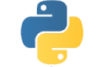
23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook 

**UFCD 10793 - Fundamentos de Python** 

**Notebook 02 - Operações básicas e formatos de dados em Python**

**Sílvia Martins**

**AEFHP 2022**

**Formatar a saída com *format***

Pode-se formatar o resultado de um comando print de formas mais sofisticadas, usando o método format.

In [ ]:

"conseguir fazer {} dos {} exercícios do teste".format(3,5)

Ou aplicar o método format de outra forma:

In [ ]:

n1, n2**=** 3, 5 *#atribuição múltipla*

print(f"conseguir fazer {n1} dos {n2} exercícios do teste.")

**Efetuar alinhamentos na saída**

**Exemplos:**

In [ ]:

*#usando o métodos da classe string ( ver abaixo)*

msg**=**"Olá turma PI6!"

print(f"{msg}".rjust(30)) *# alinha à direita num campo de edição de 30 carateres* print(f"{msg}".center(30)) *# alinha ao centro ...*

print(f"{msg}".ljust(30)) *# alinha à esquerda...*

127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 1/13

23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook

In [ ]:

*#usando formatações da classe format*

msg1, msg2, msg3**=**"Ana" , "Rui", "Maria"

loc1, loc2, loc3**=**"Lisboa", "Porto", "Faro"

print(f"{msg1:**<**5}{loc1}") *# alinha à esquerda msg1 num campo de edição de 15 char seguido* print(f"{msg2:**^**5}{loc2:**>**5}") *# alinha ao centro msg2 num campo de edição de 5 char e loc2* print(f"{msg3:**>**10}{loc3:**<**5}") *# alinha à direita msg3 num campo de edição de 10 char e lo*

**Efetuar formatos de valores numéricos**

**Exemplos:**

In [ ]:

x, y, z**=**4**\***(12**-**4), 25**/**9 ,7**//**6 *#resto da divisão inteira*

print(f"{x:3d}") *# 3 casas reservadas no total*

print(f"{y:4.2f} €") *# 4 casas reservadas no total, sendo 2 para a parte decimal* print(f"{y:3.0f}")

print(f"{z:3d}")

print(f"{y:4.2f} €")

**Funções matemáticas**

**Exemplos:**

In [ ]:

**import** math

E1, E2**=**math.pow(2,3) , math.sqrt(125) *# 2 elevado a 3, raíz quadrada de 125* E3, E4**=**math.floor(**-**4.6), math.radians(60) *# maior int contido em -4.6 , seno de 60* print(f"E1={E1:3.0f}\t E2={E2:5.2f}")

print(f"E3={E3:3d}\t E4={E4:5.2f}")

**Operações com alfanuméricos**

**Alguns *métodos* da *classe string***

*str.capitalize()* Devolve uma cópia da string com o 1.º caráter capitalizado (em maiúscula)

*string.center(length[, character])* Centra uma string relativamente a um tamanho especificado (campo de edição com um nº de carateres especificado)e por defeito considera o carater espaço " "

*string.find(value[, start[, end]])* Encontra a primeira ocorrência do valor (value) especificado. *start* e *end* são parâmetros opcionais e definem a posição de início a partir do qual se pesquisará se o valor ocorre na string, e fim até que posição se efetuará a pesquisa na string.

Caso não se defina o intervalo de pesquisa, esta ocorrerá em toda a string.

Se não encontrado o valor é devolvido -1.

127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 2/13

23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook

Consulta os métodos para string em:Python references (https://docs.python.org/3.9/library/stdtypes.html? highlight=find#text-sequence-type-str) ou em W3Shools

(https://www.w3schools.com/python/python\_ref\_string.asp)

**Exemplos:**

In [ ]:

E1, E2 **=** len("Porto"), "Lisboa " *# tamanho de 'Porto'*

E3, E4 **=** "fica no norte ", "Coimbra".upper() *# converter string para maiúsculas* E5 **=** "Maria Sílvia Martins"[0:5] *# extrair um substring de uma string* E6, E7, E8**=**"Adriana", "Ruben", "Dinis"

E9, E10**=** "Lucas Samuel Porto", " Bom dia ".strip(" ") *#elimina espaços à esq. e à dta. d*

print(E1)

print(E2**+**E3)

print(E4)

print("E5: ",E5)

print("E6==E7: ",E6**==**E7)

print("E7<E8: ",E7**<**E8)

print("Find 'Porto':" ,E9.find("Porto",0, 10)) *# determina a posição do 1.º carater da #substring "Porto" na string E9,nas posições de 0 a 10*

print("Find 'Porto':",E9.find("Porto")) *# determina a posição do 1.º carater de "Porto" #em toda a string E9*

