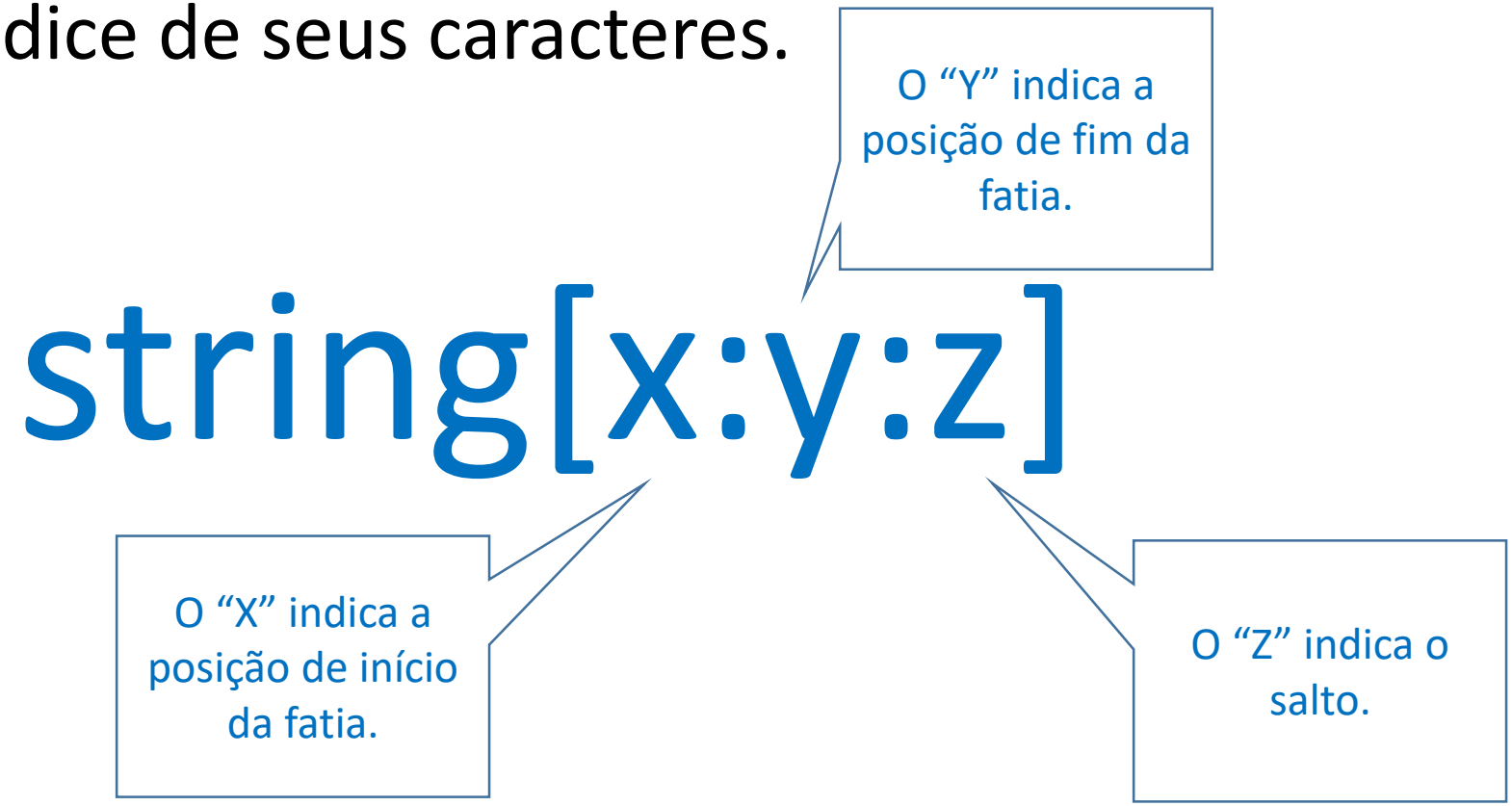
Variáveis e tipos de dados

Aprendendo mais sobre strings

Aprendendo mais sobre strings

Fatiamento

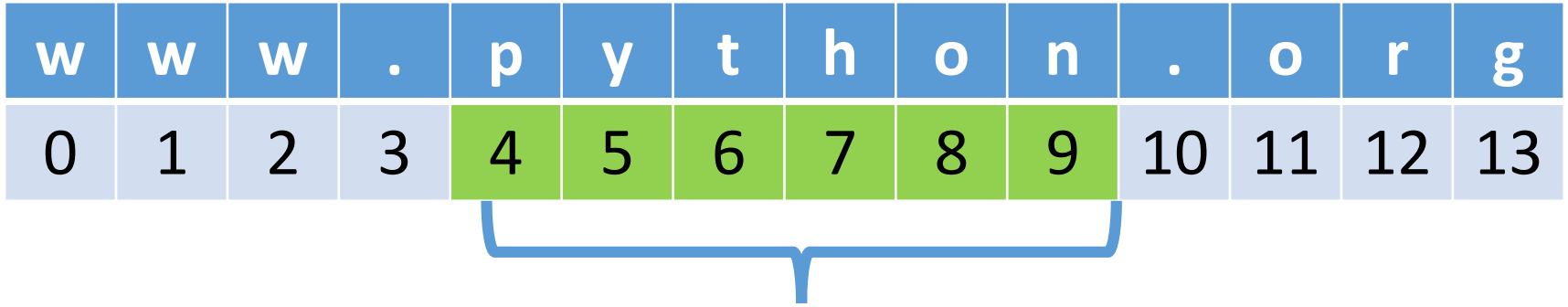
Fatiamento (slicing) é o recurso de extrair partes de uma string utilizando o índice de seus caracteres.



Aprendendo mais sobre strings

Fatiamento

No fatiamento a posição inicial (x) é incluída no resultado, porém, a posição final (y) não é incluída.



Neste exemplo será impressa a string "python".

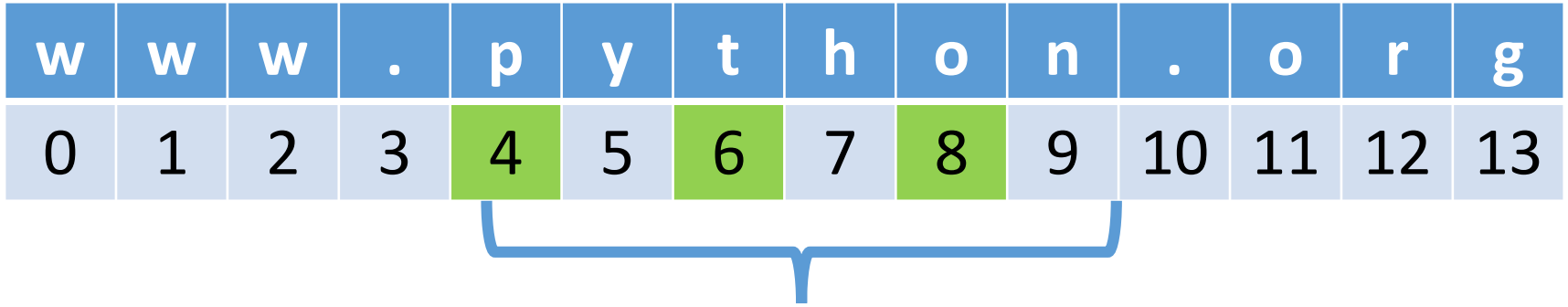
```
endereco = "www.python.org"
fatia = endereco[4:10]
print(fatia)
```

Se o salto não for informado, será considerado igual a um. O que fará com que retorne o conteúdo da posição inicial até a posição final, caractere por caractere.

Aprendendo mais sobre strings

Fatiamento

Vamos repetir o exemplo, adicionando um salto de 2.



Vamos agora fatiar usando um salto igual a 2. O resultado será a string “pto”.

```
endereco = "www.python.org"
fatia = endereco[4:10:2]
print(fatia)
```

Aprendendo mais sobre strings

Fatiamento

Se os índices de limite inicial e final não forem informados, é realizada apenas uma cópia da string completa.

w	w	w	.	p	y	t	h	o	n	.	o	r	g
0	1	2	3	4	5	6	7	8	9	10	11	12	13

Nesse exemplo não foi informado o valor de salto, sendo assim, serão retornados todos caracteres, um a um.
Será impressa a string completa:
www.python.org

