CURSO DE PYTHON3 MUNDO 03

O curso de Python3, é oferecido pelo **Curso em vídeo**, e tem como orientador o professor Gustavo Guanabara.

🌍 O Mundo 03 do curso de Pyhton3 segue sequencialmente os Mundos 01 e 02. Por isso as aulas e exercícios continuam na sequência.

AULA 16 - Tuplas

📘Nessa aula:

- O professor Gustavo Guanabara iniciou o assunto de variáveis compostas:

▼ Falou sobre um dos tipos de variável composta, a tupla;

A tupla é uma estrutura que permite uma variável receber mais de um objeto;

Tuplas são entre parênteses ( ).

- Mostrou que tuplas são imutáveis uma vez que sejam definidas;

▼ Guanabara ensinou alguns comandos para efetuar certas funções na hora de criar o código;

count - Serve para saber quantas vezes algo aparece em uma string;

index - Serve para saber a posição de algo em uma string;

del - Serve para deletar uma variável;

sorted - Serve paraordenar a tupla.

- Os exercícios de número 072 a 077 foram disponibilizados.

AULA 17 - Listas (Parte 1)

📋Nessa aula:

▼ O professor Gustavo Guanabara apresentou outra variável composta, as listas;

Diferentemente das tuplas que são imutáveis, as listas podem ser modificadas;

As listas são entre colchetes [ ];

▼ Mostrou comandos que podem facilitar nossa manejo com as listas;

append - É uma função que serve para adicionar um elemento a mais no **final** da lista;

insert - Para adicionar elementos em outras posições da lista;

del - Também falado na aula de tuplas, serve para apagar um item dentro da lista;

pop - Efetua a mesma função do del;

remove - Serve para eliminar um item da lista através do conteúdo, como o nome por exemplo. Se não for especificado, o **último** item da lista é eliminado. Caso ocorra de ter dois valores iguais na lista, o remove eliminará o que aparecer primeiro.

▼ Guanabara explicou que podemos criar uma lista através do range;

Ex: valores = list(range(4,11))

▼ Mostrou como organizar uma lista, utilizando o método sort;

Ex: valores.sort ( )

De maneira inversa usamos;

Ex: valores.sort(reverse = True)

- Os exercícios de número 078 a 083 foram disponibilizados.

AULA 18 - Listas (Parte 2)

📋Nessa aula:

- Guanabara deu seguimento ao assunto de listas, iniciado na aula anterior;
- Fez um breve resumo sobre o que foi explicado;

- Ensinou a juntar estruturas;

▼ Mostrou o que são listas **compostas**;

Ex:

NOMES	‘PEDRO’ 25	‘MARIA’ 19	‘JOÃO’ 32
ÍNDICES	0	1	2

pessoas = [[’Pedro’, 25] , [’Maria’, 19] , [’João’, 32]]

Os colchetes vermelhos representam uma lista principal, e cada colchete azul representa uma lista diferente, dentro da lista principal.

▼ O professor também ensinou a selecionar um item específico dentro de uma lista composta;

Ex: print(pessoas [0] [0]) Isso resultará em ‘Pedro’ que é o índice 0 que está dentro de outro índice 0;

print(pessoas [0] [1]) Isso resultará em ‘25’ que é o índice 1 que está dentro do índice 0.

- Ensinou a copiar listas;
- Os exercícios de número 084 a 089 foram disponibilizados.

AULA 19 - Dicionários

📚Nessa aula:

- Relembrou o que é uma variável composta;

▼ O professor Gustavo Guanabara deu sequência nas variáveis compostas, dessa vez nos apresentou os dicionários;

- Os dicionários são estruturas parecidas com tuplas e listas, porém, diferentemente dessas duas, os dicionários apresentam a possibilidade de termos índices **literais**;

- Dicionários são entre chaves { }

▼ Para declarar um dicionário pode se usar apenas chaves ou o comando dict;

Ex: dicionário = dict( ) ou dicionário = { }

▼ Os dicionários trazem a possibilidade de substituirmos os índices **numéricos** por índices **literais**;

Ex:

Listas ou Tuplas:

' PEDRO '	25
0	1

Dicionários:

' PEDRO '	25
Nome	Idade

- Perceba que nos dicionários, os índices **0** e **1**, foram substituídos por **Nome** e **Idade.**

- O mesmo exemplo, só que agora representado em uma string:

dados = {'nome': 'Pedro' , 'idade': 25}

print (dados["nome"]) = Pedro

print (dados["idade"]) = 25

▼ Para adicionar outro item dentro de um dicionário, basta escrever o nome do dicionário e o que deseja adicionar nele;

Ex:

Utilizando a variável sexo como exemplo;

dados ['sexo'] = 'M'

'PEDRO'	25	'M'
Nome	Idade	Sexo

- Note que a variável sexo, foi adicionada ao dicionário dados.