*#para verificar se uma string se encontra noutra é preferível usar in:* print("'Porto' está em 'Lucas Samuel Porto'?","Porto" **in** E9)

print(E10)

print(E10.center(20,"@"))

print(E10.center(20))

**Formatos para percentagens**

**Exemplos:**

In [ ]:

CD**=**0.456789

print("{0:5.3%}".format(CD))

print(f"{CD:5.2**%**}")

**Conversões de inteiros, reais, alfanuméricos e datas e alguns formatos para data e hora**

Podes consultar os *métodos* da classe *datetime* e os formatos permitidos para datas em: W3Shools (https://www.w3schools.com/python/python\_datetime.asp) ou em :docs.python

(https://docs.python.org/3/library/datetime.html)

**Exemplos:**

127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 3/13

23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook

In [ ]:

**import** datetime

I**=**int(5.76)

R**=**float(4)

S**=**str(4.345)

A**=**float('12.45')

D **=**datetime.datetime(2022,9,22,11,0,0)

x **=** datetime.datetime.now()

DHoje**=**datetime.datetime.now()

print(D.strftime("%H:%M"))

print(x.strftime("%d/%m/%y"))

print(f"{I} \t{S} \t{A}")

print(DHoje)

print(f"{DHoje:**%**d**/%**m**/%**y}") *#sem utilização do método strftime*

In [ ]:

**import** datetime

DataCorrente**=**datetime.datetime.today()

print(DataCorrente)

print(DataCorrente.strftime("%d-%m-%y"))

print(DataCorrente.strftime("%d-%m-%y %H:%M:%S"))

**Leitura de datas cronológicas**

In [ ]:

**import** datetime

Dnas**=**input("insira a sua data de nascimento (dd/mm/aaaa):") dia, mes, ano**=**map(int,Dnas.split('/'))

DataNascimento**=**datetime.date(ano,mes,dia)

print("Data de nascimento: ")

print(DataNascimento.strftime("%d-%m-%Y"))

print(DataNascimento.strftime("%d-%m-%y"))

print(f"{DataNascimento:**%**y**-%**m**-%**d}")

127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 4/13

23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook

In [88]:

*#é necessário instalar o pacote backports.zoneinfo para poder listar as timezone #no python executar: pip install backports.zoneinfo*

**from** datetime **import** date, datetime, tzinfo, timedelta, timezone **import** pytz

dt **=** datetime(2022, 10, 31, 12,0,0, tzinfo**=**ZoneInfo("America/Los\_Angeles")) print(dt)

print(datetime.time(dt))

print(dt.strftime("%H:%M:%S %Z"))

JST **=** timezone(timedelta(hours**=+**9))

dt **=** datetime(2022, 1, 1, 12, 0, 0, tzinfo**=**JST)

print(dt)

*# 2022-01-01 12:00:00+09:00*

print(dt.tzname())

*# UTC+09:00*

dt **=** datetime(2022, 1, 1, 12, 0, 0, tzinfo**=**timezone(timedelta(hours**=**9), 'JST')) print(dt.tzname())

*# 'JST'*

2022-10-31 12:00:00-07:00

12:00:00

12:00:00 PDT

2022-01-01 12:00:00+09:00

UTC+09:00

JST

127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 5/13

23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook

In [89]:

*#é necessário instalar o pacote backports.zoneinfo para poder listar as timezone #no python executar: pip install backports.zoneinfo*

**from** zoneinfo **import** ZoneInfo

zoneinfo.available\_timezones()

Out[89]:

{'Africa/Abidjan',

'Africa/Accra',

'Africa/Addis\_Ababa',

'Africa/Algiers',

'Africa/Asmara',

'Africa/Asmera',

'Africa/Bamako',

'Africa/Bangui',

'Africa/Banjul',

'Africa/Bissau',

'Africa/Blantyre',

'Africa/Brazzaville',

'Africa/Bujumbura',

'Africa/Cairo',

'Africa/Casablanca',

'Africa/Ceuta',

'Africa/Conakry',

'Africa/Dakar',

In [90]:

