# Strings

Prof. Alberto Costa Neto

Programação em Python

# Tipo de Dados String

- Uma string é uma seqüência de caracteres
- Uma literal string usa apóstrofos ou aspas para delimitar os caracteres
- Para strings, + significa “concatenar”
- Mesmo que  uma string contenha apenas números, ela continua sendo uma string
- Podemos converter uma string contendo caracteres numéricos em valor numérico usando int() ou float()

```
>>> str1 = "Bom"
>>> str2 = 'dia'
>>> bom = str1 + str2
>>> print(bom)
Bomdia
>>> str3 = '123'
>>> str3 = str3 + 1
Traceback (most recent call last):
  File "<stdin>", line 1, in
<module>
TypeError: must be str, not int
```

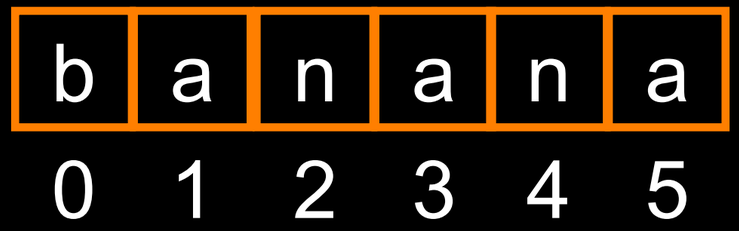
# Lendo e Convertendo

- Preferimos ler dados usando strings e então analisar e converter os dados conforme seja necessário
- Isto nos dá mais controle sobre situações de erro e/ou entrada do usuário inválida
- Caracteres numéricos na entrada devem ser convertidos a partir das strings

```
>>> nome = input('Nome:')
Nome:Alberto
>>> print(nome)
Alberto
>>> x = input('Numero:')
Numero:100
>>> x = x + 1
Traceback (most recent call
last):
  File "<stdin>", line 1, in
<module>
TypeError: must be str, not int
>>> num = int(x) + 1
>>> print(num, x)
101 100
>>>
```

# Detalhes Internos de Strings

- Podemos acessar individualmente qualquer caractere em uma String usando o índice entre colchetes
- O valor do índice deve ser um inteiro e inicia-se em 0 (zero)
- O valor do índice pode ser obtido de uma expressão numérica




```
>>> fruta = 'banana'
>>> letra = fruta[1]
>>> print(letra)
a
>>> n = 3
>>> w = fruta[n - 1]
>>> print(w)
n
```

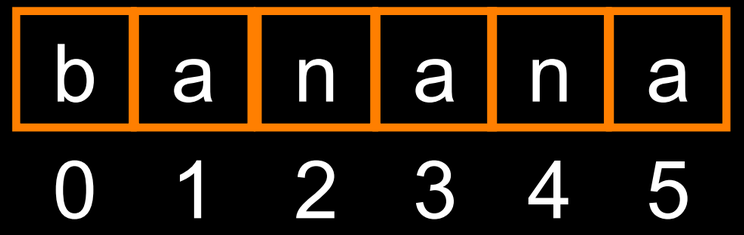
# Não passe dos limites!

- Caso tente acessar um índice além do final da String, você irá obter erro.
- Então, tenha cuidado ao calcular os índices e construir fatias (slices)

```
>>> s = 'abc'
>>> print(s[5])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: string index out of range
>>>
```

