Table of contents

- 1. List Indexing and Slicing
- 2. List Operations
- 3. List Methods
- 4. Nested Lists
- 5. With For Loops
- 6. Listing Iterables
- 7.
- 8.
- 9.
- 10.
- 11.
- 12.
- 13.
- 14.
- 15.
- 16.
- 17.
- 18.
- 19.
- 20.
- 21.
- 22.
- 23.

---

1. List Indexing and Slicing

(go to top)

In [4]:
```
list1 = ['a', 'b', 'c', 'd', 'e']
```

In [5]:
```
list1[0]
```

Out[5]: 'a'

In [7]:
```
list1[-1]
```

Out[7]: 'e'

In [30]:
```
list1[4]
```

Out[30]: 'e'

In [14]:
```
list1[0] = 'z'
list1
```

Out[14]: ['z', 'b', 'c', 'd', 'e']

In [15]:
```
list1[0:3]
```

Out[15]: ['z', 'b', 'c']

In [16]:
```
list1[0:3] = 'ayz'
list1
```

Out[16]: ['a', 'y', 'z', 'd', 'e']

In [2]:
```
house = [['hallway', 11.25],['kitchen', 18.0], ['kiving room', 20.0],
```

In [4]:
```
house[0]
```

Out[4]: ['hallway', 11.25]

In [4]:
```
house[0][:]
```

Out[4]: ['hallway', 11.25]

In [3]:
```
house[0][0]
```

Out[3]: 'hallway'

In [5]:
```
house[0][1]
```

Out[5]: 11.25

Reverse Slicing

In [1]:
```
lst = ['1', '2', '3', '4', '5']
```

In [57]:
```
lst[5::-1]
```

Out[57]:
```
['5', '4', '3', '2', '1']
```

In [56]:
```
lst[::-1]
```

Out[56]:
```
['5', '4', '3', '2', '1']
```

In [ ]:
```

```

2. List Operations

(go to top)

In [18]:
```
list1 = ['a', 'bbbbb', 'ca', 'deb', 'e']
list2 = ['f', 'g', 'g', 'i', 'j']
list5 = [5,6,9,8,2,1,6,3,4,0]
```

In [19]:
```
list1 + list2
```

Out[19]:
```
['a', 'bbbbb', 'ca', 'deb', 'e', 'f', 'g', 'g', 'i', 'j']
```

In [20]:
```
list1 * 2
```

Out[20]:
```
['a', 'bbbbb', 'ca', 'deb', 'e', 'a', 'bbbbb', 'ca', 'deb', 'e']
```

3. List Methods

(go to top)

In [46]:
```
list1 = ['a', 'ca', 'deb', 'e', 'bbbbb',]
list2 = ['f', 'g', 'g', 'i', 'j']

list3 = ['a', 'b', 2, 'd', 'e']
list4 = ['f', 'g', 45, 'i', 'j']
list5 = [5,6,9,8,2,1,6,3,4,0]
```

len()

In [47]:
```
len(list4)
```

Out[47]: 5

sorted()

In [48]:
```
sorted(list5)
```

Out[48]: [0, 1, 2, 3, 4, 5, 6, 6, 8, 9]

In [49]:
```
sorted(list1)
```

Out[49]: ['a', 'bbbbb', 'ca', 'deb', 'e']

sort()

- list.sort(reverse=True|False, key=myFunc)
- #list5.sort() #alphabetical order if strings
- #list5.sort(key=int)

In [63]:
```
list5.sort()
```

In [64]:
```
list5
```

Out[64]: [0, 1, 2, 3, 4, 5, 6, 6, 8, 9]

In [65]:
```
list1.sort()
```

In [66]:
```
list1
```

Out[66]:
```
['a', 'bbbbb', 'ca', 'deb', 'e']
```

In [67]:
```
list5.sort(reverse=True)
```

In [68]:
```
list1.sort(reverse=True)
```

In [69]:
```
list5
```

Out[69]:
```
[9, 8, 6, 6, 5, 4, 3, 2, 1, 0]
```

In [70]:
```
list1
```

Out[70]:
```
['e', 'deb', 'ca', 'bbbbb', 'a']
```

In [71]:
```
list5.sort(reverse=True, key=int)
```

In [72]:
```
list1.sort(key=len)
```

In [73]:
```
list5
```

Out[73]:
```
[9, 8, 6, 6, 5, 4, 3, 2, 1, 0]
```

In [74]:
```
list1
```

Out[74]:
```
['e', 'a', 'ca', 'deb', 'bbbbb']
```

In [75]:
```
list1.sort(reverse=True, key=len)
```

In [76]:
```
list1
```

Out[76]:
```
['bbbbb', 'deb', 'ca', 'e', 'a']
```

.extend()

In [20]:
```
list3.extend('xyz')
print(list3)

list4.extend(list1)
print(list4)
```

```
['a', 'b', 2, 'd', 'e', 'x', 'y', 'z']
['f', 'g', 45, 'i', 'j', 'a', 'b', 'c', 'd', 'e']
```

.append()

In [21]:
```
list1.append('abc')
print(list1)
list2.append(list1)
print(list2)
```

```
['a', 'b', 'c', 'd', 'e', 'abc']
['f', 'g', 'g', 'i', 'j', ['a', 'b', 'c', 'd', 'e', 'abc']]
```

In

In [22]:
```
'a' in list1
```

Out[22]:
```
True
```

list( )

In [23]:
```
list({'Test', 'Math', 1, 3, 'Five'})
```

Out[23]:
```
['Test', 1, 3, 'Math', 'Five']
```

In [24]:
```
new_list = list('abcde')
new_list
```

Out[24]:
```
['a', 'b', 'c', 'd', 'e']
```

sum( )

In [26]:
```
sum(list5)
```

Out[26]: 44

index( )

- The index() method returns an integer that represents the index of first match of specified element in the List.
- `list_name.index(element, start, end)`
  - element - The element whose lowest index will be returned.
  - start (Optional) - The position from where the search begins.
  - end (Optional) - The position from where the search ends.