**from** datetime **import** date, datetime, tzinfo, timedelta, timezone

**import** pytz

uct\_time**=**datetime.utcnow()

time\_zone**=**pytz.utc.localize(uct\_time).astimezone()

*#a=tz.fromutc(utc\_time)*

print(time\_zone)

fuso\_hor**=**pytz.timezone("UTC")

time\_zone**=**pytz.utc.localize(utc\_time).astimezone(fuso\_hor)

print(time\_zone)

fuso\_hor **=** pytz.timezone('Europe/Paris')

time\_zone**=**pytz.utc.localize(utc\_time).astimezone(fuso\_hor)

*#utc\_time =utc\_time.replace(tzinfo=pytz.UTC) #replace method*

*# #time\_zone=utc\_time.astimezone(tz) #astimezone method*

print(time\_zone)

2022-09-23 11:50:40.117262+01:00

2022-09-23 10:19:33.054805+00:00

2022-09-23 12:19:33.054805+02:00

**Enumerações (enum)**

Enumeração, em Python, são um conjunto de nomes simbólicos (membros) vinculados a valores únicos e constantes. As enumerações podem ser usadas para criar tipos de dados personalizados simples que incluem coisas como estações, semanas, tipos de armas em um jogo, planetas, notas ou dias e podem ser iteradas.

127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 6/13

23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook

As enumerações em Python são criadas usando o módulo "enum". As enumerações são criadas usando classes. As **enumerações** têm nomes e valores que lhes estão associados. Deste módulo tem classes de enumeração que podem ser usadas para definir conjuntos de nomes e valores: Enum , IntEnum , Flag , and IntFlag . Também define um **decorator** (decorador), unique() , e um **auxiliar**, auto (automáticos).

***Enum*** - Classe base para criação de constantes enumeradas.

Por convenção, os nomes de enumeração começam com uma letra maiúscula e são singulares. Uma enumeração tem as seguintes caraterísticas:

As enumerações são representações de string de um objeto também chamado repr() (método). O nome da enumeração é exibido usando a palavra-chave ‘name’.

Usando type() podemos verificar o tipo de enumeração.

**Nota:** str() e repr() são usados para obter uma representação de string de um objeto. str() é usado para criar saída para utilizadores finais enquanto repr() é usado principalmente para depuração e desenvolvimento. O objetivo de repr é ser inequívoco e str é ser legível.

Exemplo:

In [91]:

**import** datetime

today **=** datetime.datetime.now()

*#imprime num formato legível para objetos de data e hora*

print (str(today))

*#imprime o formato oficial do objeto de data e hora*

print (repr(today))

2022-09-23 11:50:44.329288

datetime.datetime(2022, 9, 23, 11, 50, 44, 329288)

**Exemplos (classe enum):**

127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 7/13

23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook

In [92]:

*#Criação de classe Enum*

**from** enum **import** Enum

""" A enumeração de nome Estacao é criada com a palavra-chave class. Herda-se da classe bas Definimos explicitamente os números para os valores de enumeração, que são: PRIMAVERA, VERÃ Para aceder a um dos seus membros, especificamos o nome da enumeração seguido por um ponto

**class** Estacao(Enum):

PRIMAVERA **=** 1

VERAO **=** 2

OUTONO **=** 3

INVERNO **=** 4

*# imprime o nome do membro enum:*

print ("The name of enum member is : ",end **=**"")

print (Estacao.VERAO.name)

*# imprime o tipo do membro enum:*

print ("The type of enum member is : ",end **=**"")

print (type(Estacao.VERAO))

*# um valor de enumeração (membro enum) é atribuído a uma variável e é impresso na consola.* est **=** Estacao.PRIMAVERA

print(est)

*# imprime o membro enum como uma string:*

print ("O membro enum como uma string : ",end**=**"")

print (Estacao.PRIMAVERA)

*#Um valor de enumeração avaliado numa condição*

**if** est **==** Estacao.PRIMAVERA:

print("Primavera")

*#Listagem de todos os valores da enumeração usando a função list*

list(Estacao)

*# imprime o número representativo do membro enum:*

print ("Número representativo do valor da enumeração : ",end**=**"")

print (repr(Estacao.PRIMAVERA))

