Hello, scientists!

Would scientist use python to do your own academic research?

Prof. Iury Adones Xavier dos Santos

Rural Federal University of Pernambuco

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **List**

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **List**
  - Is a collection which is **ordered** and **changeable**.

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **List**
  - Is a collection which is **ordered** and **changeable**.
  - **Allows duplicate** members.

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **List**
  - Is a collection which is **ordered** and **changeable**.
  - **Allows duplicate** members.
- **Tuple**

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **List**
  - Is a collection which is **ordered** and **changeable**.
  - **Allows duplicate** members.
- **Tuple**
  - Is a collection which is **ordered** and **unchangeable**.

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **List**
  - Is a collection which is **ordered** and **changeable**.
  - **Allows duplicate** members.
- **Tuple**
  - Is a collection which is **ordered** and **unchangeable**.
  - **Allows duplicate** members.

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **Set**

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **Set**
  - Is a collection which is **unordered** and **unindexed**.

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **Set**
  - Is a collection which is **unordered** and **unindexed**.
  - **No duplicate** members.

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **Set**
  - Is a collection which is **unordered** and **unindexed**.
  - **No duplicate** members.
- **Dictionary**

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **Set**
  - Is a collection which is **unordered** and **unindexed**.
  - **No duplicate** members.
- **Dictionary**
  - Is a collection which is **unordered**, **changeable** and **indexed**.

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **Set**
  - Is a collection which is **unordered** and **unindexed**.
  - **No duplicate** members.
- **Dictionary**
  - Is a collection which is **unordered**, **changeable** and **indexed**.
  - **No duplicate** members.

About List

```
list()
```

List

Create a List:

```
1 this_list = ["apple", "banana", "cherry"]
2 print(this_list)
```

Access Items

Print the second item of the list:

```
1 this_list = ["apple", "banana", "cherry"]
2 print(this_list[1])
```

Change Item Value

Change the second item:

```
1 this_list = ["apple", "banana", "cherry"]
2 this_list[1] = "blackcurrant"
3 print(this_list)
```

List Length

Print the number of items in the list:

```
1 this_list = ["apple", "banana", "cherry"]
2 print(len(this_list))
```

Add Items

Using the **append()** method to append an item:

```
1 this_list = ["apple", "banana", "cherry"]
2 this_list.append("orange")
3 print(this_list)
```

Add Items

Insert an item as the second position:

```
1 this_list = ["apple", "banana", "cherry"]
2 this_list.insert(1, "orange")
3 print(this_list)
```

Remove Item

The textbfremove() method removes the specified item:

```
1 this_list = ["apple", "banana", "cherry"]
2 this_list.remove("banana")
3 print(this_list)
```

Remove Item

The **pop()** method removes the specified index, (or the last item if index is not specified):

```
1 this_list = ["apple", "banana", "cherry"]
2 this_list.pop()
3 print(this_list)
```

Remove Item

The **del** keyword removes the specified index:

```
1 this_list = ["apple", "banana", "cherry"]
2 del this_list[0]
3 print(this_list)
```

Remove Item

The **del** keyword can also delete the list completely:

```
1 this_list = ["apple", "banana", "cherry"]
2 del this_list
3 print(this_list)
```

Remove Item

The clear() method empties the list:

```
1 this_list = ["apple", "banana", "cherry"]
2 this_list.clear()
3 print(this_list)
```

The list() Constructor

Using the list() constructor to make a List:

```
1 this_list = list(("apple", "banana", "cherry"))
2 print(this_list)
```

Note the double round-brackets

List Methods

Python has a set of built-in methods that you can use on lists.

- How to get all the methods of **list()**?

List Methods

Python has a set of built-in methods that you can use on lists.