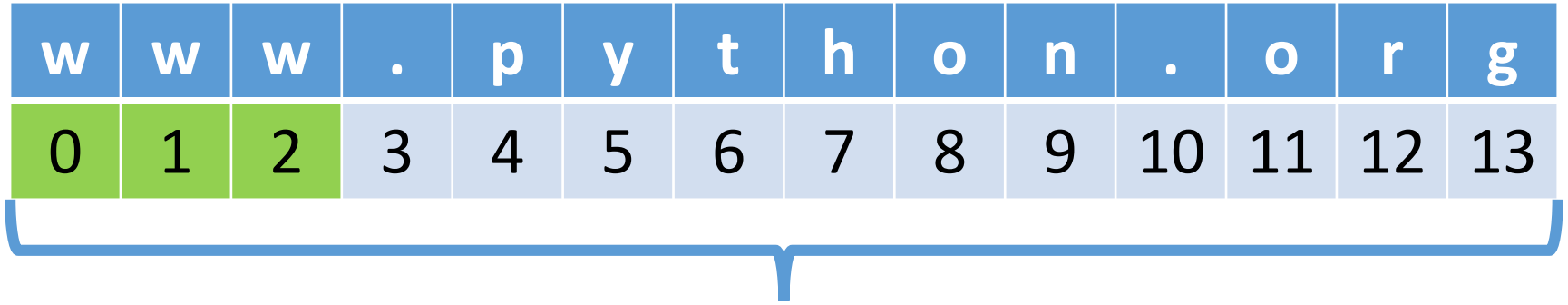
```
endereco = "www.python.org"
fatia = endereco[:]
print(fatia)
```

Se modificarmos o exemplo colocando um salto de -1, será retornada a string completa, caracter por caracter, porém, em ordem invertida.
gro.nohtyp.www

Aprendendo mais sobre strings

Fatiamento

Se for omitido o índice inicial será considerado à partir do primeiro caractere.



Nesse exemplo será impressa a string “www”.

```
endereco = "www.python.org"
fatia = endereco[:3]
print(fatia)
```

Aprendendo mais sobre strings

Fatiamento

Se for omitido o índice final será considerado até o último caractere (incluindo o último caractere).

w	w	w	.	p	y	t	h	o	n	.	o	r	g
0	1	2	3	4	5	6	7	8	9	10	11	12	13

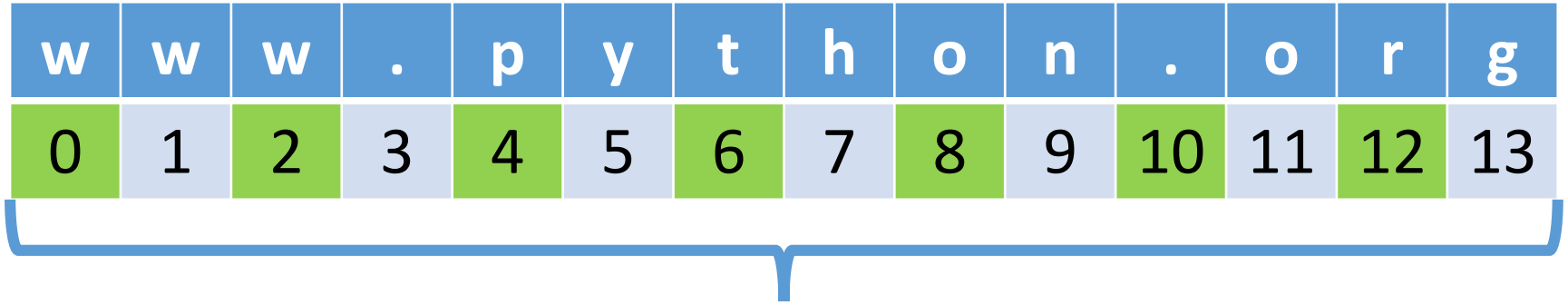
Nesse exemplo será impressa a string “org”.

```
endereco = "www.python.org"
fatia = endereco[11:]
print(fatia)
```

Aprendendo mais sobre strings

Fatiamento

Agora vamos retornar a string completa, porém, com um salto igual a 2.



Nesse exemplo será impressa a string “wwpto.r”

```
endereco = "www.python.org"
fatia = endereco[::2]
print(fatia)
```

Aprendendo mais sobre strings

Fatiamento

É possível utilizar um índice negativo para trabalharmos com o conteúdo da string de trás pra frente. O último caractere é o -1.

w	w	w	.	p	y	t	h	o	n	.	o	r	g
-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

Nesse exemplo será impressa a string “python”

```
endereco = "www.python.org"
fatia = endereco[-10:-4]
print(fatia)
```

Aprendendo mais sobre strings

Index

O método *index* pode ser utilizado para retornar o índice da primeira ocorrência do elemento encontrado em uma string.

Sintaxe:

string.index(“a”)

Será retornado o índice da primeira ocorrência do caractere “a” na string.

string.index(“Python”) # Texto

Será retornado o índice da primeira ocorrência do texto “Python” na string.

Aprendendo mais sobre strings

Index

Exemplo prático 1:

Obtendo o usuário de um endereço de e-mail.

Como todo endereço de e-mail possui um arroba após o nome do usuário, podemos identificar a posição do caractere arroba dentro do endereço de e-mail usando o método “index”.

Veja:

string.index(“@”)

Sabendo a posição do caractere “@” podemos usar fatiamento para retornar o usuário.

Aprendendo mais sobre strings

Index

Veja como obter o índice da parte da *string* que nos interessa.

e	v	a	l	d	o	w	o	l	k	e	r	s	@	g	m	a	i	l	.	c	o	m
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22

```
email = "evaldowolkers@gmail.com"
indice = email.index(".com")
print(indice) # impresso 19
indice = email.index("@gmail")
print(indice) # impresso 13
indice = email.index("@")
print(indice) # impresso 13
```

Aprendendo mais sobre strings

Index

Veja agora o programa exemplo:

```
email = "evaldowolkers@gmail.com"
indice_arroba = email.index("@")
usuario = email[0:indice_arroba]
print("O nome de usuário é:", usuario)
```

Resultado:
O nome de usuário é: evaldowolkers

Aprendendo mais sobre strings

Index

Exemplo prático 2: Obtendo o provedor de um endereço de e-mail.

O provedor é a informação que fica entre o arroba e um ponto.

Exemplo:

fulano@**gmail**.com (o provedor é o **gmail**)

Aprendendo mais sobre strings

Index

Foi incrementado 1 para não pegar o arroba.

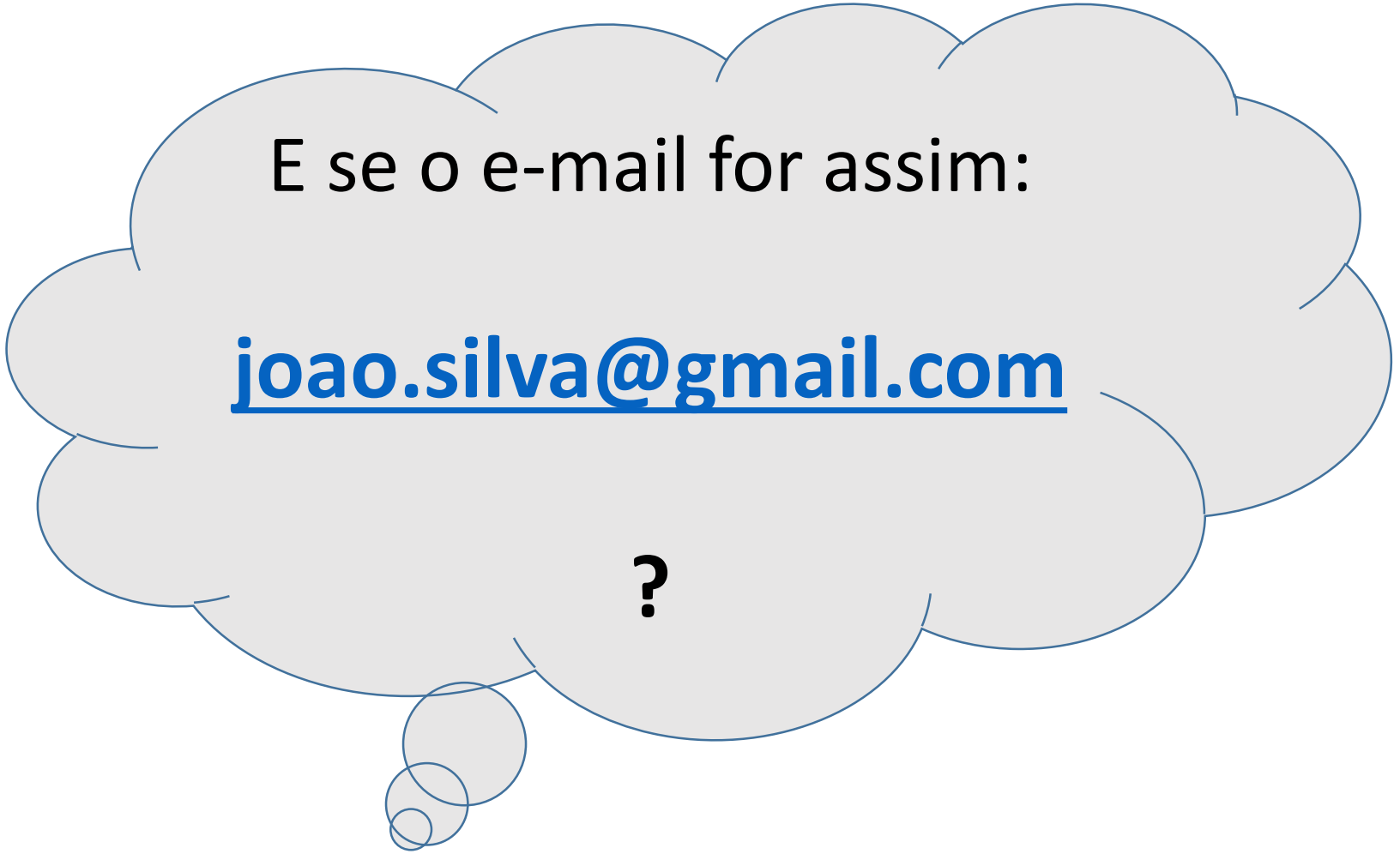
Veja agora o programa exemplo:

```
email = "evaldowolkers@gmail.com"
indice_arroba = email.index("@")
indice_ponto = email.index(".")
provedor = email[indice_arroba+1:indice_ponto]
print("O nome do provedor é:", provedor)
```

