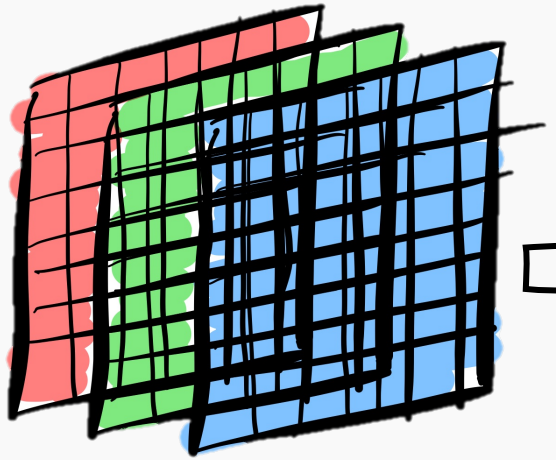
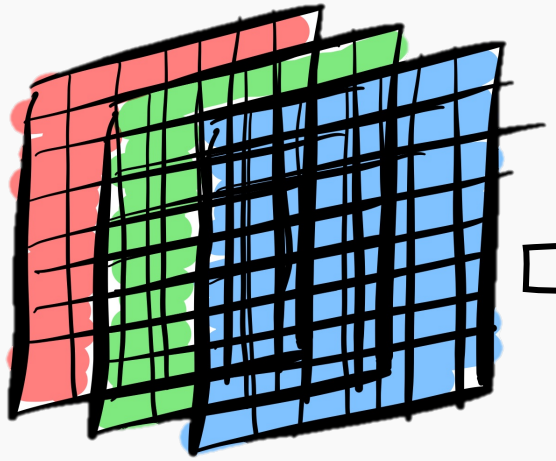
Lists

Image pixels



```
array([[[ 63,  95, 208, 231, 190, 186, 196, 208,  91, 110,  31,  66,
          163,  75,  96,  84,  86, 106,   0,  57,  90, 147,  31, 148,
           63,  10,  61, 231,  80, 127, 174, 240,  45,  56, 153, 193,
          238,  85,  38, 118,  94, 145, 144, 192,  82,  48, 145,  17,
          176, 150,  43, 111,  55, 104, 249,  42,  25, 145,  88, 102,
          115,  12,  91, 238, 235, 198, 141,  96, 118, 167, 222, 123,
           43, 235,  10, 205,  68, 249, 208,  71, 112,  94, 146,   1,
           16, 134,   7,  94,  28, 148,  77,  58, 122, 232,  59, 194,
           16,   7, 204, 120, 174, 192,  72, 197,  95,  18, 188, 243,
           83, 201, 137, 250, 237, 149, 222, 150,  96,   2, 191,  15,
          164, 167, 147,  70, 193,  30,  33, 108,  65,  77, 192, 237,
          189,  47, 166, 116, 141, 107,  39, 158, 241,  33, 170,  37,
          102, 173,  82, 181, 229, 189, 161,  18, 251, 187,  95,  28,
           45, 132,  66, 201, 166, 243,  18, 135,   9, 171, 110,  74,
           99, 207, 185, 228, 243,  17, 236,  60,  61, 142,  12,  54,
          203,  33,  30,  97, 202, 231,  34, 137, 162, 172, 100, 233,
          116,  58, 172,  35,  91, 100,   2,   6, 221,  86, 176,  15,
           15, 117,  57,  54, 108, 158, 145, 155,  89, 116,  73,  28,
          169,  33, 175, 120, 129, 187,  18,  82],
         [ 36,  94,  73, 107, 127, 205, 232, 139, 189, 180,  94, 192,
           64, 218, 138,  38, 249, 243,  36,  31,  41, 116,  81, 208,
          106, 176, 108,   2,  38,  86, 152, 185,  43, 145, 217, 109,
           93, 144, 250,  56,  83,  52, 248, 115, 129, 236, 102, 117,
          211, 200,   9,  11,  49, 120, 251, 101, 177,  39,  43,  25,
          217, 108, 121, 217, 225, 201, 124,  10,  80, 102, 161, 100,
```



```
array([[[154, 187,  54,  33,  40,  57,  67, 196,  48,  77, 223,  36,
           70,  36, 173, 229, 131,  53,  15,  68,  39, 241, 210,   3,
            7, 135,  11,  35,  63,  19, 183,  65, 214, 194, 156, 177,
            5, 238, 241, 112, 183,  30, 234,  50, 201,  67,   1, 174,
            2, 190, 161, 210,  90,  44,  66,   1, 145, 176,  22, 232,
           57, 133, 234,  52, 195, 104,  18, 180,  65,  67, 244,  72,
          188, 104, 206, 186, 224,  95, 247, 145, 212, 126,  40,  54,
          170, 251, 131, 157,   9, 132, 221, 194, 205,  52,  97, 106,
          145,  64,  12, 244,  57, 110,   5, 126, 106, 121, 145,  31,
          185,  10,   5,  25,   8,  99,  17,  24, 228, 169, 112, 154,
           50, 207, 215, 211,  92,  25, 230,  40,  70, 213, 226, 218,
           72, 161,  32,  90,   7, 250,  79, 221, 127, 187, 175, 228,
            4,  56, 122, 206, 130, 104, 231, 155,  60, 144, 106, 242,
           21, 237, 217, 120, 137, 191,  87, 218, 207,  22,  11,   0,
           38,  28,  19,  21, 253, 205,  28, 181,  40, 236, 204, 233,
          244, 167,  10, 191,  28,   3,  53, 102, 157, 148, 143, 105,
          210, 195,  83, 101, 235, 204, 166,  58, 241, 170,  26, 162,
          110, 162,  43, 200,   5, 213,  80, 130, 237, 193, 241, 103,
           68,  17, 115,  15,  81, 171,  34,  33],
         [172, 181,  55, 176,  99, 100,   8, 105, 252,  17, 212, 233,
          214, 174,  91, 212,   5,  13,  18,  21, 183,  99,  78,  44,
          100, 175, 241, 135,  37, 135, 191, 215, 129, 138, 221,  25,
           90, 184,  69,  95,  58, 133, 217, 188,  65, 160, 180, 198,
           70,   1, 104, 136,  39, 197,  68, 220, 239,  26, 104,  96,
           79,  44,  97, 131, 214,  19, 240, 247,  36,  23, 205,  85,
```


MATRIX


LISTS

Lists [ ]

What is a python list?

1. An ordered collection
2. That is resizable
3. And contains elements of different types

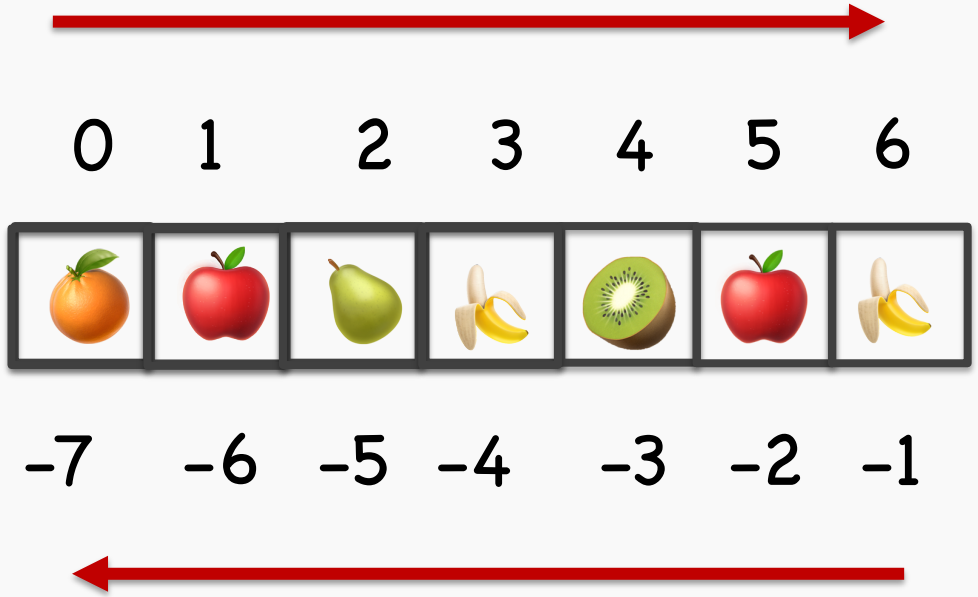
```
#Create a list
num_list = [1,2,3]

#A list can have a mixture of all types
mix_list = ['hello’, 1, True]
```

How to access elements of a list?

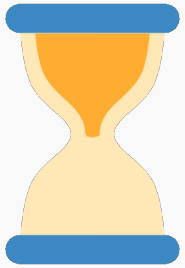
List Indexing

How do we access elements of a list ?