▼ Para deletar um elemento de um dicionário, basta usar o comando del, assim como nas listas;

Ex: del dados ['idade']

▼ Guanabara mostrou a diferença entre valores, chaves e itens, em dicionários;

Ex:

1. print (dados.values( )) Ele pegará apenas os valores que são nesse caso 'Pedro' , '25' e 'M;
2. print (dados.keys( )) Ele pegará apenas as chaves que nesse caso são 'Nome', 'Idade' e 'Sexo'
3. print (dados.items( )) Ele pegará os itens, que basicamente é a junção de chaves e valores. Ou seja, ele pega tudo que estiver dentro de um dicionário.

- Ensinou a copiar um elemento utilizando o método copy, que só funciona em **dicionários;**
- Mostrou a função sum, para realizar operações de soma de maneira mais prática dependo da ocasião;
- Nos mostrou que é possível conter dicionários dentro de listas;
- Os exercícios de número 090 a 095 foram disponibilizados;
- Na resolução do exercício 091, Guanabara apresentou o itemgetter, importado através da biblioteca operator e serve para ordernar listas **de dicionários.**

AULA 20 - Funções (parte 1)

💿Nessa aula:

- O professor Gustavo Guanabara explicou o que são funções;
- Mostrou como criá-las;

▼ Nos apresentou o def;

Ele serve para declararmos uma função **personalizada.**

▼ Guanabara nos ensinou a criar uma rotina, para evitar várias repetições de códigos;

Ex: def mostrarLinha ( )

print('- - - - - - ') ou print ('-' * 30) O **número 30** foi usado apenas como exemplo, mas pode ser o valor que desejar

Basicamente ele criará uma função que serve para escrever essa linha. E ao invés de escrevermos vários prints, basta escrever mostrarLinha.

▼ Mostrou que podemos trabalhar com parâmetros;

Ex: Supondo que desejamos mudar somente uma mensagem dentro de linhas;

ANTES	DEPOIS
print(' - - - - - ')	def mensagem (msg):
print(Alguma mensagem)	print(' - - - - -')
print('- - - - - ')	print(msg)
	print(' - - - - -')

Depois a mensagem escolhida deve ser declarada. Ex: mensagem('Curso de Python').

Logo, onde estava escrito 'msg' será substituído pela mensagem declarada. Que nesse caso é 'Curso de python'

▼ Guanabara ensinou a como desempacotar parâmetros;

Ex: Com o def sendo contador:

```
def contador ( * num )
```

O **asterisco** nesse caso significa desempacotar. Então todo parâmetro que seja escrito em 'contador' **nessa ocasião**, será jogado dentro de 'num'.

▼ Ensinou a usarmos listas em funções;

Na lista não necessitamos do asterisco.

Ex: def nome (lista): ← Não teve o uso do asterisco

- Os exercícios de número 096 a 100 foram disponibilizados;
- Na resolução do exercício 098, Guanabara nos mostrou a função flush.

AULA 21 - Funções (parte 2)

💿Nessa aula:

- O professor Gustavo Guanabara deu continuação ao conteúdo de funções;

▼ Falou sobre Interactive Help, que significa **ajuda interativa**, e pode ser acessada através da função help ( );

Basicamente ela vai ajudar caso alguma dificuldade ou dúvida venha a surgir. Assim, mostrando como se fosse um manual de uma função desejada.

Ex: Com help(print);

```
print(*args, sep=' ', end='\n', file=None, flush=False)
    Prints the values to a stream, or to sys.stdout by defa

    sep
      string inserted between values, default a space.
    end
      string appended after the last value, default a newl
    file
      a file-like object (stream); defaults to the current
    flush
      whether to forcibly flush the stream.
```

Veja que o próprio sistema ofereceu informações necessárias sobre o print.

▼ Além disso, Guanabara também mostrou o doc, para saber os parâmetros de um comando, assim como o help. Porém de uma maneira alternativa;

Ex: Com input

Utilizando o comando print(input.__doc__)

```
Read a string from standard input.  The trailing newline i

The prompt string, if given, is printed to standard output
trailing newline before reading input.

If the user hits EOF (*nix: Ctrl-D, Windows: Ctrl-Z+Return
```

Veja que o próprio sistema ofereceu informações necessárias sobre o input.