The name of enum member is : VERAO

The type of enum member is : <enum 'Estacao'>

Estacao.PRIMAVERA

O membro enum como uma string : Estacao.PRIMAVERA

Primavera

Número representativo do valor da enumeração : <Estacao.PRIMAVERA: 1>

**Garantir valores únicos para numa numeração** Por padrão, as enumerações permitem vários nomes como aliases para o mesmo valor. Quando esse comportamento não é desejado,

usa-se o decorador @enum.unique para garantir que não haja repetição de valores na enumeração. **Exemplo:**

127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 8/13

23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook

In [93]:

**import** enum

*# Using enum class create enumerations* **class** Days(enum.Enum):

Sun **=** 1

Mon **=** 2

Tue **=** 3*# imprime o membro enum como uma string:*

print ("O membro enum como uma string : ",end**=**"") print (Days.Mon)

*# print the enum member as a repr*

print ("he enum member as a repr is : ",end**=**"") print (repr(Days.Sun))

*# Check type of enum member*

print ("The type of enum member is : ",end **=**"") print (type(Days.Mon))

*# print name of enum member*

print ("The name of enum member is : ",end **=**"") print (Days.Tue.name)

O membro enum como uma string : Days.Mon he enum member as a repr is : <Days.Sun: 1> The type of enum member is : <enum 'Days'> The name of enum member is : Tue

127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 9/13

23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook

In [94]:

*#Garantir valores de enumeração exclusivos*

**from** enum **import** Enum, unique

@unique

**class** Mistake(Enum):

ONE **=** 1

TWO **=** 2

THREE **=** 3

FOUR **=** 3

**--------------------------------------------------------------------------- ValueError** Traceback (most recent call last) **C:\Users\SLVIAM~1\AppData\Local\Temp/ipykernel\_2112/993266228.py** in <module> 2 **from** enum **import** Enum**,** unique

3 **@**unique

**----> 4 class** Mistake**(**Enum**):**

5 ONE **= 1**

6 TWO **= 2**

**C:\ProgramData\Anaconda3\lib\enum.py** in unique**(enumeration)**

1011 alias\_details = ', '.join(

1012 ["%s -> %s" % (alias, name) for (alias, name) in dup licates])

**-> 1013 raise ValueError('duplicate values found in %r: %s' %** 1014 (enumeration, alias\_details))

1015 **return** enumeration

**ValueError**: duplicate values found in <enum 'Mistake'>: FOUR -> THREE

**Utilização de valores automáticos** Se não for necessário explicitar determinados valores, pode-se recorrer ao auxiliar auto .

**Exemplo:**

In [95]:

**from** enum **import** Enum, auto

**class** Color(Enum):

RED **=** auto()

BLUE **=** auto()

GREEN **=** auto()

list(Color)

Out[95]:

[<Color.RED: 1>, <Color.BLUE: 2>, <Color.GREEN: 3>]

A classe IntEnum , certifica-se de que os membros são do tipo inteiro.

**Exemplo:**

127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 10/13

23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook

In [96]:

**from** enum **import** IntEnum

**class** Color(IntEnum):

RED **=** 1

BLUE **=** a

**--------------------------------------------------------------------------- TypeError** Traceback (most recent call last) **C:\Users\SLVIAM~1\AppData\Local\Temp/ipykernel\_2112/4038971975.py** in <module >

1 **from** enum **import** IntEnum

**----> 2 class** Color**(**IntEnum**):**

3 RED **= 1**

4 BLUE **=** a

**C:\ProgramData\Anaconda3\lib\enum.py** in \_\_new\_\_**(metacls, cls, bases, classdi ct, \*\*kwds)**

286 enum\_member**.**\_value\_ **=** value

287 **else:**

**--> 288** enum\_member **=** \_\_new\_\_**(**enum\_class**, \***args**)** 289 **if not** hasattr**(**enum\_member**, '\_value\_'):**

290 **if** member\_type **is** object**:**

**TypeError**: int() argument must be a string, a bytes-like object or a number, not 'datetime.datetime'

In [97]:

**from** enum **import** IntEnum

**class** Color(IntEnum):

RED **=** 1

BLUE **=** 2

Color.BLUE**==**2

Out[97]:

True

\*\*Enumeração com combinação dos seus membros usando a classe Flag e os operadores bit a bit & (AND), | (OR), ^ (XOR), and ~ (INVERT)

**Exemplo:**

127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 11/13

23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook

In [98]:

**from** enum **import** Flag, auto

*#os valores das flags individuais serão sempre potências de 2 (1,2,4,8...), as combinações* **class** Dia(Flag):