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple',
'pear', 'banana', 'kiwi', 'apple',
'banana']
```



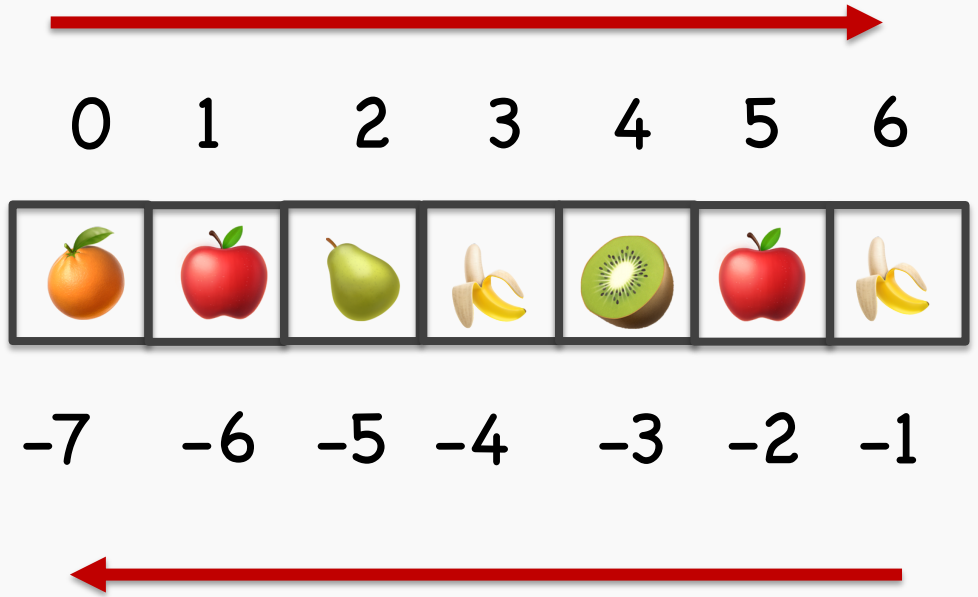

```
List[start:stop:(Optional)step]
```



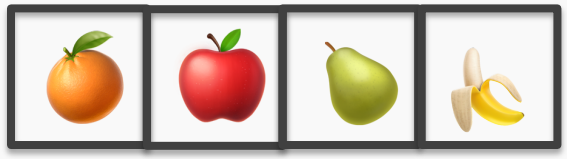
Digestion Time

List Indexing

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple',
'pear', 'banana', 'kiwi', 'apple',
'banana']
```