# Strings têm comprimento (Length)

- Existe uma função built-in len que nos dá o comprimento de uma string



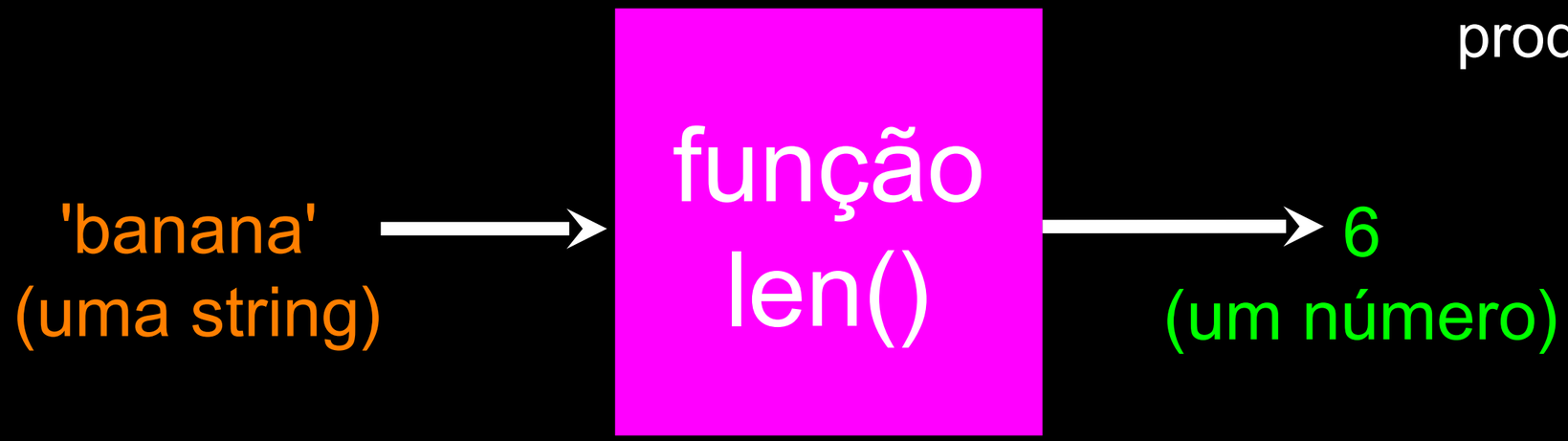

```
>>> fruta = 'banana'
>>> print(len(fruta))
6
```

# Função len

```
>>> fruta = 'banana'
>>> x = len(fruta)
>>> print(x)
6
```

Uma função é um bloco de código fonte armazenado que usamos.

Uma função recebe uma entrada e produz uma saída.



# Laço While e Strings

- Podemos usar o comando while (com uma variável de iteração) e a função len para criar um laço para acessar cada caractere de uma String individualmente

```
fruta = 'banana'
indice = 0
while indice < len(fruta):
    letra = fruta[indice]
    print(indice, letra)
    indice = indice + 1
```

```
0 b
1 a
2 n
3 a
4 n
5 a
```

# Usando in como um operador lógico

- A palavra chave in pode ser usada para checar se uma string está dentro (in) de outra string
- in é uma expressão lógica (retorna True ou False)
- Pode ser usado em um comando if

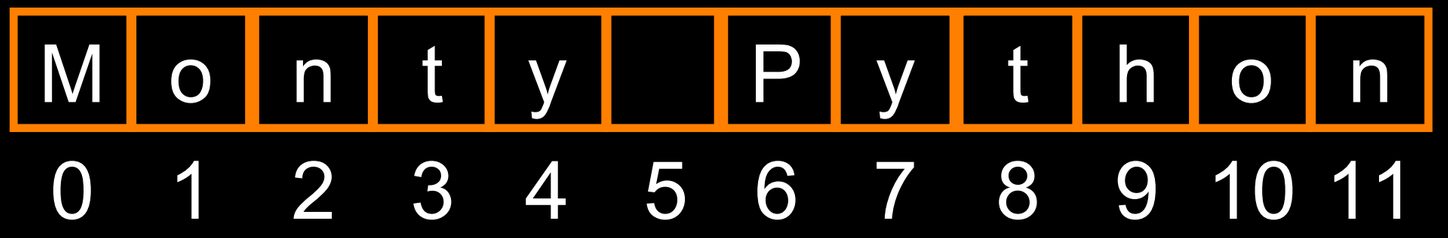
```
>>> fruta = 'banana'
>>> 'n' in fruta
True
>>> 'm' in fruta
False
>>> 'nan' in fruta
True
>>> if 'a' in fruta :
...     print('Encontrado!')
...
Encontrado!
>>>
```

# Comparação de Strings

```
palavra = input()
if palavra == 'banana':
    print('Ok, bananas.')

if palavra < 'banana':
    print('A palavra,' + palavra + ', vem antes de banana.')
elif palavra > 'banana':
    print('A palavra,' + palavra + ', vem depois de banana.')
else:
    print('Ok, bananas.')
```

# Slicing (particionando) Strings




- Podemos obter pedaços (slices ou substrings) de uma String usando o operador colon ':' (dois pontos)
- O segundo número é o índice do final da substring. ATENÇÃO: Observe que ele não fará parte da substring!
- Se ele for maior que o final da string, a substring termina no último caractere

```
>>> s = 'Monty Python'
>>> print( s[0:4] )
Mont
>>> print( s[0:5] )
Monty
>>> print( s[6:7] )
P
>>> print( s[6:20] )
Python
```