**09 - Instruções if-else, if, elif e else**

Nesse vídeo falaremos sobre as instruções **if-else**, e **if-elif-else**. Em nossos exemplos usaremos os mesmos dados do vídeo anterior, portanto pode ser possível executá-los novamente.

# 1º item da lista - Nome do veículo

# 2º item da lista - Ano de fabricação

# 3º item da lista - Veículo é zero km?

dados = [

['Jetta Variant', 2003, False],

['Passat', 1991, False],

['Crossfox', 1990, False],

['DS5', 2019, True],

['Aston Martin DB4', 2006, False],

['Palio Weekend', 2012, False],

['A5', 2019, True],

['Série 3 Cabrio', 2009, False],

['Dodge Jorney', 2019, False],

['Carens', 2011, False]COPIAR CÓDIGO

Estávamos construindo duas listas que separavam os dados entre veículos que são ou não são 0 km. Para isso, usando somente a condição if, precisávamos realizar a mesma operação duas vezes, ainda que realizando testes diferentes. Entretanto, com o if/else é possível realizarmos somente uma operação, seguindo o formato padrão abaixo:

if <condição>:

<instruções caso a condição seja verdadeira>

else:

<instruções caso a condição não seja verdadeira>COPIAR CÓDIGO

Para testarmos, começamos com a estrutura de condicional que utilizamos para criar a lista zero\_km\_Y.

zero\_km\_Y = []

for lista in dados:

if (lista[2] == True):

zero\_km\_Y.append(lista)

zero\_km\_YCOPIAR CÓDIGO

Aqui, quando a condição lista[2] é verdadeira (True), executamos o código que adiciona os itens à lista zero\_km\_Y. Para esse tipo de situação, na qual estamos testando uma variável booleana que só assume os valores True ou False, o uso do if/else se encaixa perfeitamente. Primeiro declararemos, na mesma linha de zero\_km\_Y, a variável zero\_km\_N (que também será uma lista do Python.

Manteremos o for varrendo todas as listas em dados, além da condição if preenchendo a lista zero\_km\_Y caso o retorno da verificação seja verdadeiro. Para incluirmos a cláusula contrária, adicionaremos um else seguido de dois pontos (:) no mesmo nível de indentação do if. Na linha seguinte, usaremos o método append() para adicionarmos os demais itens de lista à variável zero\_km\_N.

zero\_km\_N, zero\_km\_Y = [], []

for lista in dados:

if (lista[2] == True):

zero\_km\_Y.append(lista)

else:

zero\_km\_N.append(lista)COPIAR CÓDIGO

Com isso conseguiremos resolver o nosso problema utilizando uma única estrutura.

print(zero\_km\_Y)COPIAR CÓDIGO

[['DS5', 2019, True], ['A5', 2019, True]]

print(zero\_km\_N)COPIAR CÓDIGO

[['Jetta Variant', 2003, False], ['Passat', 1991, False], ['Crossfox', 1990, False], ['Aston Martin DB4', 2006, False], ['Palio Weekend', 2012, False], ['Série 3 Cabrio', 2009, False], ['Dodge Jorney', 2019, False], ['Carens', 2011, False], ['Jetta Variant', 2003, False], ['Passat', 1991, False], ['Crossfox', 1990, False], ['Aston Martin DB4', 2006, False], ['Palio Weekend', 2012, False], ['Série 3 Cabrio', 2009, False], ['Dodge Jorney', 2019, False], ['Carens', 2011, False], ['Jetta Variant', 2003, False], ['Passat', 1991, False], ['Crossfox', 1990, False], ['Aston Martin DB4', 2006, False], ['Palio Weekend', 2012, False], ['Série 3 Cabrio', 2009, False], ['Dodge Jorney', 2019, False], ['Carens', 2011, False], ['Jetta Variant', 2003, False], ['Passat', 1991, False], ['Crossfox', 1990, False], ['Aston Martin DB4', 2006, False], ['Palio Weekend', 2012, False], ['Série 3 Cabrio', 2009, False], ['Dodge Jorney', 2019, False], ['Carens', 2011, False]]

Agora partiremos para o **elif**, que basicamente consiste na junção de um else e um if. A ideia é que testamos uma primeira condição com o if e executamos determinada instrução caso ela seja verdadeira. Caso contrário, se tivermos outra condição para testar, usamos o elif. Se a condição for verdadeira, executamos um trecho de código; do contrário, podemos partir para outra condição com o elif, e assim sucessivamente. Por fim, encerramos a sequência com um else que executará outro código caso nenhuma das condições anteriores sejam verdadeiras.

if <condição 1>:

<instruções caso a condição 1 seja verdadeira>

elif <condição 2>:

<instruções caso a condição 2 seja verdadeira>

elif <condição 3>:

<instruções caso a condição 3 seja verdadeira>

.

.