fruits[0:4] =

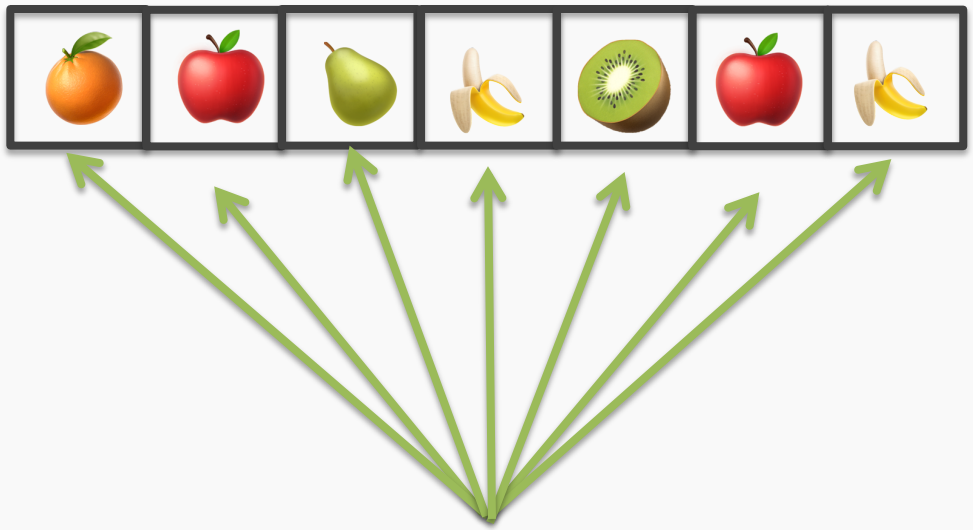


List methods

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']
```

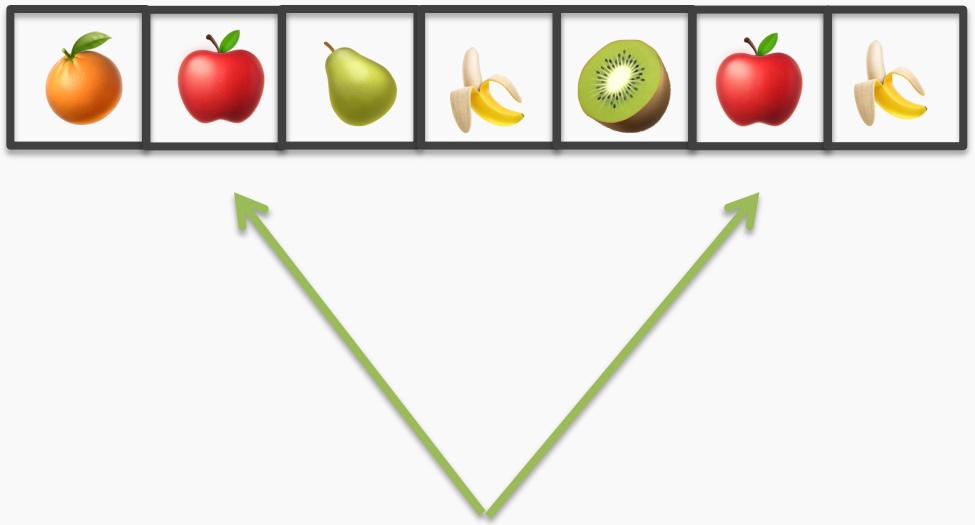
List methods

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']

>>> len(fruits)
7
```



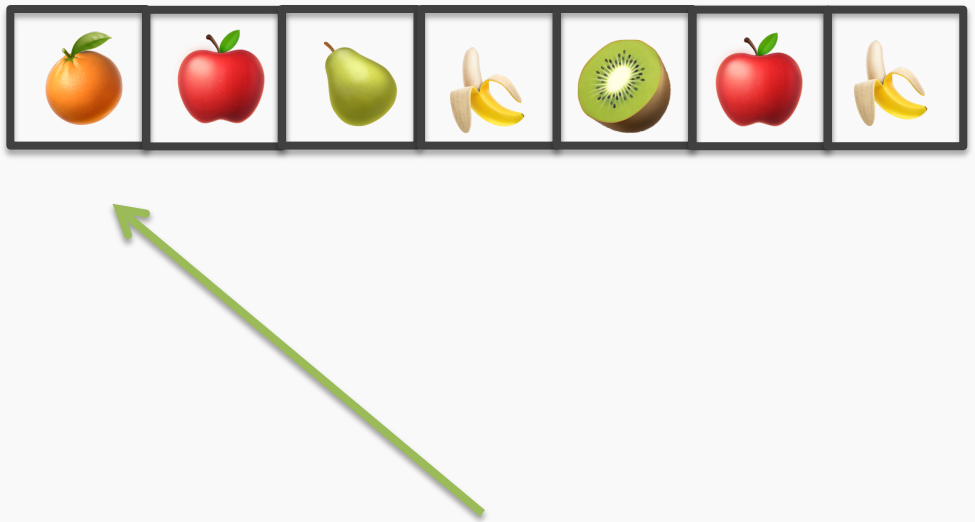
Count()

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']

>>> fruits.count('apple')
2
```



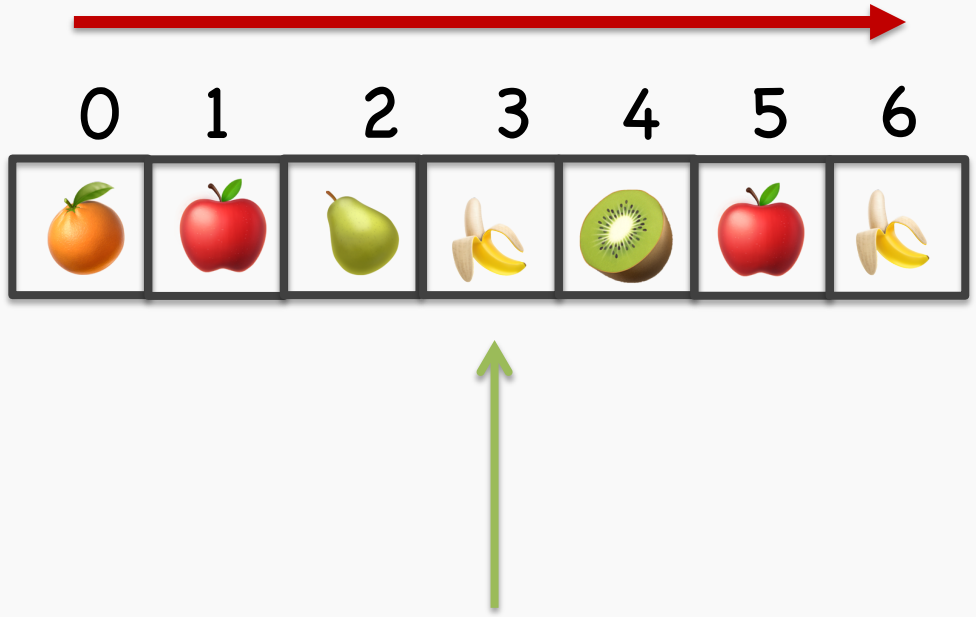
Count()

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']

>>> fruits.count('tangerine')
1
```