SEGUNDA **=** auto() *#1*

TERCA **=** auto() *#2*

QUARTA **=** auto() *#4*

QUINTA **=** auto() *#8*

SEXTA **=** auto() *#16*

SABADO **=** auto() *#32*

DOMINGO **=** auto() *#64*

*#flag fds*

fds **=** Dia.SABADO **|** Dia.DOMINGO

*#flag dds*

dds **=**Dia.QUARTA **|** Dia.QUINTA **|** Dia.SEGUNDA **|** Dia.SEXTA **|** Dia.TERCA

print(fds)

*#my\_it = iter(dds) verifica se um objeto é iterável#RESOLVER ????PQ o objeto flag NÃO É #list(my\_it)*

print(Dia(8))

print("verifica se o dia inserido é membro da flag dds")

d**=**input("Qual o dia?")

**if** d**==** Dia.SABADO.name **or** d**==**Dia.DOMINGO.name:

print(d," é Fim de semana.")

**else**:

print(d," é Dia da semana.")

print("imprime todos os elementos da enumeração Dia")

**for** d **in** Dia:

print(d)

print("Imprime o nº de membros da enumeração Dia")

print(len(Dia))

print("Imprime os membros da flag fds")

print(fds)

print("Escreve o total de valores correspondentes aos membros da flag fds") print(fds.value)

print("Escreve o nome da enumeração ")

print(Dia)

print("Imprime os membros que pertencem às duas flag : dds e fds")

print (dds**^**fds )

print("~fds imprime os membros da enumeração que não estão na flag fds") print(**~**fds)

Dia.DOMINGO|SABADO

Dia.QUINTA

verifica se o dia inserido é membro da flag dds

Qual o dia?QUINTA

QUINTA é Dia da semana.

imprime todos os elementos da enumeração Dia

Dia.SEGUNDA

127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 12/13

23/09/22, 11:51 NB02. 10793 - Operações básicas e formatos de dados em Python - Jupyter Notebook

Dia.TERCA

Dia.QUARTA

Dia.QUINTA

Dia.SEXTA

Dia.SABADO

Dia.DOMINGO

Imprime o nº de membros da enumeração Dia

7

Imprime os membros da flag fds

Dia.DOMINGO|SABADO

Escreve o total de valores correspondentes aos membros da flag fds

96

Escreve o nome da enumeração

<enum 'Dia'>

Imprime os membros que pertencem às duas flag : dds e fds

Dia.DOMINGO|SABADO|SEXTA|QUINTA|QUARTA|TERCA|SEGUNDA

~fds imprime os membros da enumeração que não estão na flag fds

Dia.SEXTA|QUINTA|QUARTA|TERCA|SEGUNDA

**Bibliografia**

Caleiro.C., Ramos.J.(2016). Notebook 01 - Conceitos básicos da linguagem Python Dep. Matemática - IST.Lisboa:IST

https://www.w3schools.com/python/default.asp (https://www.w3schools.com/python/default.asp) Vasconcelos, J. (2015).Python - Algoritmia e Programação Web. Lisboa:FCA

https://docs.python.org/3.9/library/stdtypes.html?highlight=find#text-sequence-type-str

(https://docs.python.org/3.9/library/stdtypes.html?highlight=find#text-sequence-type-str)

https://docs.python.org/pt-br/3.6/library/datatypes.html (https://docs.python.org/pt-br/3.6/library/datatypes.html) https://www.tutorialspoint.com/enum-in-python (https://www.tutorialspoint.com/enum-in-python) https://www.geeksforgeeks.org/str-vs-repr-in-python/ (https://www.geeksforgeeks.org/str-vs-repr-in-python/) https://docs.python.org/pt-br/3.6/library/enum.html#enum.IntFlag (https://docs.python.org/pt br/3.6/library/enum.html#enum.IntFlag)

https://docs.python.org/pt-br/3/contents.html (https://docs.python.org/pt-br/3/contents.html) https://zetcode.com/python/enum/ (https://zetcode.com/python/enum/)

https://pypi.org/project/backports.zoneinfo/ (https://pypi.org/project/backports.zoneinfo/)

https://adamj.eu/tech/2021/05/06/how-to-list-all-timezones-in-python/ (https://adamj.eu/tech/2021/05/06/how-to list-all-timezones-in-python/)





127.0.0.1:8889/notebooks/Desktop/notebook\_s/NB02. 10793 - Operações básicas e formatos de dados em Python.ipynb# 13/13