



INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL



Cofinanciado pela
União Europeia

Os Fundos Europeus mais próximos de si.



REPÚBLICA
PORTUGUESA

TRABALHO, SOLIDARIEDADE
E SEGURANÇA SOCIAL



UFCD 10793 - Trabalhar com strings

Strings

validação de dados

Strings

- As strings permitem-nos guardar qualquer tipo de informação, uma vez que permitem armazenar qualquer sequência de caracteres.
- Se por exemplo tentássemos guardar um código postal como um inteiro, não conseguiríamos armazenar o “-” a separar ambas as partes deste código, posto isto, temos que optar obrigatoriamente por guardá-lo como string.

```
codPostal = input("Qual o seu código postal?\n")
```

```
print("Código postal inserido:", codPostal);
```

```
print("O tipo de dados da variável que contem o código postal é:", type(codPostal))
```



Esta variável será vista como uma string, uma vez que o `input()` por natureza devolve uma string - contendo os dados do utilizador.



`type(x)` permite-nos identificar o tipo de “dados” (na realidade trata-se do tipo de objeto, uma vez que no Python, todos os elementos são objetos).



C:\Users\userHP\AppData\Local\Programs\Python\Python39\python.exe

Qual o seu código postal?

3670-555

Código postal inserido: 3670-555

O tipo de dados da variável que contem o código postal é: `<class 'str'>`

Press any key to continue . . .



Confirma-se que a variável ficou do tipo string

Strings - validação de dados

- Sendo uma string (**apenas funciona com strings!**) é possível fazer validações de dados, por forma a por exemplo obrigar o utilizador a que sejam respeitados determinados critérios.

```
nome = input("Qual o seu nome?\n")  
if nome.isalpha():  
    print("Tudo OK!!")  
else:  
    print("Hum? Algo está errado!")
```

Instrução	Descrição
.isalpha()	Carateres alfabéticos a-z, A-Z, letras com acentos e “ç” (Não são permitidos espaços!)
.isalnum()	Carateres alfanuméricos 0-9, a-z, A-Z, letras com acentos e “ç” (As mesmas regras do anterior, apenas acrescem os números!)
.istitle()	Maiúsculas no início das palavras, seguido de minúsculas: Exemplo: <u>A</u> na Faria - 15 <u>A</u> nos @Viseu (Permite ainda que existam números, letras e símbolos!)
.islower()	Apenas minúsculas e números: a-z, letras com acentos e “ç” e 0-9
.isupper()	Apenas maiúsculas e números: A-Z, letras com acentos e “ç” e 0-9

Strings - validação de dados

- Existe ainda a possibilidade de validar números.
- Existem 3 formas de validar um número: `.isdigit()`, `.isdecimal()` e `.isnumeric()`
 - A diferença entre os vários tipos está nos caracteres numéricos que são aceites, uma vez que existe uma vasta lista de grupos de caracteres distintos, do qual são abrangidos por exemplo os caracteres chineses, romanos, símbolos com caracteres, etc.:

```
idade = input("Qual a sua idade?\n")  
  
if idade.isdigit():  
    print("Idade válida")  
else:  
    print("Idade inválida")
```



```
C:\Users\userHP\AppData\Local\Programs\Python\Pytl  
Qual a sua idade?  
12  
Idade válida  
Press any key to continue . . .
```

Instrução	
<code>.isnumeric()</code>	Qualquer carater que represente um número
<code>.isdigit()</code>	O mais utilizado no Python, porque representa maioritariamente os caracteres com aspeto de: 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9
<code>.isdecimal()</code>	Qualquer carater da tabela Unicode (expansão da tabela ASCII) respeitante ao grupo dos decimais



Nota: Casas decimais ou números negativos não são complementados nestes métodos!

Strings - validação de dados

- Exemplos quando `.isnumeric() = true`, `.isdigit() = true`, `.isdecimal() = true`

```
"0123456789"    DIGIT ZERO~NINE
"٠١٢٣٤٥٦٧٨٩" ARABIC-INDIC DIGIT ZERO~NINE
"०१२३४५६७८९" DEVANAGARI DIGIT ZERO~NINE
"০১২৩৪৫৬৭৮৯" BENGALI DIGIT ZERO~NINE
"੦੧੨੩੪੫੬੭੮੯" GURMUKHI DIGIT ZERO~NINE
"૦૧૨૩૪૫૬૭૮૯" GUJARATI DIGIT ZERO~NINE
"୦୧୨୩୪୫୬୭୮୯" ORIYA DIGIT ZERO~NINE
"௦௧௨௩௪௫௬௭௮௯" TAMIL DIGIT ZERO~NINE
"౦౧౨౩౪౫౬౭౮౯" TELUGU DIGIT ZERO~NINE
"೦೧೨೩೪೫೬೭೮೯" KANNADA DIGIT ZERO~NINE
"൦൧൨൩൪൫൬൭൮൯" MALAYALAM DIGIT ZERO~NINE
"๐๑๒๓๔๕๖๗๘๙" THAI DIGIT ZERO~NINE
"໐໑໒໓໔໕໖໗໘໙" LAO DIGIT ZERO~NINE
"ཀྲཱུང་ཁྲཱུང་" TIBETAN DIGIT ZERO~NINE
"၀၁၂၃၄၅၆၇၈၉" MYANMAR DIGIT ZERO~NINE
"០១២៣៤៥៦៧៨៩" KHMER DIGIT ZERO~NINE
" 0 1 2 3 4 5 6 7 8 9 " FULLWIDTH DIGIT ZERO~NINE
"0123456789"    MATHEMATICAL BOLD DIGIT ZERO~NINE
"0123456789"    MATHEMATICAL DOUBLE-STROKE DIGIT ZERO~NINE
"0123456789"    MATHEMATICAL SANS-SERIF DIGIT ZERO~NINE
"0123456789"    MATHEMATICAL SANS-SERIF BOLD DIGIT ZERO~NINE
"0123456789"    MATHEMATICAL MONOSPACE DIGIT ZERO~NINE
```

Strings - validação de dados

- Exemplos quando `.isnumeric() = true`, `.isdigit() = true`, `.isdecimal() = false`

```
"0 1 2 3 4 5 6 7 8 9" SUPERScript ZERO~NINE
"0 1 2 3 4 5 6 7 8 9" SUBSCRIPT ZERO~NINE
"0 1 2 3 4 5 6 7 8 9" DIGIT ZERO~NINE FULL STOP
"0,1,2,3,4,5,6,7,8,9" DIGIT ZERO~NINE COMMA
"0 1 2 3 4 5 6 7 8 9" CIRCLED DIGIT ZERO~NINE
"0 1 2 3 4 5 6 7 8 9" NEGATIVE CIRCLED DIGIT ZERO~NINE
"(1)(2)(3)(4)(5)(6)(7)(8)(9)" PARENTHESIZED DIGIT ONE~NINE
"①②③④⑤⑥⑦⑧⑨" DINGBAT CIRCLED SANS-SERIF DIGIT ONE~NINE
"⓪①②③④⑤⑥⑦⑧⑨" DOUBLE CIRCLED DIGIT ONE~NINE
"⓪①②③④⑤⑥⑦⑧⑨" DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT ONE~NINE
"፩፪፫፬፭፮፯፰፱" ETHIOPIC DIGIT ONE~NINE
```

Strings - validação de dados

- Exemplos quando `.isnumeric() = true`, `.isdigit() = false`, `.isdecimal() = false`

"𑌕𑌖𑌗𑌘𑌙𑌚𑌛𑌜𑌝𑌞𑌟𑌠𑌡𑌢𑌣𑌤𑌥𑌦𑌧𑌨𑌩𑌪𑌫𑌬𑌭𑌮𑌯𑌰𑌱𑌲𑌳𑌴𑌵𑌶𑌷𑌸𑌹𑌺𑌻𑌼𑌽𑌾𑌿𑍀𑍁𑍂𑍃𑍄𑍅𑍆𑍇𑍈𑍉𑍊𑍋𑍌𑍍𑍎𑍏𑍐𑍑𑍒𑍓𑍔𑍕𑍖𑍗𑍘𑍙𑍚𑍛𑍜𑍝𑍞𑍟𑍠𑍡𑍢𑍣𑍤𑍥𑍦𑍧𑍨𑍩𑍪𑍫𑍬𑍭𑍮𑍯𑍰𑍱𑍲𑍳𑍴𑍵𑍶𑍷𑍸𑍹𑍺𑍻𑍼𑍽𑍾𑍿𑎀𑎁𑎂𑎃𑎄𑎅𑎆𑎇𑎈𑎉𑎊𑎋𑎌𑎍𑎎𑎏𑎐𑎑𑎒𑎓𑎔𑎕𑎖𑎗𑎘𑎙𑎚𑎛𑎜𑎝𑎞𑎟𑎠𑎡𑎢𑎣𑎤𑎥𑎦𑎧𑎨𑎩𑎪𑎫𑎬𑎭𑎮𑎯𑎰𑎱𑎲𑎳𑎴𑎵𑎶𑎷𑎸𑎹𑎺𑎻𑎼𑎽𑎾𑎿𑏀𑏁𑏂𑏃𑏄𑏅𑏆𑏇𑏈𑏉𑏊𑏋𑏌𑏍𑏎𑏏𑏐𑏑𑏒𑏓𑏔𑏕𑏖𑏗𑏘𑏙𑏚𑏛𑏜𑏝𑏞𑏟𑏠𑏡𑏢𑏣𑏤𑏥𑏦𑏧𑏨𑏩𑏪𑏫𑏬𑏭𑏮𑏯𑏰𑏱𑏲𑏳𑏴𑏵𑏶𑏷𑏸𑏹𑏺𑏻𑏼𑏽𑏾𑏿𑐀𑐁𑐂𑐃𑐄𑐅𑐆𑐇𑐈𑐉𑐊𑐋𑐌𑐍𑐎𑐏𑐐𑐑𑐒𑐓𑐔𑐕𑐖𑐗𑐘𑐙𑐚𑐛𑐜𑐝𑐞𑐟𑐠𑐡𑐢𑐣𑐤𑐥𑐦𑐧𑐨𑐩𑐪𑐫𑐬𑐭𑐮𑐯𑐰𑐱𑐲𑐳𑐴𑐵𑐶𑐷𑐸𑐹𑐺𑐻𑐼𑐽𑐾𑐿𑑀𑑁𑑂𑑃𑑄𑑅𑑆𑑇𑑈𑑉𑑊𑑋𑑌𑑍𑑎𑑏𑑐𑑑𑑒𑑓𑑔𑑕𑑖𑑗𑑘𑑙𑑚𑑛𑑜𑑝𑑞𑑟𑑠𑑡𑑢𑑣𑑤𑑥𑑦𑑧𑑨𑑩𑑪𑑫𑑬𑑭𑑮𑑯𑑰𑑱𑑲𑑳𑑴𑑵𑑶𑑷𑑸𑑹𑑺𑑻𑑼𑑽𑑾𑑿𑒀𑒁𑒂𑒃𑒄𑒅𑒆𑒇𑒈𑒉𑒊𑒋𑒌𑒍𑒎𑒏𑒐𑒑𑒒𑒓𑒔𑒕𑒖𑒗𑒘𑒙𑒚𑒛𑒜𑒝𑒞𑒟𑒠𑒡𑒢𑒣𑒤𑒥𑒦𑒧𑒨𑒩𑒪𑒫𑒬𑒭𑒮𑒯𑒰𑒱𑒲𑒳𑒴𑒵𑒶𑒷𑒸𑒻𑒻𑒼𑒽𑒾𑒿𑓀𑓁𑓃𑓂𑓄𑓅𑓆𑓇𑓈𑓉𑓊𑓋𑓌𑓍𑓎𑓏𑓐𑓑𑓒𑓓𑓔𑓕𑓖𑓗𑓘𑓙𑓚𑓛𑓜𑓝𑓞𑓟𑓠𑓡𑓢𑓣𑓤𑓥𑓦𑓧𑓨𑓩𑓪𑓫𑓬𑓭𑓮𑓯𑓰𑓱𑓲𑓳𑓴𑓵𑓶𑓷𑓸𑓹𑓺𑓻𑓼𑓽𑓾𑓿𑔀𑔁𑔂𑔃𑔄𑔅𑔆𑔇𑔈𑔉𑔊𑔋𑔌𑔍𑔎𑔏𑔐𑔑𑔒𑔓𑔔𑔕𑔖𑔗𑔘𑔙𑔚𑔛𑔜𑔝𑔞𑔟𑔠𑔡𑔢𑔣𑔤𑔥𑔦𑔧𑔨𑔩𑔪𑔫𑔬𑔭𑔮𑔯𑔰𑔱𑔲𑔳𑔴𑔵𑔶𑔷𑔸𑔹𑔺𑔻𑔼𑔽𑔾𑔿𑕀𑕁𑕂𑕃𑕄𑕅𑕆𑕇𑕈𑕉𑕊𑕋𑕌𑕍𑕎𑕏𑕐𑕑𑕒𑕓𑕔𑕕𑕖𑕗𑕘𑕙𑕚𑕛𑕜𑕝𑕞𑕟𑕠𑕡𑕢𑕣𑕤𑕥𑕦𑕧𑕨𑕩𑕪𑕫𑕬𑕭𑕮𑕯𑕰𑕱𑕲𑕳𑕴𑕵𑕶𑕷𑕸𑕹𑕺𑕻𑕼𑕽𑕾𑕿𑖀𑖁𑖂𑖃𑖄𑖅𑖆𑖇𑖈𑖉𑖊𑖋𑖌𑖍𑖎𑖏𑖐𑖑𑖒𑖓𑖔𑖕𑖖𑖗𑖘𑖙𑖚𑖛𑖜𑖝𑖞𑖟𑖠𑖡𑖢𑖣𑖤𑖥𑖦𑖧𑖨𑖩𑖪𑖫𑖬𑖭𑖮𑖯𑖰𑖱𑖲𑖳𑖴𑖵𑖶𑖷𑖸𑖹𑖺𑖻𑖼𑖽𑖾𑗀𑖿𑗁𑗂𑗃𑗄𑗅𑗆𑗇𑗈𑗉𑗊𑗋𑗌𑗍𑗎𑗏𑗐𑗑𑗒𑗓𑗔𑗕𑗖𑗗𑗘𑗙𑗚𑗛𑗜𑗝𑗞𑗟𑗠𑗡𑗢𑗣𑗤𑗥𑗦𑗧𑗨𑗩𑗪𑗫𑗬𑗭𑗮𑗯𑗰𑗱𑗲𑗳𑗴𑗵𑗶𑗷𑗸𑗹𑗺𑗻𑗼𑗽𑗾𑗿𑘀𑘁𑘂𑘃𑘄𑘅𑘆𑘇𑘈𑘉𑘊𑘋𑘌𑘍𑘎𑘏𑘐𑘑𑘒𑘓𑘔𑘕𑘖𑘗𑘘𑘙𑘚𑘛𑘜𑘝𑘞𑘟𑘠𑘡𑘢𑘣𑘤𑘥𑘦𑘧𑘨𑘩𑘪𑘫𑘬𑘭𑘮𑘯𑘰𑘱𑘲𑘳𑘴𑘵𑘶𑘷𑘸𑘹𑘺𑘻𑘼𑘽𑘾𑘿𑙀𑙁𑙂𑙃𑙄𑙅𑙆𑙇𑙈𑙉𑙊𑙋𑙌𑙍𑙎𑙏𑙐𑙑𑙒𑙓𑙔𑙕𑙖𑙗𑙘𑙙𑙚𑙛𑙜𑙝𑙞𑙟𑙠𑙡𑙢𑙣𑙤𑙥𑙦𑙧𑙨𑙩𑙪𑙫𑙬𑙭𑙮𑙯𑙰𑙱𑙲𑙳𑙴𑙵𑙶𑙷𑙸𑙹𑙺𑙻𑙼𑙽𑙾𑙿𑚀𑚁𑚂𑚃𑚄𑚅𑚆𑚇𑚈𑚉𑚊𑚋𑚌𑚍𑚎𑚏𑚐𑚑𑚒𑚓𑚔𑚕𑚖𑚗𑚘𑚙𑚚𑚛𑚜𑚝𑚞𑚟𑚠𑚡𑚢𑚣𑚤𑚥𑚦𑚧𑚨𑚩𑚪𑚫𑚬𑚭𑚮𑚯𑚰𑚱𑚲𑚳𑚴𑚵𑚷𑚶𑚸𑚹𑚺𑚻𑚼𑚽𑚾𑚿𑛀𑛁𑛂𑛃𑛄𑛅𑛆𑛇𑛈𑛉𑛊𑛋𑛌𑛍𑛎𑛏𑛐𑛑𑛒𑛓𑛔𑛕𑛖𑛗𑛘𑛙𑛚𑛛𑛜𑛝𑛞𑛟𑛠𑛡𑛢𑛣𑛤𑛥𑛦𑛧𑛨𑛩𑛪𑛫𑛬𑛭𑛮𑛯𑛰𑛱𑛲𑛳𑛴𑛵𑛶𑛷𑛸𑛹𑛺𑛻𑛼𑛽𑛾𑛿𑜀𑜁𑜂𑜃𑜄𑜅𑜆𑜇𑜈𑜉𑜊𑜋𑜌𑜍𑜎𑜏𑜐𑜑𑜒