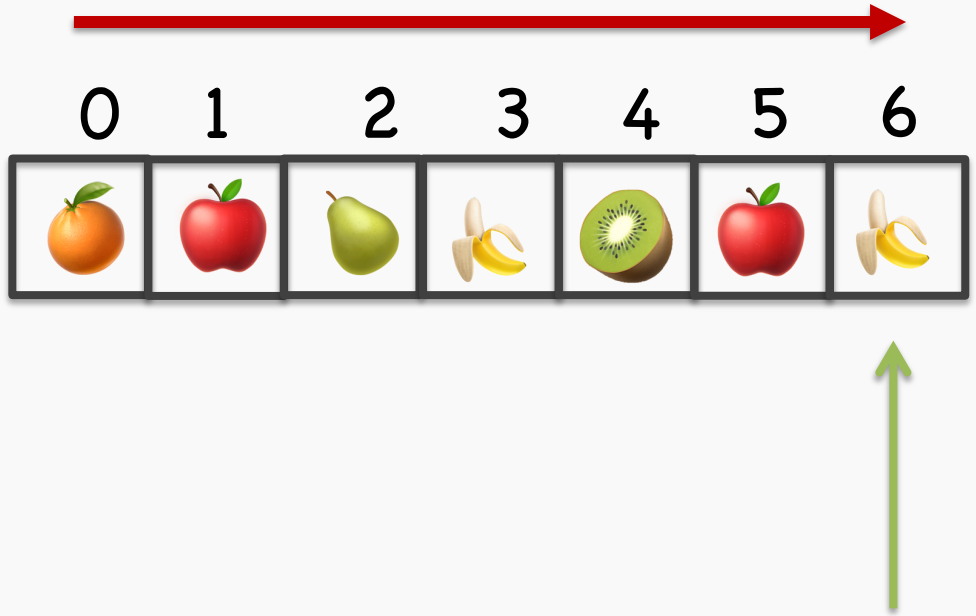
Index()

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']


>>> fruits.index('banana')
3
```



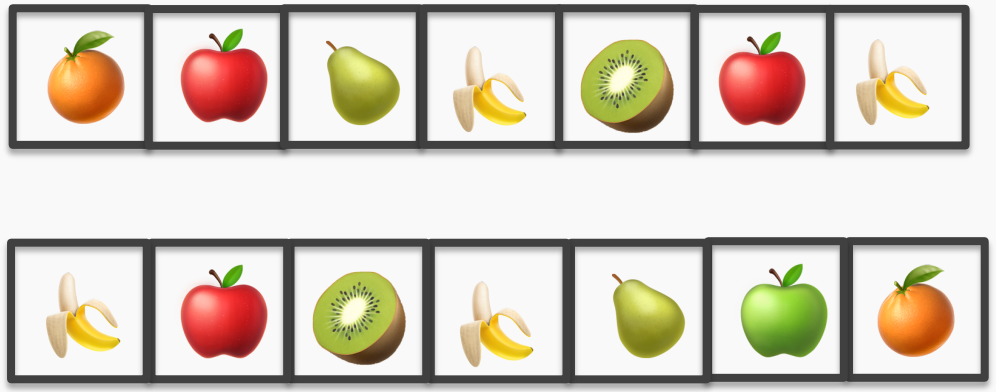
Index()

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']

>>> fruits.index('banana', 4) # Find next
banana starting a position 4
6
```



Reverse()

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']
>>> fruits.reverse()
>>> fruits
['banana', 'apple', 'kiwi', 'banana', 'pear', 'apple', 'tangerine']
```



List methods – sort()

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']
```

