Strings, Loops

CPE101
Winter 2019
@ Cal Poly SLO
By
Toshi

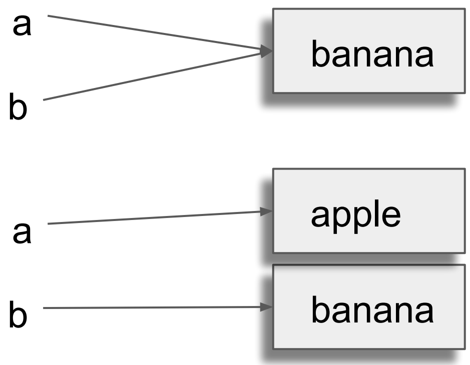
Learning Objectives

1. Strings
    a. Idea and basic operations
    b. Not arrays and immutable
    c. Build new strings to "change"

Strings

- A string is data with one or multiple characters strung together.
  - Each character can be any characters including numbers, special characters, etc.
- A string is immutable
  - You can not change the content of a string.
  - If you try, the computer will just create a new string with updated content and point the string variable to the new string in memory.

```
>>> a = 'banana'
>>> b = 'banana'
>>> a is b
True
>>> a = 'apple'
>>> a is b
False
```



Python String Operations

- Substring
    - greeting = “Hello”
    - print greeting[0] # H
    - print greeting[1:] # ello
- Concatenation
    - greeting = “Hello”
    - greeting += “ world!” # a new string created
    - print greeting # Hello World!

Strings and Arrays (Lists)

- Python Strings and Arrays are different, although they have some similarities.
    - Python strings (str) and arrays (list) are both objects but belong to different classes.

```
>>> a = ['h','e','l','l','o','!']
>>> b = 'hello!'
>>> type(a)
<type 'list'>
>>> type(b)
<type 'str'>
>>> a is b
False
```

```
>>> len(a)
6
>>> len(b)
6
>>> a[0]
'h'
>>> b[0]
'h'
```

Iterating over elements of list / string

- While loop
  - You are responsible for
    - specifying the termination condition
    - (incrementing the index to access the next element)
- For loop
  - In Python, you will automatically get each element in a list in each iteration of the loop.
  - for item in collection:
    - Item will contain each element starting from the left end of the collection (list / string) to the end in each iteration.

Iterating over elements in list / string

```
letters = “abcdef”

for  letter in letters:
    print(letter)
```

```
a
b
c
d
e
f
```

Iterating over elements in list / string

```
letters = “abcdef”
i = 0
size = len(letters)
while i < size:
    print(letters[i])
    i += 1
```

```
a
b
c
d
e
f
```

Iterating over elements in list / string

```
letters = [‘a’,’b’,’c’,’d’,’e’,’f’]

for  letter in letters:
    print(letter)
```

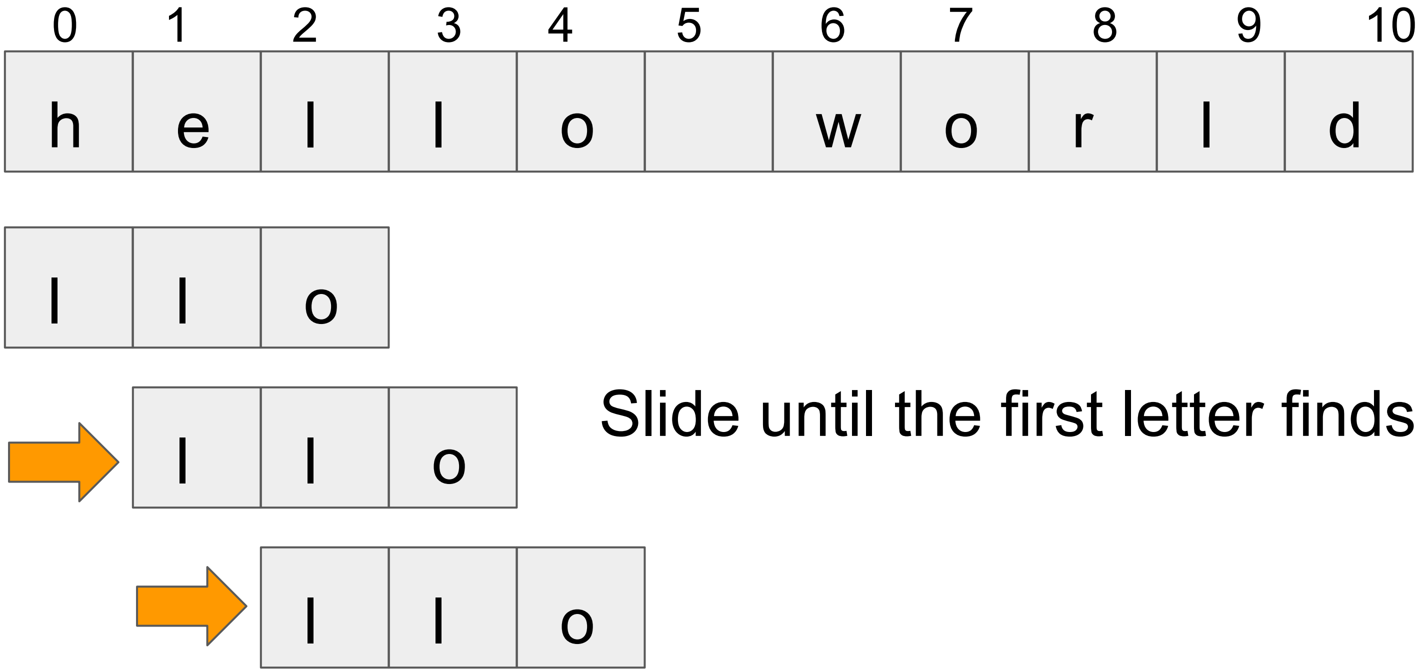
```
a
b
c
d
e
f
```

Iterating over elements in list / string

```
letters = [‘a’,’b’,’c’,’d’,’e’,’f’]
i = 0
size = len(letters)
while i < size:
    print(letters[i])
    i += 1
```

```
a
b
c
d
e
f
```