In [94]:
```
list5
```

Out[94]: [9, 8, 6, 6, 5, 4, 3, 2, 1, 0]

In [102]:
```
list5.index(6)
```

Out[102]: 2

In [101]:
```
list5.index(6,3, -1)
```

Out[101]: 3

set( )

- will remove duplicates and create dictinary in random order

In [4]:
```
lst = ['apple', 'banana', 'apple', 'orange']
lst_2 = set(lst)
```

In [57]:
```
lst_2
```

Out[57]: {'apple', 'banana', 'orange'}

reverse()

In [5]:
```
lst = ['apple', 'banana', 'apple', 'orange']
```

In [78]:
```
lst
```

Out[78]:
```
['apple', 'banana', 'apple', 'orange']
```

In [83]:
```
lst.reverse()
```

In [84]:
```
lst
```

Out[84]:
```
['orange', 'apple', 'banana', 'apple']
```

insert(i,x)

- Inserts x into list at index i.

In [5]:
```
lst = ['apple', 'banana', 'apple', 'orange']
```

In [103]:
```
lst
```

Out[103]:
```
['orange', 'apple', 'banana', 'apple']
```

In [106]:
```
lst.insert(0, 'guava')
```

In [107]:
```
lst
```

Out[107]:
```
['guava', 'orange', 'apple', 'banana', 'apple']
```

count(x)

- Returns the number of occurrences of x in list.

In [5]:
```
lst = ['apple', 'banana', 'apple', 'orange']
```

In [108]:
```
lst
```

Out[108]:
```
['guava', 'orange', 'apple', 'banana', 'apple']
```

In [109]:
```
lst.count('apple')
```

Out[109]:
```
2
```

remove(x)

- Deletes the first occurrence of x in list.

```
In [5]: lst = ['apple', 'banana', 'apple', 'orange']
```

```
In [110]: lst
Out[110]: ['guava', 'orange', 'apple', 'banana', 'apple']
```

```
In [111]: lst.remove('apple')
```

```
In [112]: lst
Out[112]: ['guava', 'orange', 'banana', 'apple']
```

pop(i)

- Deletes the ith element of the list and returns its value.

```
In [6]: lst = ['apple', 'banana', 'apple', 'orange']
```

```
In [7]: lst
Out[7]: ['apple', 'banana', 'apple', 'orange']
```

```
In [8]: lst.pop(2)
Out[8]: 'apple'
```

```
In [12]: lst
Out[12]: ['apple', 'banana', 'orange']
```

```
In [15]: lst.pop(-1)
Out[15]: 'orange'
```

```
In [16]: lst
Out[16]: ['apple', 'banana']
```

4. Nested Lists

(go to top)

In [1]:
```
house = [['hallway', 11.25],['kitchen', 18.0], ['kiving room', 20.0],
```

5. With For Loops

(go to top)

as a sequence / range

(go to top)

In [2]:
```
list1 = ['a', 'b', 'c', 'd', 'e']
```

In [3]:
```
for entry in list1:
    print(entry)
```

```
a
b
c
d
e
```

.append()

(go to top)

In [10]:
```
squares = []
numbers = []
```

In [11]:
```
for i in range(1,10):
    squares.append(i**2)
    numbers.append(i)
```

In [12]:
```
print(numbers, '\n', squares)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
 [1, 4, 9, 16, 25, 36, 49, 64, 81]
```

Nested Lists

(go to top)

In [6]:
```
house = [['hallway', 11.25],['kitchen', 18.0], ['kiving room', 20.0],
```

In [7]:
```
for i in house:
    print(i)
```

```
['hallway', 11.25]
['kitchen', 18.0]
['kiving room', 20.0]
['bedroom', 10.75]
['bathroom', 9.5]
```

- for the first iteration i is ['hallway', 11.25]. therefor i[0] is "hallway"

In [8]:
```
for i in house:
    print('the ' + i[0] + ' is ' + str(i[1]) + ' sqm')
```

```
the hallway is 11.25 sqm
the kitchen is 18.0 sqm
the kiving room is 20.0 sqm
the bedroom is 10.75 sqm
the bathroom is 9.5 sqm
```

6. Listing iterables

(go to top)

`list(range())`

(go to top)

In [1]:
```
list(range(10))
```

Out[1]:
```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

7. Title

(go to top)

8. Title

(go to top)

9. Title

(go to top)

10. Title

(go to top)

11. Title

(go to top)

12. Title

(go to top)

13. Title

(go to top)

14. Title

(go to top)

15. Title

(go to top)

16. Title

(go to top)

17. Title

(go to top)

18. Title

(go to top)

19. Title

(go to top)

20. Title

(go to top)

In [ ]:

In [ ]: