinada condição.
print(not 5 == 5) # // False
```

PY - Revisão Lógica

HORA DE PRATICAR!

1 - Faça um Programa que peça dois números ao usuário e imprima o maior deles.

2 - Faça um Programa que peça ao usuário um valor e mostre na tela se o valor é positivo ou negativo.

3 - Faça um Programa que imprima na tela "Femino", "Masculino" ou "Sexo inválido" a partir de uma letra digitada: F ou f para feminino, M ou m para masculino e qualquer outra letra para sexo inválido.

4 - Faça um Programa que verifique se a letra digitada pelo usuário é vogal ou consoante.

5 - Faça um programa para a leitura de quatro notas de um aluno. O programa deve calcular a média alcançada apresentar:

a. A mensagem "Aprovado", se a média alcançada for maior ou igual a sete;
b. A mensagem "Reprovado", se a média for menor do que sete;
c. A mensagem "Aprovado com Distinção", se a média for igual a dez.

6 - Faça um Programa que receba 2 números e em seguida pergunte ao usuário qual operação ele deseja realizar. O resultado da operação deve aparecer com uma frase que diga se o número é:

a. par ou ímpar;
b. positivo ou negativo;

PY - Revisão Lógica

ESTRUTURA DE REPETIÇÃO: FOR

Estrutura ou laços de repetição em Python é um recurso para desenvolver tarefas repetitivas em um loop contínuo. O loop funciona enquanto uma condição ser satisfeita.

É necessário organizar corretamente essa condição para não cometer erros em um programa. Se não for bem estruturado, o laço pode se tornar uma armadilha para o programador.

PY - Revisão Lógica

ESTRUTURA DE REPETIÇÃO: FOR

```
1  empresa = "INFINITY SCHOOL"
2  for letras in empresa:
3      print(letras)
```

O loop faz o que chamamos de iteração, ou seja, uma repetição que analisa, percorre ou repete finita ou infinitamente uma estrutura, coleção ou sequência de valores.

Tente também usando a função range()

```
1  for valores in range(0,5): #<- Sequencia
   de numeros de 0 a 5
2      print(valores)
```

A função range() nos permite especificar o início da sequência, o valor final, e o passo. O único parâmetro obrigatório é o que define quem será o último elemento da sequência.

PY - Revisão Lógica

VARIÁVEL FLAG

Variável que só é definida para setar um valor, geralmente através de atribuições utilizando operadores **+=**, **-=** ou ***=** ex:

```python
soma = 0 #<- variável flag
for x in range(0,3):
    nota = int(input("digite um valor: "))
    soma += nota #<- atribuindo valores
print(f'a média dos valores: {soma/3:.1f}')
```

PY - Revisão Lógica

LAÇO DE REPETIÇÃO: WHILE

O while é uma estrutura de repetição ou laço que é utilizada quando queremos que determinado bloco de código seja executado enquanto (while) determinada condição for atendida.

Em outras palavras: só vai sair da estrutura de repetição quando a condição não for mais satisfeita. Sua sintaxe básica:

```
1  cont = 0
2  while cont <= 5: #<- enquanto a condição for verdadeira.
3      print(cont)
4      cont += 1 #<- a variavel flag será incrementada.
```

PY - Revisão Lógica

LAÇO DE REPETIÇÃO: WHILE INFINITO

Um laço de repetição é considerado como infinito quando sua condição estabelecida nunca retorna como False. Eventualmente alguns programadores já passaram por esse tipo de problema em seus códigos, ao tentarem executar tal condição em sua estrutura.

Quando usar e como usar o Loop INFINITO?
Usamos o loop infinito quando não sabemos quantas vezes iremos rodar o programa, ou executar um comando, geralmente definimos uma condição para o usuário escolher quando parar.

```python
while True:
  palavra = input("digite qualquer palavra: ")
  if palavra == palavra.upper():
    print(palavra,"Maiusculo!")
  else:
    print(palavra,"Minusculo!")

  question = input("deseja continuar verificando? [Y/N]: ")
  if question == "n":
    break
```

TENTE EXECUTAR ESSE CÓDIGO!

AULA 01 | Python

PY - Revisão Lógica

BREAK + CONTINUE

Em Python, a instrução **break** força a interrupção do laço sempre que encontrada no bloco de código. A instrução **break** será colocada dentro do bloco de código, geralmente após uma instrução condicional if

```python
# Exemplo utilizando o for

for numero in range(10):
    if numero == 5:
        break
    print(numero)


# Exemplo utilizando o while

numero = 0

while numero < 10:
    if numero == 5:
        break
    print(numero)
    numero += 1
```

A instrução **continue** interrompe a execução do ciclo sem interromper a execução do laço de repetição. Ou seja, a iteração atual do loop será interrompida, mas o programa retornará ao topo do loop.

A instrução **continue** ficará dentro do bloco de código abaixo da instrução de loop, geralmente após uma instrução condicional if.