- How to get all the methods of **list()**?
- With **dir()**

List Methods

Use dir():

```
1 list_all_methods = dir(list)
2 print(list_all_methods)
```

List Methods

Look up!

```
1 list.append(self, value)
2 list.clear(self)
3 list.insert(self, index, value)
4 list.pop(self, index=-1)
5 list.remove(self, value)
```

List Methods

More on Lists

```
1 list.copy(self)
2 list.count(self, value)
3 list.extend(self, iterable)
4 list.index(self, value, start=0, stop=len(self))
5 list.reverse(self)
6 list.sort(self, key=None, reverse=False)
```

Method copy()

Copy

```
 1 fruits = ['orange', 'apple', 'pear',
 2           'banana', 'kiwi', 'apple',
 3           'banana']
 4
 5 a_fruits = fruits
 6 b_fruits = fruits.copy()
 7
 8 id_a = id(a_fruits)
 9 id_b = id(b_fruits)
10 id_c = id(fruits)
11
12 print(id_a == id_c)
13 print(id_b == id_c)
```

Methods count() and index()

Count and Index

```
1 fruits = ['orange', 'apple', 'pear',
2           'banana', 'kiwi', 'apple',
3           'banana']
4
5 print(fruits.count('apple'))
6 print(fruits.count('tangerine'))
7
8 print(fruits.index('banana'))
9 print(fruits.index('banana', 4))
```

Methods reverse() and sort()

Copy

```
1  fruits = ['orange', 'apple', 'pear',
2            'banana', 'kiwi', 'apple',
3            'banana']
4
5  fruits.reverse()
6  print(fruits)
7
8  fruits.sort()
9  print(fruits)
10
11 fruits.sort(reverse=True)
12 print(fruits)
```

Methods sort()

Sort

```
1 fruits = ['orange', 'apple', 'pear',
2           'banana', 'kiwi', 'apple',
3           'banana']
4
5 fruits.sort(key=lambda f: f[-1])
6 print(fruits)
7
8 fruits.sort(key=lambda f: f[2])
9 print(fruits)
```

Methods extend()

Extend

```
1  fruits = ['orange', 'apple', 'pear']
2  add_fruits = ['banana', 'kiwi', 'apple', 'banana']
3
4  fruits.append(add_fruits)
5  print(fruits)
6
7  fruits.pop()
8
9  fruits.extend(add_fruits)
10 print(fruits)
```

Methods built-ins max() and min()

Max and Min

```
1 points = [5.3, 6, 7. 8, 10]
2
3 max_point = max(points)
4 min_point = min(points)
5
6 print(f"Max: {max_point}")
7 print(f"Min: {min_point}")
```

Loop Through a List

Print all items in the list, one by one:

```
1 this_list = ["apple", "banana", "cherry"]
2 for x in this_list:
3     print(x)
```

Pratice

Make a Program that asks for the ***4 bimonthly notes*** and ***adds to a list***, then calculate and print to show the information of **mean**, **median**, **standard deviation**, **maximum note** and **minimum note**.

Mean: $\bar{x} = \frac{1}{n}\sum_{i=0}^{n-1} x_i$

Standard deviation:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1}(x_i - \bar{x})^2}$$

Standard deviation relative to sample:

$$s = \sqrt{\frac{1}{n-1}\sum_{i=0}^{n-1}(x_i - \bar{x})^2}$$

About tuple

```
tuple()
```

Tuple

Create a Tuple:

```
1 this_tuple = ("apple", "banana", "cherry")
2 print(this_tuple)
```

Access Tuple Items

Return the item in position 1:

```
1 this_tuple = ("apple", "banana", "cherry")
2 print(this_tuple[1])
```

Tuple Length

Print the number of items in the tuple:

```
1 this_tuple = ("apple", "banana", "cherry")
2 print(len(this_tuple))
```

Remove Items

The del keyword can delete the tuple completely:

```
1 this_tuple = ("apple", "banana", "cherry")
2 del this_tuple
```

The tuple() Constructor

Using the tuple() method to make a tuple:

```
1 this_tuple = tuple(("apple", "banana", "cherry"))
2 print(this_tuple)
```

Note the double round-brackets

Tuple Methods

Python has a set of built-in methods that you can use on tuples.

- How to get all the methods of **tuple()**?

Tuple Methods

Python has a set of built-in methods that you can use on tuples.

- How to get all the methods of **tuple()**?
- With **dir()**

Tuple Methods

Use dir():

```
1 tuple_all_methods = dir(tuple)
2 print(tuple_all_methods)
```

Tuple Methods

More on tuples

```
1 tuple.count(self, value)
2 tuple.index(self, value, start=0, stop=len(self))
```

The end

Thant’s all Folks!