List methods – sort()

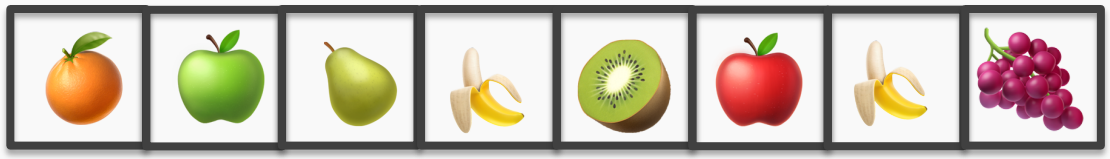
```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']

>>> fruits.sort()
>>> fruits
['apple', 'apple', 'banana', 'banana', 'kiwi', 'pear', 'tangerine']
```

List methods

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']
```

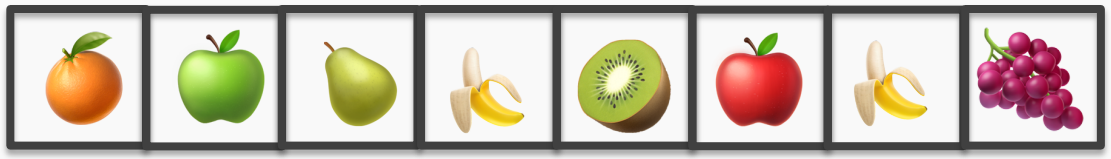
List methods – append()

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']

>>> fruits.append('grape')
>>> fruits
['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana', 'grape']
```



List methods – pop()

```
>>> fruits.pop('grape')
```



List methods – pop()

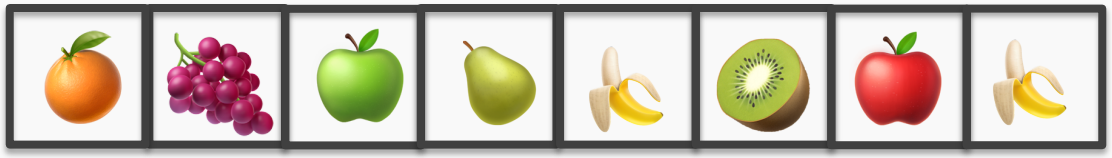
```
>>> fruits.pop('grape')
>>> fruits
['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']
```

List methods

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']
```

List methods – insert()

```
# Make a list of fruits
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']

>>> fruits.insert(1,'grape')
>>> fruits
['tangerine', 'grape', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']
```



List methods – remove()

```
>>> fruits.remove('grape')
>>> fruits
['tangerine','apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']
```

List methods

```
>>> fruits = ['tangerine', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']
>>> fruits.count('apple')
2
>>> fruits.count('tangerine')
1
>>> fruits.index('banana')
3
>>> fruits.index('banana', 4) # Find next banana starting a position 4
6
>>> fruits.reverse()
>>> fruits
['banana', 'apple', 'kiwi', 'banana', 'pear', 'apple', 'tangerine']
>>> fruits.append('grape')
>>> fruits
['banana', 'apple', 'kiwi', 'banana', 'pear', 'apple', 'tangerine', 'grape']
>>> fruits.sort()
>>> fruits
['apple', 'apple', 'banana', 'banana', 'grape', 'kiwi', 'pear', 'tangerine']
>>> fruits.pop()
'tangerine'
```

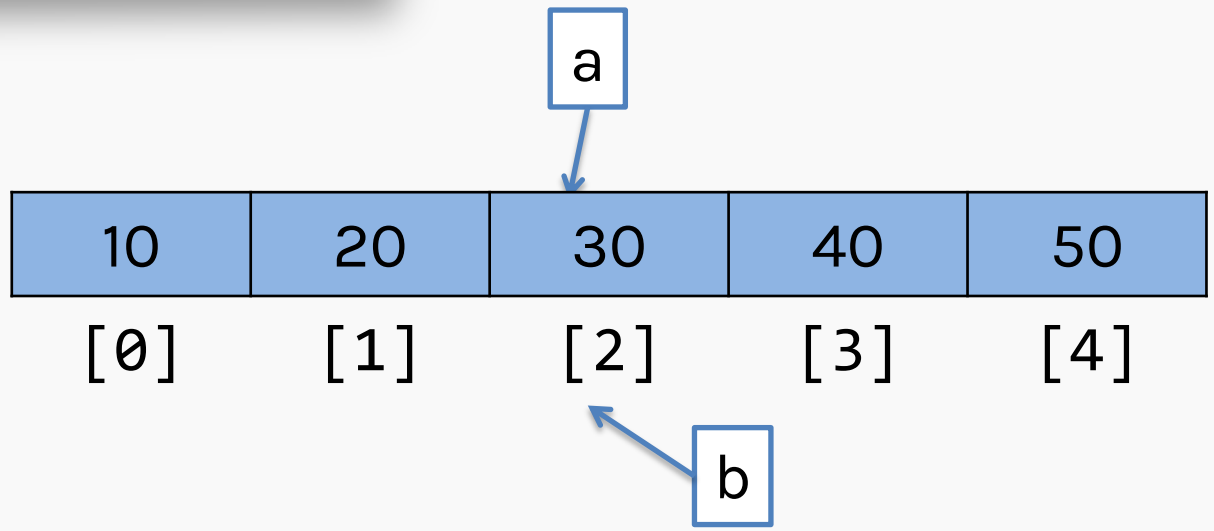
What does 'Mutable' mean ?

List Aliasing

Aliasing means giving another name to the existing object. It doesn't n
a