```python
cont = 0
while cont <= 5:
    if cont == 4:
        continue
    print(f'Numero {cont}')
    cont += 1
print("fim do teste.")
```

AULA 01 | Python

PY - Revisão Lógica

HORA DE PRATICAR !

1 - Escreva um programa que solicite ao usuário um número inteiro positivo e realize uma contagem regressiva a partir desse número até zero, imprimindo cada número no processo.

2 - Escreva um programa que solicite ao usuário um número inteiro e imprima a tabuada desse número, de 1 a 10.

3 - Escreva um programa que solicite ao usuário uma frase e conte quantas vogais (a, e, i, o, u) existem nessa frase.

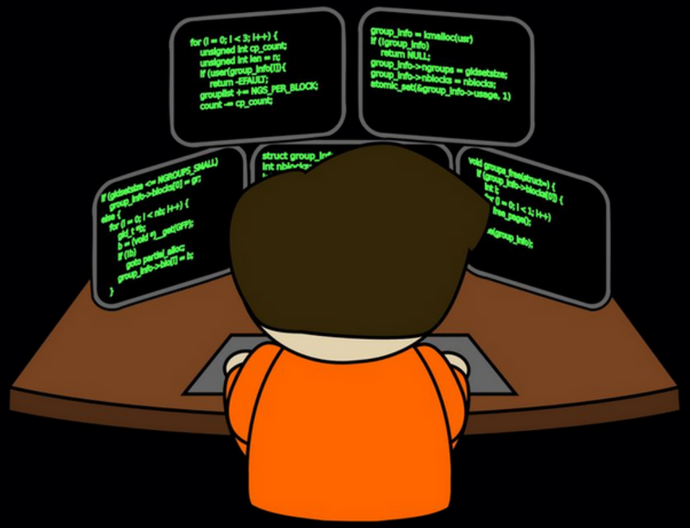
4 - Escreva um programa que possa verificar uma sequência de 10 números se são par ou ímpar.

5 - Hora de otimizar a questão número 3, após ter criado seu programa que conta quantas vogais existem numa frase, implemente mais uma funcionalidade. O programa agora deve imprimir a quantidade de vogais e consoantes encontradas.

PY - Revisão Lógica

HORA DE PRATICAR !

1 - Você é um operador de caixa em um supermercado e precisa implementar um programa para registrar as compras dos clientes. O programa deve solicitar o nome e o preço de cada produto e somar o total da compra no final. O operador pode inserir o nome e o preço de cada produto repetidamente até que ele deseje encerrar a compra, digitando um valor especial (por exemplo, -1). Utilize o laço de repetição **while** para permitir a inserção contínua dos preços dos produtos até que o operador decida encerrar a compra.



2 - Você foi contratado como desenvolvedor para criar um programa que verifique a senha digitada pelos usuários. O programa deve solicitar ao usuário que digite uma senha e verificar se ela atende aos critérios estabelecidos. Os critérios são os seguintes:

•A senha deve ter no mínimo 8 caracteres e no máximo 12 caracteres.

PY - Revisão Lógica

Conteudo Complementar !

PY - Revisão Lógica

NÚMERO PRIMO

Um número é dito primo quando é possível dividir ele (divisão de inteiro com inteiro) por 1 e por ele mesmo.

Exemplos de números primos:
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197...

Exemplo de código:

```python
n = int(input("Digite um número inteiro: "))
cont = 0
i = 0

while i <= n or cont < 2:
    i = i + 1
    x = n % i
    if x == 0:
        cont = cont + 1

if cont <= 2:
    print("primo")
else:
    print("não primo")
```

PY - Revisão Lógica

FATORIAL

O fatorial é muito usado em um ramo da Matemática chamado Análise Combinatória.

O fatorial (!) de um número n, representado por n!, é a multiplicação de n por seus antecessores maiores ou iguais a 1.

Essa operação é muito comum em análise combinatória.
Vejamos um exemplo de como calcular o fatorial de um número,
usando o laço for:

```
1    numero = int(input("Fatorial de: "))
2    resultado = 1
3    for n in range(1,numero+1):
4        resultado *= n
5        print(resultado)
```

PARABÉNS! VOCÊ TERMINOU A AULA 01 DO MÓDULO DE PYTHON