03

Fatiamento e índices de strings

Transcrição

[00:00] Fala, pessoal. Bem-vindos a mais esse vídeo. Nele, quero começar com vocês do zero um projeto do PyCharm.

[00:08] Clicando no programa, ele já renderizou para mim uma janela com opções de criar projeto, abrir um projeto. Vou escolher criar um novo. O nome do meu projeto vai ser “Python”. Simples e direto.

[00:28] Ele me dá já algumas dicas. Vocês podem ver se quiserem. Tenho do lado esquerdo a pasta que simboliza o meu projeto. Posso clicar com o botão direito nela e escolher criar um arquivo novo. O nome vai ser “main”. Posso também minimizar ou maximizar a aba projeto.

[00:59] Agora, quero começar de onde paramos na aula passada. Lembram que tínhamos uma variável que era sobreMim = ‘Meu nome é Rodrigo e minha idade é 26”. O desafio era conseguir retirar somente o 26 de dentro dessa string, para dizer somente minha idade.

[01:27] Antes disso, vou começar por uma variável mais simples: meuNome = ‘Rodrigo’. Quero dizer para vocês que até certo ponto uma string do Python é muito semelhante a uma lista, porque cada caractere dentro da string possui um índice que posso acessar e mostrar somente esse índice específico.

[01:51] Na linha de baixo, vou colocar: #.....0123456

[01:57] Fiz isso para a linha começar no R de Rodrigo, que tem índice 0, 1, 2, e assim por diante até chegar ao final do nome, com o O no final que tem índice 6.

[02:08] Eu consigo criar uma variável subString= meuNome para acessar um argumento específico. Mas tem uma notação certa para poder acessar esse argumento. Preciso adicionar um colchete: subString= meuNome [ ]

[02:30] Se eu quiser pegar o I de Rodrigo, coloco dentro desse colchete o valor 4, que é o índice desse caractere. Dando um print, o Python nos mostra no console o I, que é representado pelo índice quatro.

[03:02] Mas consigo fazer muito mais coisas. Se eu colocar dentro do colchete dois pontos, consigo começar em algum índice específico e ir até um índice específico, fazendo o fatiamento da string. Por exemplo, se eu quiser pegar o drig, coloco: subString= meuNome [2:5]

[03:36] Rodando isso, ele me dá um dri. Algo deu errado. Eu coloquei os índices corretos, mas não veio tudo que eu queria, porque o índice que inicia o fatiamento é inclusivo, mas o índice que finaliza é exclusivo. Preciso colocar um índice maior do que ele para poder retornar o que eu quero: subString= meuNome [2:6]

[04:02] Tem mais dois detalhes interessantes para vermos. Por exemplo, se eu não colocar o primeiro índice, ele vai pegar do começo, do índice 0, até o índice que indiquei depois. O que faz sentido. Se eu não coloquei nada, o Python entende que quero começar do início da minha string.

[04:23] O contrário é válido também. Se eu quiser colocar o primeiro índice e deixar o segundo índice vazio, o Python sabe que eu quero ir até o final da minha string.

[04:38] Lembram do nosso desafio? O desafio maior agora é conseguir contar quantos índices existem até chegar no 26. Eu vou chutar, colocando: subString= sobreMim [27:]. Ele me mostrou a partir de “idade”. Depois, eu conto quantos faltam até o 26. Somando, dá 35. Se eu colocar agora: subString= sobreMim [35:], ele me dá o 26.

[05:26] Esse sobreMim é uma string interessante, importante, mas existe uma string mais importante ainda que usamos o tempo todo e não damos muita atenção. É a URL de um site. Aquele endereço que fica na barra do navegador que você usa.

[05:50] Acessando o YouTube, por exemplo, vamos supor que eu quero ouvir Wesley Safadão. No momento em que eu der um enter na busca, a URL muda. Depois do endereço principal do YouTube e de "results", tenho uma interrogação, um caractere especial da URL. Ela indica que estou começando a passar alguma informação. Mas passar o quê e para onde?

[06:34] Tenho que passar um argumento, search_query, que é minha pesquisa, para o back end do YouTube, para a aplicação YouTube, e aí sim ele vai conseguir acessar o banco de dados e renderizar para nós o que queremos.

[06:51] Será que só o YouTube faz isso? Outra coisa mais simples ainda que fazemos no dia a dia é uma pesquisa no Google. Vou pesquisar por Alura. Percebam como novamente a URL mudou. E a interrogação está lá, indicando que estou passando alguma informação para o Google. Temos o q, de query, provavelmente, que separa o nome do valor, que foi Alura.

[07:25] Aqui tenho mais coisas ainda. Tenho o &, que indica que estou passando mais de um argumento para o back end do Google, para ele poder resolver o que ele tem que resolver e fazer as buscas necessárias. Faz sentido, porque talvez o método de uma classe nossa precise de mais de uma coisa para funcionar, mais de um argumento. É aí que esse cara entra em ação, indicando que tem vários argumentos sendo passados.

[07:59] Vamos voltar para o PyCharm. Quero apresentar para vocês um banco chamada ByteBank. Ele não existe, mas já possui uma aplicação web, que faz transformações cambiais. A URL dele completa é a seguinte: url = "https://www.bytebank.com.br/cambio?" (https://www.bytebank.com.br/cambio?%E2%80%9D)

[08:39] Aí começo a passar os argumentos para esse endereço do ByteBank. Então, preciso de uma interrogação. Mas antes, quantos argumentos preciso para conseguir fazer uma transformação cambial?

[08:46] Supondo que eu queira transformar 1,500 de real para dólar. Três argumentos. Preciso do valor, da moeda origem e da moeda destino. Vamos colocar isso na URL: url = "https://www.bytebank.com.br/cambio?moedaorigem=real&moedadestino=dolar&valor=1500" (https://www.bytebank.com.br/cambio?moedaorigem=real&moedadestino=dolar&valor=1500%E2%80%9D)

[09:14] Tenho aqui a URL completa que preciso para fazer essa transformação cambial. Antes disso, vamos dar um zoom em um desses argumentos e vamos fazer um fatiamento com esse argumento usando o que já vimos de string.

[09:34] Ao invés de URL, vou colocar argumento. Vamos dar uma olhada no argumento "moeda origem": argumento = "moedaorigem=real". Supondo que eu já queira pegar o nome desse argumento usando fatiamento, vou comentar: #.....012345678910

[09:59] Depois: subString = argumento [:11]. Preciso colocar até o índice 11 para conseguir capturar até o M. Dando um print na subString, ele me dá o nome desse argumento. Mas e se eu quisesse pegar só origem? Eu teria que colocar o índice do O, que é 5: subString = argumento [5:11]

[10:42] Agora que já vimos que conseguimos retirar o nome desse argumento, vamos supor que quero colocar o endereço completo do ByteBank: argumento = "https://www.bytebank.com.br/cambio?moedaorigem=real&moedadestino=dolar&valva=1500" (https://www.bytebank.com.br/cambio?moedaorigem=real&moedadestino=dolar&valva=1500%E2%80%9D)

[11:05] E se eu rodar o código agora? Apareceu algo sem significado nenhum. O que aconteceu foi que eu coloquei um prefixo na minha string e isso acabou com o meu código. O código que acabamos de fazer para retirar um pedaço do argumento, um nome do argumento, estaria completamente destruído. Eu não conseguiria usar mais ele da forma como meu código está.

[11:32] Será que sempre que eu mudar alguma coisa na minha URL preciso abrir o código fonte do ByteBank e modificar o código fonte? Ou será que existe uma forma mais simples e dinâmica de resolver esse problema? Aula que vem veremos isso.