Strings - validação de dados

- Em qualquer um destes modos de verificação de dados, **se apenas existirem espaços ou vazio**, o resultado é dado como inválido:

```
print(" ".isalpha())
print(" ".isalnum())
print(" ".istitle())
print(" ".islower())
print(" ".isupper())
print(" ".isnumeric())
print(" ".isdecimal())
print(" ".isdigit())
```

[illegible]

```
print("".isalpha())
print("".isalnum())
print("".istitle())
print("".islower())
print("".isupper())
print("".isnumeric())
print("".isdecimal())
print("".isdigit())
```

[illegible]

Strings - validação de dados

- Em alguns modos (**desde que exista conteúdo**) são permitidos **espaços antes e\ou depois**:

Antes:

```
print(" a".isalpha())
print(" 1".isalnum())
print(" Aa".istitle())
print(" aa".islower())
print(" AA".isupper())
print(" 11".isnumeric())
print(" 11".isdecimal())
print(" 11".isdigit())
```

Depois:

```
print("a ".isalpha())
print("1 ".isalnum())
print("Aa ".istitle())
print("aa ".islower())
print("AA ".isupper())
print("11 ".isnumeric())
print("11 ".isdecimal())
print("11 ".isdigit())
```

ou

Antes e depois:

```
print(" a ".isalpha())
print(" 1 ".isalnum())
print(" Aa ".istitle())
print(" aa ".islower())
print(" AA ".isupper())
print(" 11 ".isnumeric())
print(" 11 ".isdecimal())
print(" 11 ".isdigit())
```

ou



```
C:\Users\userHP\AppData\Local\Programs\
False
False
True
True
True
False
False
False
False
Press any key to continue . . .
```



Apenas os `istitle()`, `islower()`
e `isupper()` permitem espaços
antes e\ou depois do texto

Strings - validação de dados

Exercício 1

Crie um programa que solicite ao utilizador os seguintes dados e apresente se são válidos ou não:

- Primeiro nome pessoal
- Localidade
- Número da porta

No fim apresente os dados todos.

Nota: O utilizador deverá obrigatoriamente inserir os 3 campos válidos. Obrigue-o!

► Cábula:

Instrução	Descrição
.isalpha()	Carateres alfabéticos
.isalnum()	Carateres alfanuméricos
.istitle()	Maiúsculas no inicio das palavras, seguido de minúsculas
.islower()	Apenas minúsculas e números
.isupper()	Apenas maiúsculas e números
.isdigit()	Apenas números

```
nome = input("Qual o seu nome?\n")
if nome.isalpha():
    print("Tudo OK!!")
else:
    print ("Hum? Algo está errado!")
```

Strings - validação de dados

Exercício 1 (solução)

```
nome = ""
localidade = ""
nPorta = ""

while True:
    if nome == "": #pedir nome
        nome = input("Qual o seu nome?\n")
        if not nome.istitle():
            print("Nome inválido!")
            nome = ""
            continue
    elif localidade == "": #pedir localidade
        localidade = input("Qual a sua localidade?\n")
        if not localidade.istitle():
            print("Localidade inválida!")
            localidade = ""
            continue
    elif nPorta == "": #Pedir nº porta
        nPorta = input("Qual o número da sua porta?\n")
        if not nPorta.isdigit():
            print("Número inválido!")
            nPorta = ""
            continue
    else:
        break #tudo OK fechar!

print(f"Nome: {nome} Localidade: {localidade} Nº da Porta: {nPorta}")
```




```
C:\Users\userHP\AppData\Local\Programs\Python\Python39\python.exe
Qual o seu nome?
pedro ferreira
Nome inválido!
Qual o seu nome?
Pedro Ferreira
Qual a sua localidade?
viseu
Localidade inválida!
Qual a sua localidade?
Viseu
Qual o número da sua porta?
e
Número inválido!
Qual o número da sua porta?
12
Nome: Pedro Ferreira Localidade: Viseu Nº da Porta: 12
Press any key to continue . . .
```

Strings - validação de dados

Exercício 1 (alternativa com uma função para recolher e validar)

```
def validaCampos(pergunta, tipo):  
    while True:  
        dados = input(pergunta)  
  
        if (tipo == 'title' and dados.istitle()) or (tipo == 'num' and dados.isdigit()):  
            break  
        else:  
            print("Dados inválidos!")  
  
    return dados  
  
nome = validaCampos("Qual o seu nome?\n", "title")  
localidade = validaCampos("Qual a sua localidade?\n", "title")  
nPorta = validaCampos("Qual o número da sua porta?\n", "num")  
  
print(f"Nome: {nome} Localidade: {localidade} Nº da Porta: {nPorta}")
```



```
C:\Users\userHP\AppData\Local\Programs\Python\Python39\pyth  
Qual o seu nome?  
pedro  
Dados inválidos!  
Qual o seu nome?  
Pedro  
Qual a sua localidade?  
Viseu  
Qual o número da sua porta?  
f  
Dados inválidos!  
Qual o número da sua porta?  
23  
Nome: Pedro Localidade: Viseu Nº da Porta: 23  
Press any key to continue . . .
```

Strings

- É ainda possível verificar se uma String começa (`startswith("xxx")`) ou acaba (`endswith("xxx")`) com um determinado texto:

```
tel = input("Indique o seu contacto telefónico:\n")  
  
if( tel.startswith("+351") ):  
    print("O seu contacto é de Portugal!")  
else:  
    print("O seu contacto é do estrangeiro!")
```



```
C:\Users\userHP\AppData\Local\Programs\Python\Pyth  
Indique o seu contacto telefónico:  
+351911155554  
O seu contacto é de Portugal!  
Press any key to continue . . .
```

```
nome = input("Indique o seu nome:\n")  
  
if( nome.endswith("Ferreira") ):  
    print("Vejo que é da família  
Ferreira!")  
else:  
    print("Não és da família Ferreira")
```



```
C:\Users\userHP\AppData\Local\Programs\Py  
Indique o seu nome:  
Tiago Reis  
Não és da família Ferreira  
Press any key to continue . . .
```

Strings - validação de dados

Exercício 2

Crie um programa que solicite de uma só vez ao utilizador o seu título (Eng. ou Dr.) e nome completo.

O nome deverá respeitar os padrões convencionais dos nomes (letra maiúscula seguida de minúsculas).

Caso seja válido:

- Se se tratar do Dr. diga-lhe bom dia.
- Se se tratar do Eng. mande-o trabalhar.

Caso seja inválido:

- Mande-o embora!

► Cábula:

Instrução	Descrição
.isalpha()	Carateres alfabéticos
.isalnum()	Carateres alfanuméricos
.istitle()	Maiúsculas no inicio das palavras, seguido de minúsculas
.islower()	Apenas minúsculas e números
.isupper()	Apenas maiúsculas e números
.isdigit()	Apenas números

```
tel = input("Indique o seu contacto telefónico:\n")
```

```
if( tel.startswith("+351") ):  
    print("O seu contacto é de Portugal!")  
else:  
    print("O seu contacto é do estrangeiro!")
```

```
nome = input("Indique o seu nome:\n")
```

```
if( nome.endswith("Ferreira") ):  
    print("Vejo que é da familia Ferreira!")  
else:  
    print("Não és da familia Ferreira")
```

Strings - validação de dados

Exercício 2 (solução)

```
nome = input ("Qual o seu nome?\n")

if nome.istitle():
    if nome.startswith("Dr."):
        print ("Bom dia")
    elif nome.startswith("Eng."):
        print ("Vai trabalhar!")
else:
    print ("Vai embora!")
```



```
C:\Users\userHP\AppData\Local\Programs\Python\Python39\python.exe
Qual o seu nome?
Dr. Pedro
Bom dia
Press any key to continue . . .
```

```
C:\Users\userHP\AppData\Local\Programs\Python\Python39\python.exe
Qual o seu nome?
Eng. Pedro
Vai trabalhar!
Press any key to continue . . .
```

```
C:\Users\userHP\AppData\Local\Programs\Python\Python39\python.exe
Qual o seu nome?
pedro
Vai embora!
Press any key to continue . . .
```


Strings

Conversão

Strings - conversão

- É ainda possível converter os caracteres dentro de uma string.

```
nome = input ("Qual o seu nome?\n")  
  
print("Tive o cuidado de o converter devidamente:", nome.title() )
```



```
C:\Users\userHP\AppData\Local\Programs\Python\Python39\python.exe  
Qual o seu nome?  
pedro ferreira  
Tive o cuidado de o converter devidamente:  Pedro ferreira  
Press any key to continue . . . _
```

Instrução	Descrição
.capitalize()	Converte para maiúscula a primeira letra de todo o texto e todas as restantes em minúsculas
.title()	Por cada palavra existente no texto: converte para maiúscula a primeira letra e as restantes em minúsculas
.lower()	Converte todos os caracteres para minúsculas
.upper()	Converte todos os caracteres de para minúsculas
.swapcase()	Converte todos os caracteres maiúsculos em minúsculos e vice-versa

Strings - conversão

Exercício 3

Crie um programa que solicite o nome do utilizador.

Caso ele não indique um nome (nome a vazio):

- Pedir o preenchimento novamente

Caso ele preencha o nome:

- Apresente o nome convertido em todos os formatos da grelha de conversões ao lado.

► Cácula:

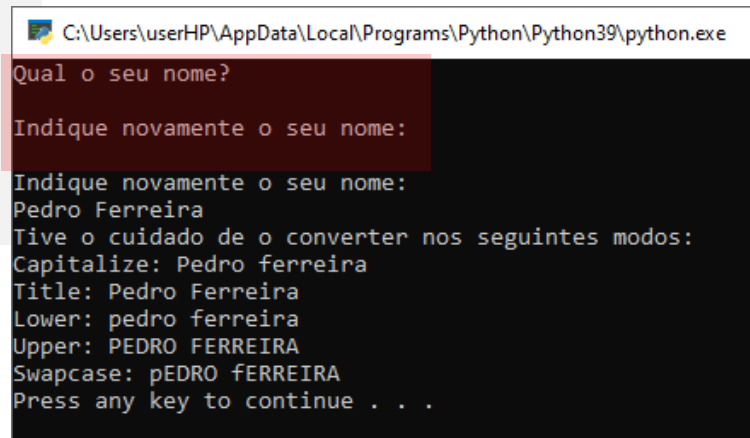
Instrução	Descrição
<code>.capitalize()</code>	Converte para maiúscula a primeira letra de todo o texto
<code>.title()</code>	Por cada palavra existente no texto: converte para maiúscula a primeira letra e as restantes em minúsculas
<code>.lower()</code>	Converte todos os caracteres para minúsculas
<code>.upper()</code>	Converte todos os caracteres de para minúsculas
<code>.swapcase()</code>	Converte todos os caracteres maiúsculos em minúsculos e vice-versa

Strings - conversão

Exercício 3 (solução)

```
nome = input ("Qual o seu nome?\n")

while True:
    if nome != "":
        print("Tive o cuidado de o converter nos seguintes modos: ")
        print("Capitalize:", nome.capitalize())
        print("Title:", nome.title())
        print("Lower:", nome.lower())
        print("Upper:", nome.upper())
        print("Swapcase:", nome.swapcase())
        break
    else:
        nome = input ("Indique novamente o seu nome:\n")
```



```
C:\Users\userHP\AppData\Local\Programs\Python\Python39\python.exe
Qual o seu nome?
Indique novamente o seu nome:
Indique novamente o seu nome:
Pedro Ferreira
Tive o cuidado de o converter nos seguintes modos:
Capitalize: Pedro ferreira
Title: Pedro Ferreira
Lower: pedro ferreira
Upper: PEDRO FERREIRA
Swapcase: pEDRO fERREIRA
Press any key to continue . . .
```

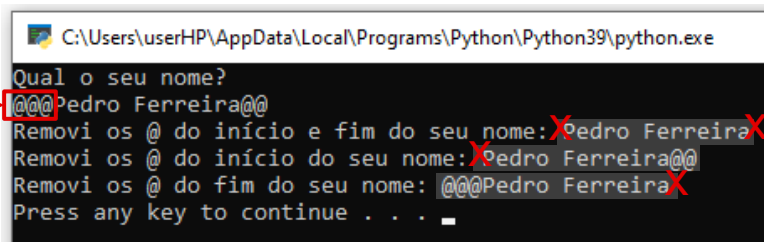
Strings

Remoção de caracteres

Strings - remoção de caracteres

- Quando existem caracteres (um ou vários) que não nos interessam antes ou depois da string, é possível fazer a sua remoção:

```
nome = input ("Qual o seu nome?\n")
print("Removi os @ do início e fim do seu nome:", nome.strip("@"))
print("Removi os @ do início do seu nome:", nome.lstrip("@"))
print("Removi os @ do fim do seu nome:", nome.rstrip("@"))
```



```
C:\Users\userHP\AppData\Local\Programs\Python\Python39\python.exe
Qual o seu nome?
@@@Pedro Ferreira@@
Removi os @ do início e fim do seu nome: Pedro Ferreira
Removi os @ do início do seu nome: Pedro Ferreira@@
Removi os @ do fim do seu nome: @@@Pedro Ferreira
Press any key to continue . . .
```

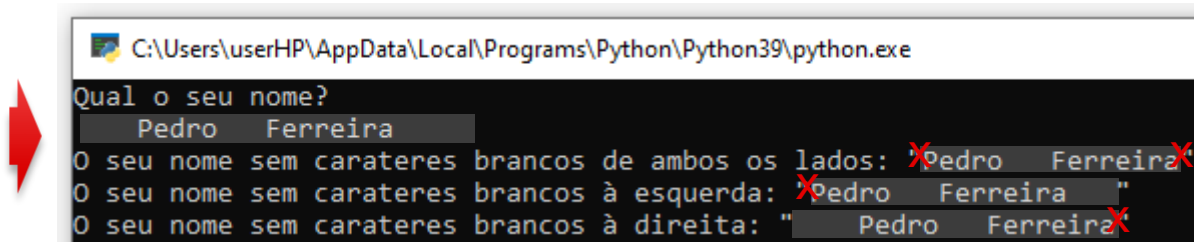
! **Note que:** independentemente da quantidade, serão todos removidos no início ou no fim.

Instrução	Descrição
.strip(x)	Remove todos os caracteres “x” que se encontrem antes e depois do texto
.rstrip(x)	Remove todos os caracteres “x” que se encontrem á direita do texto
.lstrip(x)	Remove todos os caracteres “x” que se encontrem á esquerda do texto

Strings - remoção de caracteres

- Sempre que queiramos remover caracteres brancos, como por exemplo: **tabs, espaços ou quebras de linha (equivalente ao “\n”)**, basta não indicar parâmetros nos strip() correspondente.

```
nome = input ("Qual o seu nome?\n")
print("O seu nome sem caracteres brancos de ambos os lados: \", nome.strip(), "\", sep="")
print("O seu nome sem caracteres brancos à esquerda: \", nome.lstrip(), "\", sep="")
print("O seu nome sem caracteres brancos à direita: \", nome.rstrip(), "\", sep="")
```



```
C:\Users\userHP\AppData\Local\Programs\Python\Python39\python.exe
Qual o seu nome?
Pedro Ferreira
O seu nome sem caracteres brancos de ambos os lados: XPedro FerreiraX
O seu nome sem caracteres brancos à esquerda: XPedro Ferreira
O seu nome sem caracteres brancos à direita: "Pedro FerreiraX
```

Instrução	Descrição
.strip()	Remove caracteres brancos de ambos os lados
.rstrip()	Remove caracteres brancos à direita
.lstrip()	Remove caracteres brancos à esquerda


Strings - remoção de caracteres

- Note que se colocarmos vários caracteres dentro do strip(), rstrip() ou lstrip() não estamos a pedir para remover uma frase no início e/ou no fim da string, mas sim a indicar quais os caracteres que deverão ser considerados para remoção (independentemente da sua ordem ou quantidade).

```
nome = input("Qual o seu nome?\n")  
print("Final:", nome.strip("@#$"))
```



Qualquer um dos caracteres, independentemente da sua ordem ou quantidade, será considerado para limpeza da string.



```
C:\Users\userHP\AppData\Local\Programs\Python\Python38-32\python.exe  
Qual o seu nome?  
@Pedro$  
Final: XpedroX  
Press any key to continue . . .
```

```
C:\Users\userHP\AppData\Local\Programs\Python\Python38-32\python.exe  
Qual o seu nome?  
@@@Pedro$$$$$  
Final: XpedroX  
Press any key to continue . . .
```

```
Selecionar C:\Users\userHP\AppData\Local\Programs\Python\Python38-32\python.exe  
Qual o seu nome?  
@$$$$$$ Pedro $ Ferreira @@$$$$$#  
Final: Xpedro $ FerreiraX  
Press any key to continue . . .
```


Strings - remoção de caracteres

Exercício 4

Peça o preço de um qualquer produto a um utilizador, seguidamente remova-lhe os símbolos £ \$ ou € que possam existir.

Lembre-se que o utilizador poderá sem querer indicar um ou mais espaços antes e/ou depois do seu input - trate também este caso.

Seguidamente aplique um desconto de 10% ao produto e apresente o valor.

► Cábula:

Instrução	Descrição
<code>.strip(x)</code>	Remove todos os caracteres “x” que se encontrem antes e depois do texto
<code>.rstrip(x)</code>	Remove todos os caracteres “x” que se encontrem á direita do texto
<code>.lstrip(x)</code>	Remove todos os caracteres “x” que se encontrem á esquerda do texto

```
nome = input ("Qual o seu nome?\n")  
print("Final:", nome.strip("@#$"))
```

Strings - remoção de caracteres

Exercício 4 (solução)

```
preco = input("Qual o preço do produto?\n")
preco = float(preco.strip(" €$£"))
final = preco * 0.9

print (f"Preço final: {final:.2f}")
```



Além de remove os espaços,
remove os: £, €, \$ em ambos
os lados



```
C:\Users\userHP\AppData\Local\Programs\Python\Python38-32\python.exe
Qual o preço do produto?
12.75€
Preço final: 11.47
```

```
C:\Users\userHP\AppData\Local\Programs\Python\Python38-32\python.exe
Qual o preço do produto?
£ 12.75
Preço final: 11.47
```

Strings

Tamanho

Strings - tamanho

- Sempre que precisemos de avaliar o tamanho de uma string, basta usar a instrução `len(x)`

```
nome = input("Qual o seu nome?\n")  
print("O seu nome tem", len(nome), "carateres")
```



Devolve um inteiro com a quantidade de carateres de uma string.



```
C:\Users\userHP\AppData\Local\Programs\Python\Python38-32\python.exe  
Qual o seu nome?  
Pedro  
O seu nome tem 5 carateres
```

Strings - tamanho

Exercício 5

Peça ao utilizador uma password mas obrigue-o a respeitar o seguinte:

- Deve apenas conter letras ou números
- Deve ter pelo menos 5 caracteres.

Obrigue o utilizador a inserir a password até garantir que cumpre o exigido.

► Cábula:

Instrução	Descrição
.isalpha()	Carateres alfabéticos
.isalnum()	Carateres alfanuméricos
.istitle()	Maiúsculas no inicio das palavras, seguido de minúsculas
.islower()	Apenas minúsculas e números
.isupper()	Apenas maiúsculas e números
.isdigit()	Apenas números

```
nome = input("Qual o seu nome?\n")
if nome.isalpha():
    print("Tudo OK!!")
else:
    print ("Hum? Algo está errado!")
```

```
nome = input("Qual o seu nome?\n")
print("O seu nome tem", len(nome), "carateres")
```

Strings - tamanho

Exercício 5 (solução)

```
pw = input("Indique uma password (Apenas letras ou números com um tamanho mínimo de 5 carateres):\n")

while True:
    if pw.isalnum() and len(pw) > 5:
        print("Registo realizado!")
        break

    pw = input("Erro, Indique de novo:\n")
```



```
C:\Users\userHP\AppData\Local\Programs\Python\Python39\python.exe
Indique uma password (Apenas letras ou números com um tamanho mínimo de 5 carateres):
rick
Erro, Indique de novo:
rick123
Registo realizado!
```

Strings

Fatlar

Strings - Fatiar

- Uma vez que uma string se trata de um conjunto de caracteres, é possível delimitar as partes que queremos:

```
texto[ início : fim : salto ]
```

Alternativas:

```
texto[início]
```

```
texto[início : ]    texto[ : fim]
```

```
texto[início : fim]
```

```
texto[início : : salto]
```

```
texto[ : : salto]
```

```
texto = "Mundo"  
print(texto[1:3])  
print(texto[:6])  
print(texto[4:])  
print(texto[1:-1])  
print(texto[-5])  
print(texto[::-1])  
print(texto[1:-1:2])
```

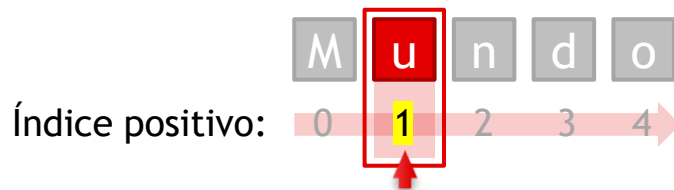


```
C:\User  
un  
Mundo  
o  
und  
M  
odnuM  
ud
```


Strings - Fatiar

- Aceder a uma posição concreta (**índice positivo**):

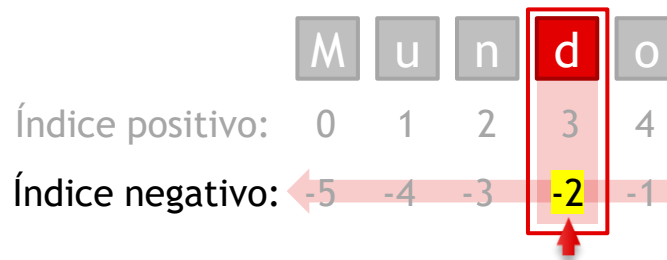
```
texto = "Mundo"  
print(texto[1])
```



Strings - Fatiar

- Aceder a uma posição concreta mas contar do fim (**índice negativo**):

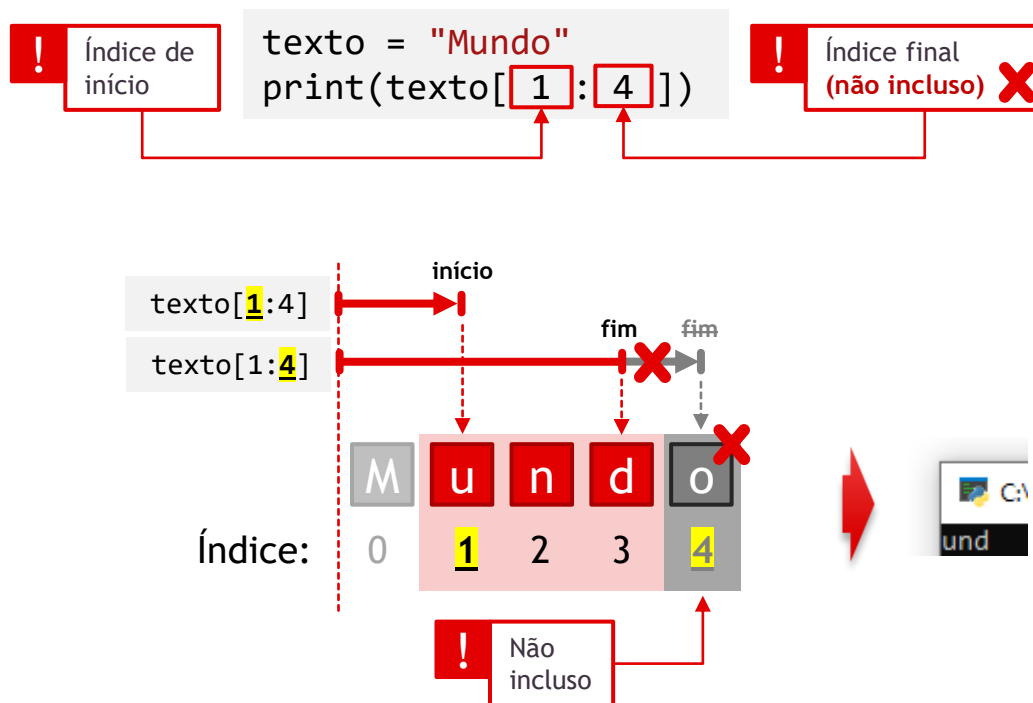
```
texto = "Mundo"  
print(texto[-2])
```



! Os índices negativos
começam a contagem
em "-1"

Strings - Fatiar

- Retirar um trecho de uma determinada zona (com índice de início e de fim).



