量化金融 python 基础 2-List

2023 年 5 月 15 日

1 Data Structure 数据结构 - List

列表是一种数据类型，用于存储一组有序的元素。列表使用方括号 [ ] 表示，其中的元素可以是任何类型的数据，包括数字、字符串、布尔值、甚至其他列表。
列表是可以改变的。

[1]:
```
# 列表用 [ ] 表示，列表中的元素以逗号，相隔。
fruits = ["apple", "banana", "cherry", "grape", "pear"]
fruits
```

[1]:
```
['apple', 'banana', 'cherry', 'grape', 'pear']
```

[2]:
```
# 列表索引 Indexing
# 索引从 0 开始，而不是从 1 开始
fruits[0]
```

[2]:
```
'apple'
```

[3]:
```
# 访问最后一个元素可以用-1
fruits[-1]
```

[3]:
```
'pear'
```

[4]:
```
# 列表切片 Slice
# 访问第一个到第三个元素，但是不包含第三个（含头不含尾）。
# 请注意，fruits[2] 表示的是第三个元素。
fruits[0:2]
```

[4]:
```
['apple', 'banana']
```

[5]:
```
# 访问列表中到最后一个元素，但是不包含最后一个。
fruits[:-1]
```

[5]:
```
['apple', 'banana', 'cherry', 'grape']
```

[6]:
```
# list[start:stop:step]
# step 是步进值，表示每隔 X 个，取一个值
fruits[::2]
```

[6]:
```
['apple', 'cherry', 'pear']
```

[7]:
```
# list.append 方法 向列表末尾添加元素
fruits.append("watermelon")
fruits
```

[7]:
```
['apple', 'banana', 'cherry', 'grape', 'pear', 'watermelon']
```

[8]:
```
# list.extend() 向列表添加多个元素，也可以说是向列表添加列表
fruits.extend([2, 5, 7, 213, 54, 235, 63, 84])
fruits
```

[8]:
```
['apple',
 'banana',
 'cherry',
 'grape',
 'pear',
 'watermelon',
 2,
 5,
 7,
 213,
 54,
 235,
 63,
 84]
```

list.append() 和 list.extend() 主要区别在于前者向列表添加单个元素，后者向列表添加多个元素（或者列表）

[9]:
```
# list.insert() 向列表指定位置添加元素
# 在第二个位置上插入"orange"
fruits.insert(1, "orange")
fruits
```

[9]:
```
['apple',
 'orange',
 'banana',
 'cherry',
 'grape',
 'pear',
 'watermelon',
 2,
 5,
 7,
 213,
 54,
 235,
 63,
 84]
```

[10]:
```
# del 语句 从列表删除指定位置的元素
del fruits[0]
fruits
```

[10]:
```
['orange',
 'banana',
 'cherry',
 'grape',
 'pear',
 'watermelon',
 2,
 5,
 7,
 213,
 54,
 235,
```

```
 63,
 84]
```

[11]:
```
# list.remove() 方法 移除列表中的指定元素
fruits.remove("orange")
fruits
```

[11]:
```
['banana',
 'cherry',
 'grape',
 'pear',
 'watermelon',
 2,
 5,
 7,
 213,
 54,
 235,
 63,
 84]
```

[12]:
```
# list.pop() 方法 移除列表中的指定位置的元素，并返回该元素
the_returned_value = fruits.pop(1)
print(fruits, "\n", the_returned_value)
```

```
['banana', 'grape', 'pear', 'watermelon', 2, 5, 7, 213, 54, 235, 63, 84]
 cherry
```

[13]:
```
# sort() 方法 对列表排序，默认为升序
num_list = [2, 5, 7, 213, 54, 235, 63, 84]
num_list.sort()
num_list
```

[13]:
```
[2, 5, 7, 54, 63, 84, 213, 235]
```

[14]:
```
# 传入 reverse 使列表改为降序
num_list.sort(reverse = True)
num_list
```

[14]:
```
[235, 213, 84, 63, 54, 7, 5, 2]
```

[15]:
```
# list.sort() 会永久性改变列表的元素排序
# 如果不想改变列表内元素本身的位置，又想以升降序的方式呈现，可以用 sorted 函数。
num_list = [2, 5, 7, 213, 54, 235, 63, 84]
print(sorted(num_list, reverse = True), "\n", num_list)
```

```
[235, 213, 84, 63, 54, 7, 5, 2]
 [2, 5, 7, 213, 54, 235, 63, 84]
```

[16]:
```
# list.reverse() 倒着输出列表内元素
num_list.reverse()
num_list
```

[16]:
```
[84, 63, 235, 54, 213, 7, 5, 2]
```

[17]:
```
# len() 函数获取列表长度，即元素的个数
len(num_list)
```

[17]:
```
8
```

2 List 进阶

[18]:
```
num_list.sort()
num_list
```

[18]:
```
[2, 5, 7, 54, 63, 84, 213, 235]
```

[19]:
```
# bisect 是 Python 标准库中的一个模块，它提供了对排序列表执行二分法搜索的函数
import bisect
# bisect.bisect 寻找并返回元素应当插入的位置
bisect.bisect(num_list, 10)
```

[19]:
```
3
```

[20]:
```
# bisect.insort() 将元素插入到目标列表中相应的位置
bisect.insort(num_list, 10)
num_list
```

[20]:
```
[2, 5, 7, 10, 54, 63, 84, 213, 235]
```

3 Data Structure 数据结构 - Tuple

Tuple 元组也是 python 中的一个有序数据类型。

Tuple 和 List 很相似，使用 ( ) 表示，和 list 最大的区别是 tuple 中元素不能修改。

[21]:
```
tuple1 = ("strawberry", "orange", "mango", "cherry")
```

[22]:
```
tuple1[0]
```

[22]:
```
'strawberry'
```

[23]:
```
# tuple 内元素不可修改
tuple1[0] = "kiwi"
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[23], line 2
      1 # tuple 内元素不可修改
----> 2 tuple1[0] = "kiwi"

TypeError: 'tuple' object does not support item assignment
```

Tuple 内部的元素不可变，但是 tuple 可以包含可变的数据结构，例如 list, dict.

[24]:
```
tuple2 = ("pineapple", ["lemon", "raspberry", "peach"])

# 通过修改 tuple 内的 list 列表，改变 tuple 内的数据。
tuple2[1].append("coconut")

tuple2
```

[24]:
```
('pineapple', ['lemon', 'raspberry', 'peach', 'coconut'])
```

那么，tuple 的不可变性质到底是什么意思呢？

从表面上看，tuple2 的内部数据发生了变化，但实际上 tuple 的不可变性质，指的是 tuple 内部的每个元素指向性不变，即 tuple 内存储的是元素的地址，这有点类似于 C++ 指针。

如果 tuple 内部存储的元素是不可变的，那么元素就不可变。

如果内部存储的元素是可变的，那么 tuple 不能改变的是指向这个元素的存储地址，而该可变元素的内部是可以发生变化的。

4 Summary

- list indexing 索引
- del list[x]
- list.append()
- list.insert()
- list.remove()
- list.pop()
- list.extend()
- list.sort()
- sorted(list)
- list.reverse()
- len(list)
- bisect.bisect()
- bisect.insort()
- tuple 的不可变性质