atalhos:

* Alt shift down copia a linha para baixo
* no Visual Studio Code: Marque as linhas que você quer comentar e depois tecle: Comentar: CTRL + K, CTRL + C. Descomentar: CTRL + K, CTRL + U
* Clicar em um elemento, segurar alt e clicar em outro altera os dois
* Selecionar um elemento e apertar F2 renomeia todas as ocorrências do mesmo elemento

Instalar no VS:

Python

Code Runner

Dracula

Comentários: # torna a linha em comentário

"""

Desta forma, cria bloco de comentários - DocString

em diversas linhas

"""

print - vírgula separa as funções

,sep="" ou ,sep='' , dentro do print, indica qual o separador será usado.

Exemplo:

print(12, 34, 56) - saída: 12 34 56

print(12, 34, 56, sep='pexe') - saída: 12pexe34pexe56

print(12, 34, 56, sep=", ") - saída: 12, 34, 56

Um print abaixo do outro imprime em linhas diferentes (por padrão)

Para alterar essa propriedade, colocar , end=”” ou ,end=’’ ao final

print(12, 34, 56, sep="abc\n\n\t")

print(12, 34, 56)

saída:

12abc

34abc

56

12 34 56

Tipos de variáveis

Tipagem estática – necessário sempre informar o tipo de variável.

Tipagem dinâmica (sabe qual o tipo de variável) forte

Por ser fortemente tipada, necessita da type conversion implementada.

- String (int)– textos entre “” ou ‘’ (aspas simples ou duplas).

Metacaracter – utiliza a \ antes.

Caracter de escape é o caracter que encerra o texto ou comando.

Exemplo: print("Luiz \" Otavio")

Saída: Luiz “ Otavio

Normalmente, para evitar isso, utiliza-se aspas duplas ou simples conforme necessidade. Se possui “ no texto, utiliza-se ‘ para iniciar a informação de texto e vice-versa.

a função type pode mostrar o tipo da variável

-int e float igual java

- booleano (bool) – True ou False

- Type conversion – muda os tipos de dados.

Tipos de dados imutáveis e primitivos: str, int, float e bool -> Int pode ser somado com float

Lança erros ao tentar juntar tipos diferentes. Exemplo - print(1+"1") – lança erro de type conversion

Para converter:

int(variável); float(valor); str(valor); bool(valor)

print(int(‘7’)+7)

print(float(7)+5)

Somente será feita a conversão se for possível.

De acordo com a PEP8 – padronização:

Variáveis – devem iniciar com letras minúsculas e podem possuir números ou underline.

Sinal de = é operador de atribuição de valor para a variável.

var1 = 'teste'

var2 = 2

print(var1, var2)

Expressoes aritméticas

Divisão inteira 10//2.5

Exponenciação - 2\*\*5 (representa 2^5)

Modulo 55 % 2 – mostra o resto da divisão

- Concatenação – pode ser feito em strings

‘f string – adiciona-se o f à frente das strings para indicar que pode receber variáveis e oque se deseja substituir coloca-se entre colchetes. Para definir a qtd de casas, usa-se :.2f após o nome da variável.

Exemplo: altura = f’{altura:.5f} metros’

Formatação de textos:

#Formatação tipo 1:

texto = f'{nome} tem {peso:.1f} kg e {altura:.2f} m'

print(texto + "\n")

print(f'SEM VARIÁVEL - {nome} tem {peso :.1f} kg e {altura :.2f} m. \nPortanto, seu imc é {imc :.4f}')

#Formatação tipo 2:

#Outra opção de formatação: - adicionar o format após a variável

a = 'AAA'

b = 'BB'

c = 5

var1 = ''.format(a, b, c)

print (var1)

#cada chave referencia um valor do format

var1 = 'a={} b={} c={:.3f}'.format(a, b, c)

print (var1)

#O código abaixo resultará em erro, devido ao excesso de variáveis.

# var1 = 'a={} b={} c={:.3f} {}'.format(a, b, c)

# print (var1)

#FORMATAÇÃO TIPO 3

var3 = 'a={1} b={0} c={teste:.5f} IMC = {1}'

formatoVar3 = var3.format(

    b, a, *teste* = imc

)

print(formatoVar3)

#Após alguma variável receber um parâmetro, todas as outras também deverão receber As anteriores a ela não precisam

#FORMATAÇÃO TIPO 4

var4 = 'a={1} b={0} c={2} IMC = {1}'

formatoVar4 = var4.format(

    b, a, imc

)

print(formatoVar3)

Inserir inputs

input('Qual seu nome? ')

Pergunta ao usuário qual o nome e aguarda a resposta – não continua sem o usuário executar ação.

nome = input('Qual seu nome? ')

print(f'O seu nome é {nome}')

print(f'O seu nome é {nome=}')

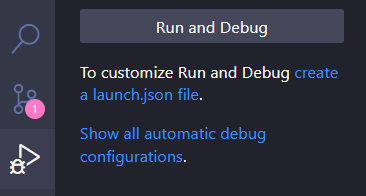
O segundo caso recebe o valor recebido, retornando inclusive as aspas.

Saída:

O seu nome é Lucas  
O seu nome é ‘Lucas’

-------- Iniciando o Debugger para python:

Run and Debug -> Create a launch.json file -> Python Debugger (na parte superior) -> Python file



Marcar a linha que deseja que seja pausada a execução. – breakpoint – pausa antes de executar a linha marcada com o breakpoint.

Caso o ponto vermelho do breakpoint não estiver aparecendo, pesquisar nas settings glyph Margin

Quanto maior a complexidade, mais difícil de manutenção é o código.

Complexidade: Em python pode ser dita como a quantidade de afastamentos da linha lateral inicial esquerda.

#DOCUMENTAÇÃO OFICIAL DO PYTHON: https://docs.python.org/pt-br/3/library/stdtypes.html