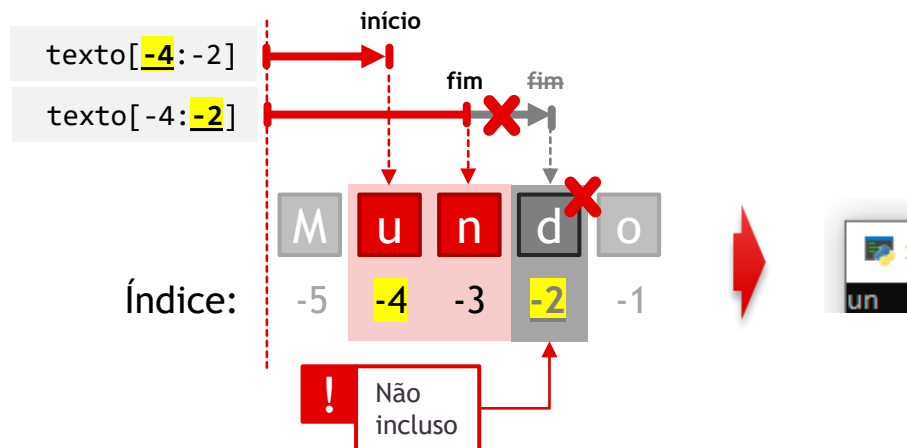
Strings - Fatiar

- Também se podem usar **índices negativos**:

! Índice de início

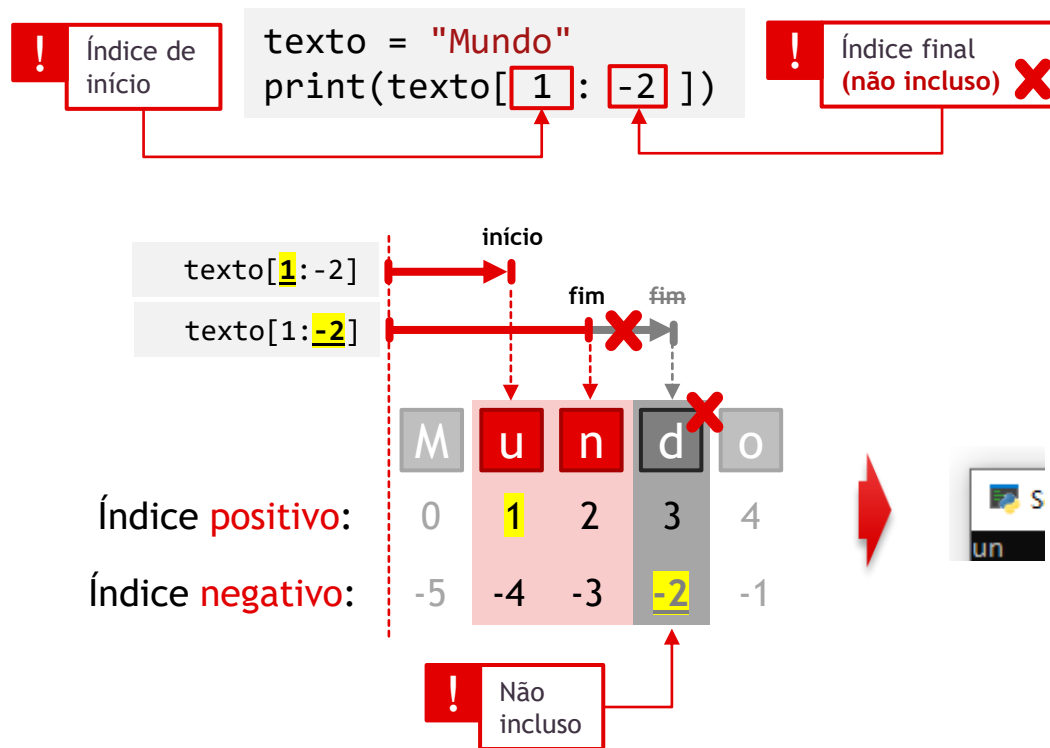
```
texto = "Mundo"  
print(texto[-4 : -2])
```

! Índice final (não incluso) ✗



Strings - Fatiar

- Ou índices negativos em conjunto com positivos:

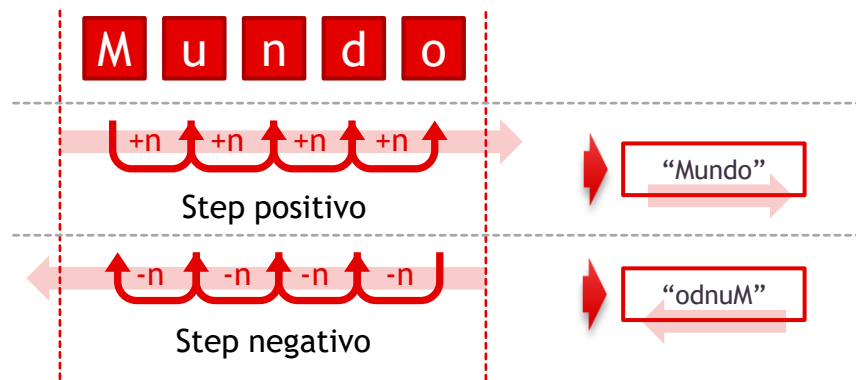


Strings - Fatiar

- O salto (**step**) define a **orientação de como os caracteres serão obtidos.**
- Por defeito, quando não preenchido, o step é positivo (valor “1”).

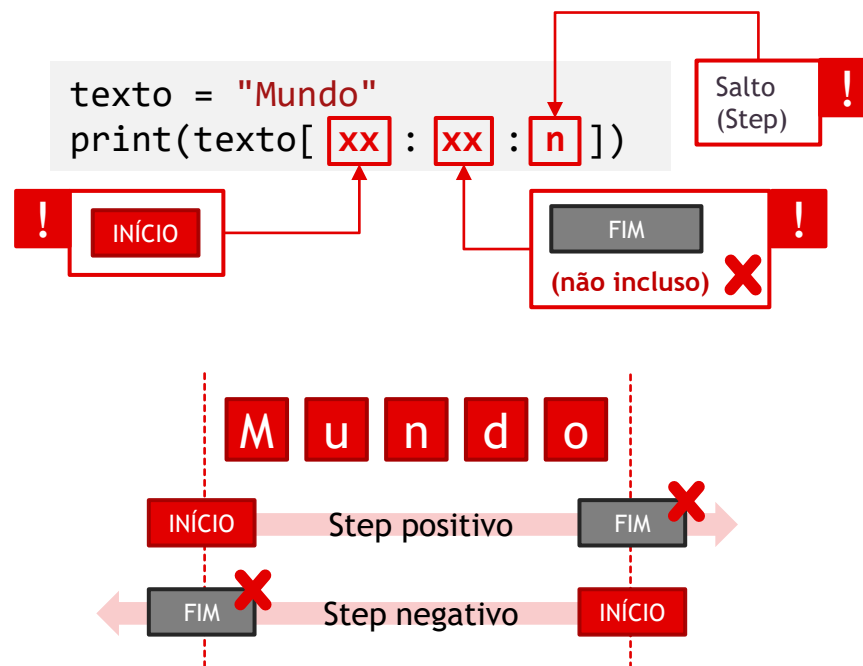
```
texto = "Mundo"  
print(texto[ xx : xx : n ])
```

! Salto (Step)



Strings - Fatiar

- O salto (**step**) ainda **orienta a posição de início e fim** do corte:



Strings - Fatiar

- Exemplo com um **step positivo**:

```
texto = "Mundo"
```

Salto positivo

```
print(texto[ 0 : 3 : 1 ])
```

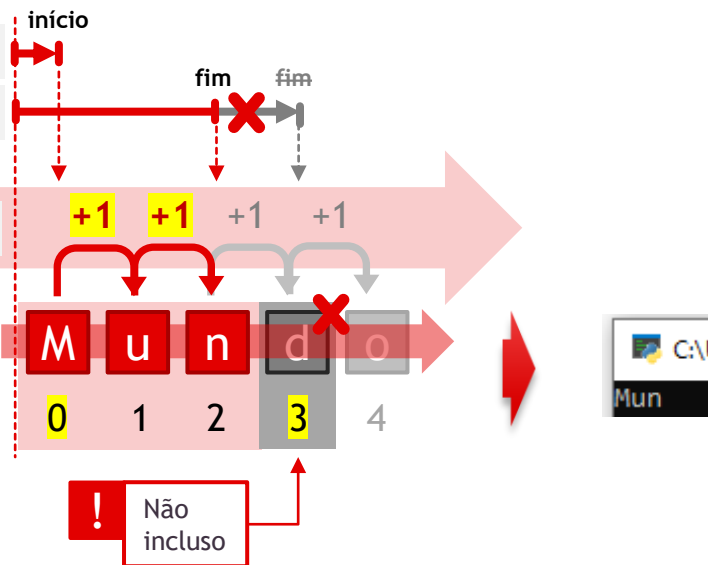
```
print(texto[0:3:1])
```

```
print(texto[0:3:1])
```

Salto positivo

```
print(texto[0:3:1])
```

Índice **positivo**:



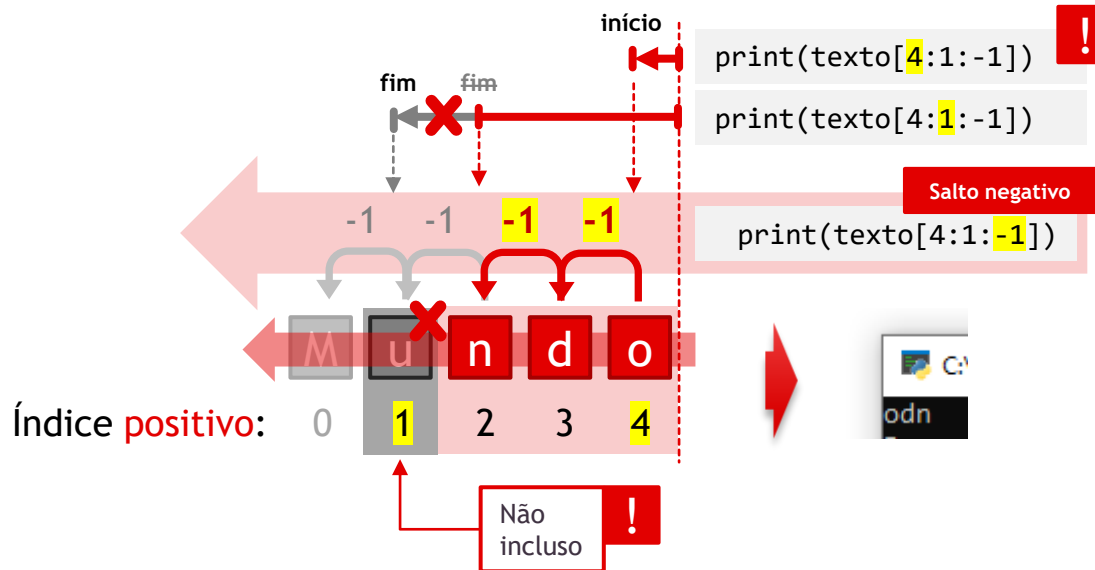
Strings - Fatiar

- Exemplo com um **step negativo**:

```
texto = "Mundo"
```

```
print(texto[ 4 : 1 : -1 ])
```

Salto negativo



! Notar que o primeiro índice a ser selecionado foi o 4, que serve de referência ao início da obtenção do corte.



Strings - Fatiar

- Exemplo com um **step positivo mais extenso**:

```
texto = "Mundo"  
print(texto[ 0 : 3 : 2 ])
```

Salto positivo

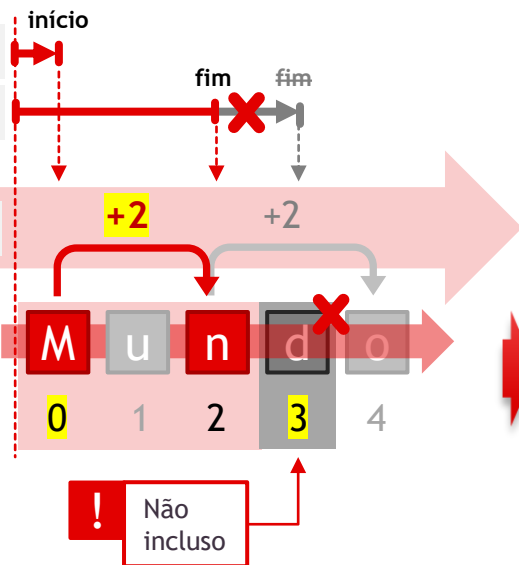
```
print(texto[0:3:2])
```

```
print(texto[0:3:2])
```

Salto positivo

```
print(texto[0:3:2])
```

Índice **positivo**:



Strings - Fatiar

- Exemplo com o **extremo do início**:

```
texto = "Mundo"  
print(texto[ : -2 ])
```

Extremo do início