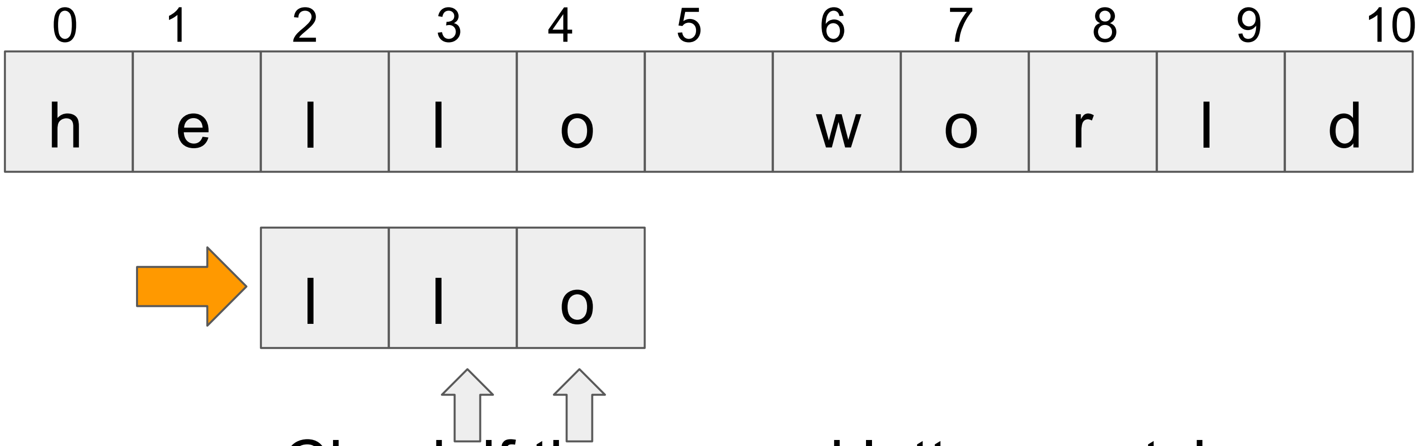
Function find()



0	1	2	3	4	5	6	7	8	9	10
h	e	l	l	o		w	o	r	l	d
l	l	o								
	l	l	o							
		l	l	o						

Slide until the first letter finds its match.

Function find()

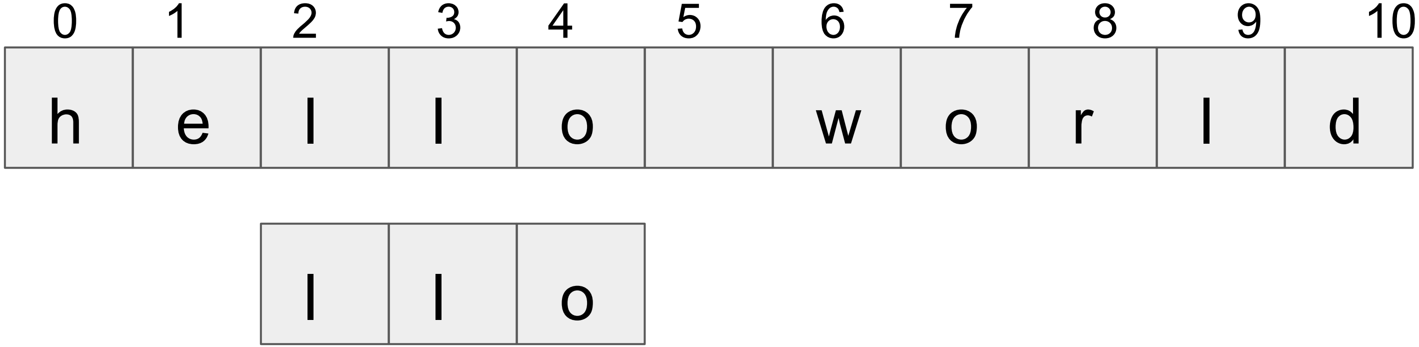


Check if the second letters match.

If they match, check the third letters.

If not, slide the string to right until finding a match for the first letter.

Function find()



0	1	2	3	4	5	6	7	8	9	10
h	e	l	l	o		w	o	r	l	d
		l	l	o						

In this example, a match is found at index 2.

So, we return 2.

```
def find(string, target):
    """search for the target string in the string.
    Args:
        string (str): string
        target (str): the target string
    Returns:
        int or None: the index in the string where the target is found.
                     If not found, None is returned.
    """
    str_len = len(string)
    target_len = len(target)
    idx1 = 0
    while idx1 < str_len:
        idx2 = 0
        while idx2 < target_len:
            if string[idx1 + idx2] != target[idx2]:
                break
            idx2 += 1
        if idx2 == target_len:
            return idx1
        idx1 += 1
    return None
```