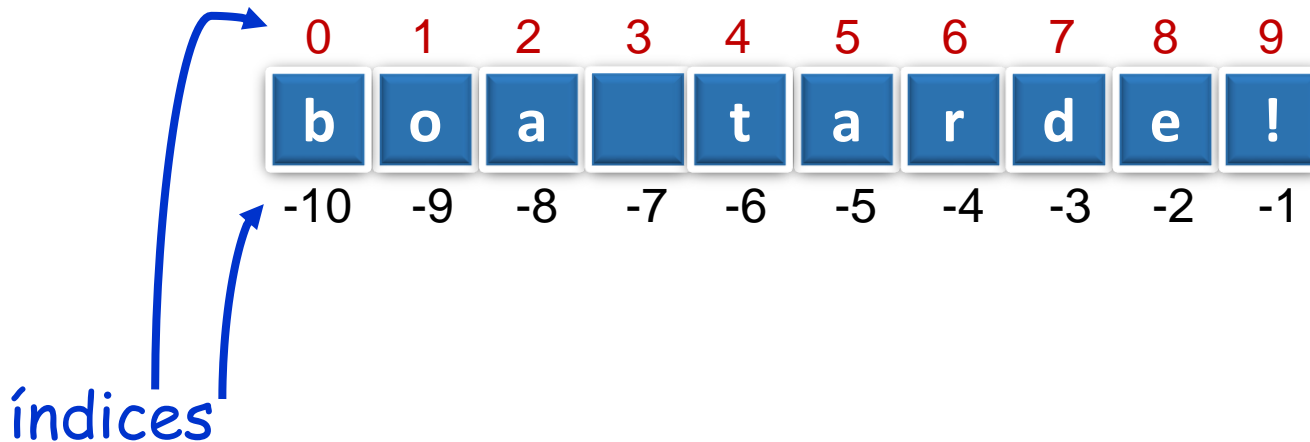


Trabalhando com strings

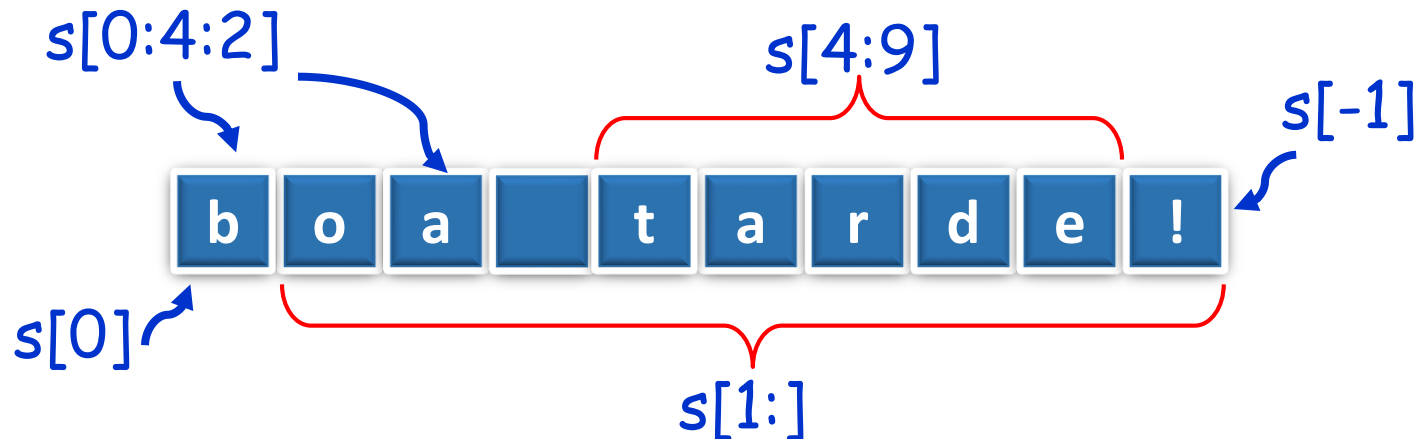


- ✓ Sequência de caracteres
- ✓ Representadas entre aspas simples ou duplas
 - 'boa tarde' ou "boa tarde"
 - '''boa tarde'''
- ✓ Um caractere é uma string de tamanho 1
- ✓ Exemplo



✓ **s[a:b:n]**

- seleciona uma fatia da string da posição **a** (inclusive) até a posição **b** (exclusive) de **n** em **n** caracteres
- se a não definido → considerado como zero
- se b não definido → considerado como tamanho da string
- se n não definido → considerado como 1



Operações com Strings

✓ Concatenação

```
msg = 'boa' + ' tarde!'
print(msg)
```

boa tarde!

✓ Réplica

```
print(3 * msg)
```

boa tarde!boa tarde!boa tarde!

✓ Tamanho

```
print(len(msg))
```

10

Strings são imutáveis

- ✓ Não é possível alterar um caractere de uma string



```
frase = 'Um pato'  
frase[0] = 'f'
```

TypeError: 'str' object does not support item assignment

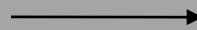
- ✓ Solução
 - criar uma nova string usando fatiamento

```
frase = frase[:3] + 'f' + frase[4:]
```

Comparação de strings

- ✓ Strings são comparadas *lexicograficamente*
 - em "ordem alfabética", caractere a caractere
- ✓ Exemplos

'maria' == 'maria'



m	a	r	i	a
m	a	r	i	a

'maria' != 'matias'



m	a	r	i	a	
m	a	t	i	a	s

'maria' < 'mario'



m	a	r	i	a
m	a	r	i	o

Percorrendo uma string

- ✓ É possível percorrer uma string caractere a caractere usando um laço while
- ✓ Exemplo

```
nome = 'boa tarde!'
ind = 0
limite = len(nome)
while ind < limite:
    print (nome[ind])
    ind = ind + 1
```

b
o
a

t
a
r
d
e
!



- ✓ Escrever uma função chamada **busca** que receba uma string e um caractere e retorne o índice da primeira ocorrência deste caractere na string
 - se não houver um caractere igual na string, a função deve retornar o valor **-1**
- ✓ Exemplos
 - `busca('testes', 's')` retorna 2
 - `busca('testes', 'a')` retorna -1

- ✓ Considerando uma string **s** e um caractere **c**

comparar o 1º caractere de **s** com **c**

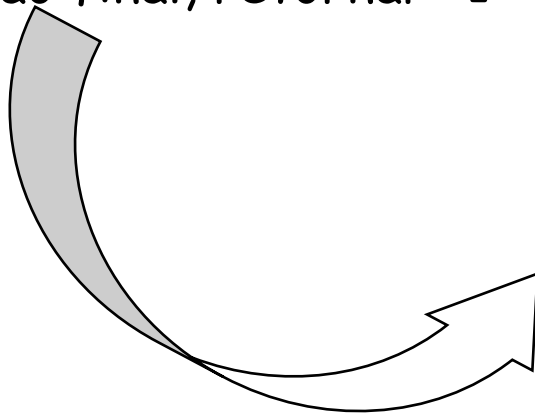
se iguais, retornar índice

comparar o 2º caractere de **s** com **c**

se iguais, retornar índice

e assim até o final de s ...

ao final, retornar -1



enquanto não terminar **s**

comparar um caractere de **s** com **c**

se iguais, retornar índice

ao final, retornar -1

indice = 0

enquanto indice < tamanho da string:

 se $s[\text{indice}] == c$

 retornar indice

 indice += 1

retornar -1

```
def busca (s, c):  
    indice = 0  
    tam = len (s)  
    while indice < tam:  
        if s[indice] == c:  
            return indice  
        indice += 1  
    return -1
```