- Nos mostrou o que é uma Docstring, que significa **string de documentação**, e basicamente é o que foi mostrado pela função help e pelo doc;

▼ Ensinou a criarmos uma docstring própria;

Ex: def sendo contador

Para isso, perceba que 3 aspas duplas foram abertas e entre elas, foi colocada a informação desejada.

```
def contador(i, f, p):
    """
    -> Faz uma contagem e mostra na tela
    i = Inicio da contagem
    f = Fim da contagem
    p = Passo da contagem
    Funcao criada por Gustavo Guanabra drurtante a aula
    """
```

Nesse caso, quando o usuário, escrever a função help, para o contador nesse caso, irá aparecer para ele a informação escrita acima.

▼ Gustavo mostrou Parâmetros Opcionais;

Ex:

Supondo que desejamos realizar uma soma entre a, b e c utilizando def somar (a,b, c). Porém suponhamos que o c não foi declarado. Para que não dê erro futuramente no seu código use def somar (a, b, c = 0). Dessa forma o c será um parâmetro opcional, e se não for declarado pelo usuário, ele valerá 0. Lembrando que iss opode ser usado em todos os parâmetros, assim: def somar (a = 0, b = 0, c = 0).

▼ Apresentou Escopo de Variáveis;

Na programação, o escopo é o local onde a variável vai existir ou não.

▼ Mostrou o que são Variáveis locais e globais;

Variável local é aquela que serve somente dentro do “corpo” da função;

Variável global é aquela que pode ser usada tanto dentro quanto fora do "corpo" da função.

Ex:

```
#Dentro
def funcao():
    n1 = 4
    print(f'N1 dentro vale {n1}')

#Fora
n1 = 2
funcao()
print(f'N1 fora vale {n1}')
```

Perceba que nesse caso n1 dentro do def vale 4, assim sendo uma variável local. E fora vale 2, assim sendo uma variável global.

Para declarar que uma variável é **global** basta usar a função global e em seguida a variável.

▼ Guanabara nos apresentou o Retorno de Valores;

Ensinou a usarmos o return, que serve para enviar um valor de volta ao ponto onde ele foi chamado.

- Os exercícios de número 101 a 106 foram disponibilizados;
- Na resolução do exercício 101 Guanabara falou sobre Escopo de Importação;
- Na resolução do exercício 103 o professor nos mostrou o isnumeric. Ele verifica se um texto é todo feito por números.

AULA 22 - Módulos e Pacotes

📦Nessa aula:

▼ O professor Gustavo Guanabara explicou o que é modularização e suas vantagens;

Algumas vantagens

1. Organização do código;
2. Aumenta a legibilidade;
3. Facilita a manutenção do código;
4. Pode ocultar o código detalhado, assim, o deixando mais organizado;
5. Reutilização em outros projetos.

- Ensinou a criarmos os nossos próprios módulos e importa-los;

▼ Explicou o que é um pacote em Python;

Basicamente é uma pasta que contém módulos. Nesses pacotes pode se conter vários módulos separados e organizados por assuntos.

Ex: Utilizando um pacote chamado uteis

Números	Datas	Cores	Strings
def abc ( )	def xvi ( )	def cfr ( )	def bkg ( )
def efg ( )	def hmn ( )	def lkj ( )	def hpo ( )
def wkm ( )	def crt ( )	def gbh ( )	def srt ( )

Perceba que dentro de um pacote temos vários módulos (Números, Datas, Cores, Strings).

É possível importar o pacote inteiro utilizando o comando import uteis, ou especificar o que deseja importar por exemplo from uteis import datas. Dessa forma só o módulo Datas será importado.

- Guanabara ensinou a criarmos nosso próprios pacotes;
- Os exercícios de número 107 a 112 foram disponibilizados.

AULA 23 - Tratamento de Erros e Exceções

❗ Nessa aula:

- O professor Gustavo nos ensinou a como **tentar** solucionar erros no nosso programa;

▼ Explicou o que é uma exceção em Python;

Ex:

1. NameError → name 'tabela' is not defined significa que a variável tabela não está definida.
2. ValueError → é uma exceção comum em Python que ocorre quando uma função recebe um argumento com o tipo correto, mas com um valor inválido.
3. TypeError → representa um erro de quando um valor não é do tipo esperado.

   Esses são apenas alguns exemplos. Existem **vários** outros.

   Toda exceção em Python vem de uma classe maior chamada Exception

▼ Ensinou a tratar uma exceção com o comando try.

O try é utilizado para envolver código que pode gerar exceções;

Também apresentou o except que dirá o que irá acontecer caso o try venha a **falhar**;

Além desse, temos o finally, que irá acontecer independentemente de ocorrer uma falha ou não;

E por último temos o else, já apresentado anteriormente no curso.

- Nos ensinou a apresentar qual foi o erro que veio a ocorrer;
- Os exercícios de número 113 a 115 foram disponibilizados. Lembrando que o exercício 115 é dividido em 3 partes. Sendo elas 115a, 115b e 115c.

🌍 Mundo 03 finalizado.