.

else:

<instruções caso as condições anteriores não sejam verdadeiras>COPIAR CÓDIGO

Na verdade o else não é estritamente necessário, sendo possível terminar as instruções com o próprio elif. Em nosso próximo exemplo, queremos separar os veículos do conjunto dados de acordo com o ano em que foram fabricados, agrupando-os em três grupos diferentes. Essa informação pode ser encontrada no índice 1 das nossas listas.

Antes de fazermos isso, vamos explorar um pouco os códigos preparados no notebook a respeito de dois operadores lógicos: **and** e **or**.

Quando testamos duas condições e usamos o operador and, se uma delas é verdadeira e a outra também é, o resultado é verdadeiro; se uma é verdadeira e a outra é falsa, o resultado e´falso; e se ambas forem falsas, o resultado também é falso.

print('AND')

print(f'(True and True) o resultado é: {True and True}')

print(f'(True and False) o resultado é: {True and False}')

print(f'(False and True) o resultado é: {False and True}')

print(f'(False and False) o resultado é: {False and False}')COPIAR CÓDIGO

AND (True and True) o resultado é: True (True and False) o resultado é: False (False and True) o resultado é: False (False and False) o resultado é: False

No caso do or, essa lógica é um pouco diferente. Se ambas as condições forem verdadeiras, o resultado é verdadeiro; se uma for verdadeira e outra não, o resultado também é verdadeiro; e a única situação em que o resultado é falso é quando as duas condições forem falsas.

print('OR')

print(f'(True or True) o resultado é: {True or True}')

print(f'(True or False) o resultado é: {True or False}')

print(f'(False or True) o resultado é: {False or True}')

print(f'(False or False) o resultado é: {False or False}')COPIAR CÓDIGO

OR (True or True) o resultado é: True (True or False) o resultado é: True (False or True) o resultado é: True (False or False) o resultado é: False

Para o próximo teste, criaremos as listas vazias A, B e C. Em seguida, usaremos o for para iterar por dados chamando cada elemento encontrado de lista. Incluiremos então a primeira verificação: se lista[1] for menor ou igual a 2000 (ou seja, se o veículo foi fabricado antes do ano 2000 ou neste ano), adicionaremos o elemento em A usando o método append().

A, B, C = [], [], []

for lista in dados:

if lista[1] <= 2000:

A.append(lista)COPIAR CÓDIGO

Com o elif, criaremos uma segunda condição, dessa vez verificando se lista[1] é maior que 2000 (ou seja, se o veículo foi fabricado após o ano 2000) e (and) menor ou igual a 2010 (fabricação antes de 2010). Em caso positivo, adicionaremos o elemento na lista B usando o append().

A, B, C = [], [], []

for lista in dados:

if lista[1] <= 2000:

A.append(lista)

elif lista[1] >= 2000 and lista[1] <= 2010:

B.append(lista)COPIAR CÓDIGO

Por fim, adicionaremos uma última condição com o else: caso nenhuma das condições anteriores sejam preenchidas, adicionaremos o carro à lista C.

A, B, C = [], [], []

for lista in dados:

if lista[1] <= 2000:

A.append(lista)

elif lista[1] >= 2000 and lista[1] <= 2010:

B.append(lista)

else:

C.append(lista)COPIAR CÓDIGO

Após executarmos esse código, verificaremos o conteúdo das listas:

ACOPIAR CÓDIGO

[['Passat', 1991, False], ['Crossfox', 1990, False]]

BCOPIAR CÓDIGO

[['Jetta Variant', 2003, False], ['Aston Martin DB4', 2006, False], ['Série 3 Cabrio', 2009, False]]

CCOPIAR CÓDIGO

[['DS5', 2019, True], ['Palio Weekend', 2012, False], ['A5', 2019, True], ['Dodge Jorney', 2019, False], ['Carens', 2011, False]]

Perceba que A só possui veículos cuja fabricação é anterior ou ano 2000; B compreende veículos com fabricação entre 2001 e 2010; e C contém somente veículos fabricados após 2010.

O Python nos permite utilizar uma notação bastante conhecida em matemática, na qual colocamos o elemento comparado entre dois valores. No nosso exemplo, lista[1] deve estar entre 2000 e 2010:

A, B, C = [], [], []

for lista in dados:

if lista[1] <= 2000:

A.append(lista)

elif 2000 < lista[1] <= 2010:

B.append(lista)

else:

C.append(lista)COPIAR CÓDIGO

Esse código funcionará da mesma forma que o anterior, Lembre-se que essa notação não é válida na maioria das linguagens de programação, mas é bem aceita no Python e pode ser encontrada em pacotes voltados à estatística ou matemática.

Encerramos então o conteúdo básico de Python apresentado neste treinamento. Mas fique tranquilo(a), pois iremos nos aprofundar ainda mais na linguagem durante a continuação, "Python para Data Science: Funções, Pacotes e Pandas básico". Na próxima seção, que encerra a primeira parte do curso, conheceremos a biblioteca **Numpy**.