```
b = a
```

Modification in *a* will affect *b* and vice versa.

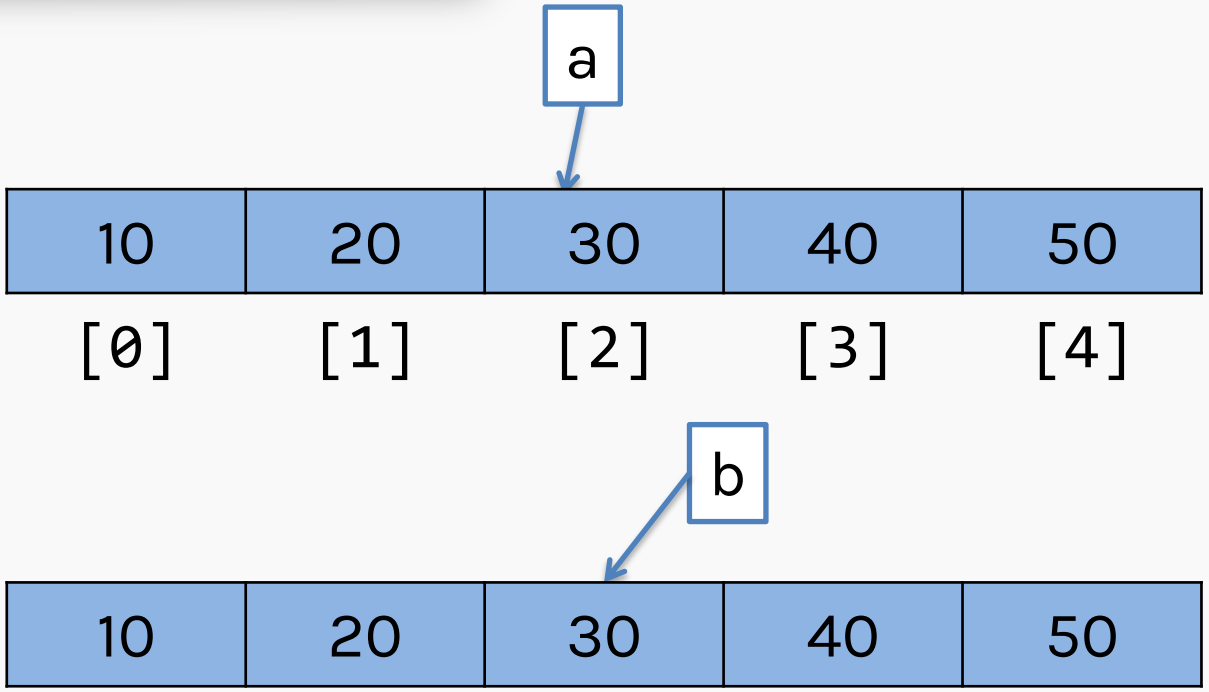


Copy a list

Whe[illegible]f all the elements is stored in a[illegible]pendent.

Modification in *a* will **not** affect *b* and vice versa.