Resultado:
O nome do provedor é: gmail

Aprendendo mais sobre strings

Index



Aprendendo mais sobre strings

Index

Veja agora o programa novamente:

```
email = "joao.silva@gmail.com"
indice_arroba = email.index("@")
indice_ponto = email.index(".")
provedor = email[indice_arroba+1:indice_ponto]
print("O nome do provedor é:", provedor)
```

Resultado:
O nome do provedor é:

Nada foi impresso porque o índice final ficou menor que o índice inicial. email[10+1:4]
Vamos ver como resolver isso.

Aprendendo mais sobre strings

Split

Vamos agora aprender um novo método para "quebrar" uma string à partir de um determinado caractere. No nosso exemplo, à partir do "@".

Veja:

evaldowolkers@gmail.com

evaldowolkers gmail.com

Aprendendo mais sobre strings

Split

Para esta tarefa podemos usar o método *split*.

O método *split* retorna uma **lista** de palavras usando um separador informado por parâmetro.

Obs.: Em aulas futuras irei falar sobre listas em *Python*.

Nesta aula, entenda apenas que o e-mail será quebrado em duas informações à partir do “@”, sendo “joao.silva” e “gmail.com”.

Aprendendo mais sobre strings

Split

Sintaxe:

string.split(“separador”)

No nosso caso usaremos assim:

“joao.silva@gmail.com”.split(“@”)

Serão retornadas duas strings, uma contendo “joao.silva” e a outra contendo “gmail.com”.

Aprendendo mais sobre strings

Split

```
email = "joao.silva@gmail.com"
usuario, dominio = email.split("@")
print("Usuário:", usuario)
print("Domínio:", dominio)
```

```
Usuário: joao.silva
Domínio: gmail.com
```

Aprendendo mais sobre strings

Split

Agora que temos o domínio separado podemos procurar o "." para indentificá-lo e separar o provedor.

```
email = "joao.silva@gmail.com"
usuario, dominio = email.split("@")
indice_ponto = dominio.index(".")
provedor = dominio[0:indice_ponto]
print("O usuário é:", usuario)
print("O domínio é:", dominio)
```

```
O usuário é: joao.silva
O domínio é: gmail.com
```

FIM