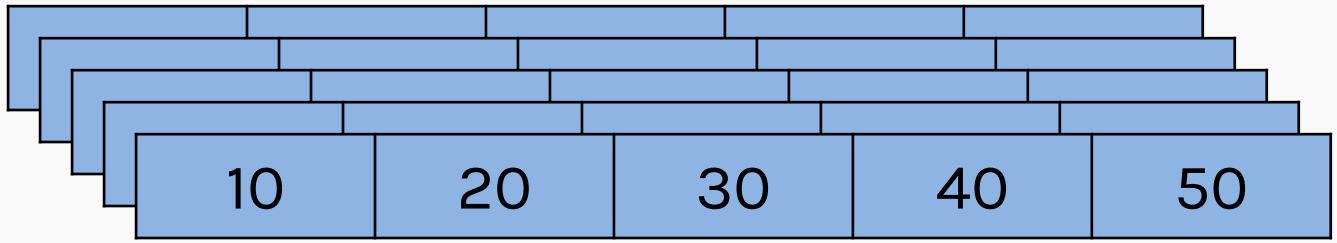
```
a = [10, 20, 30, 40, 50]
b = a.copy()
```



Nested Lists

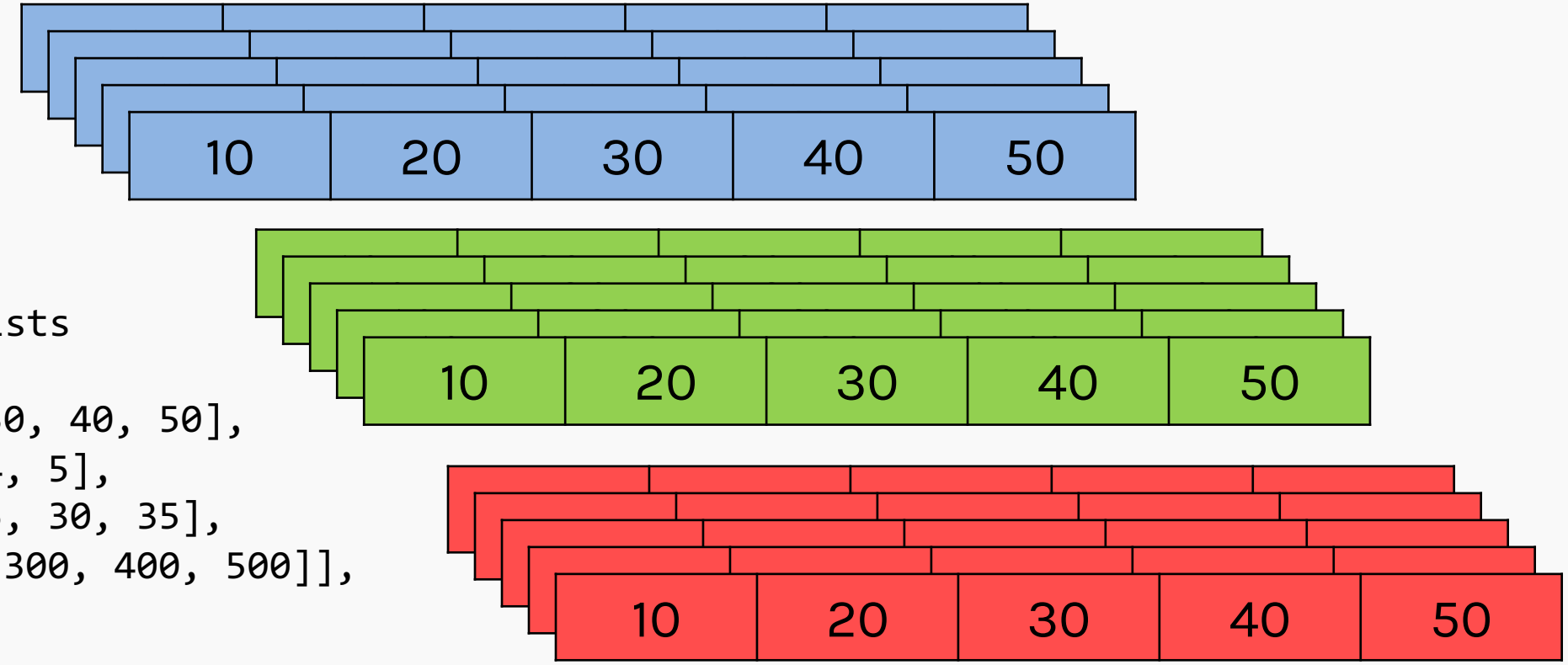
Use cases

```
# Nest multiple lists
nested_a = [
           [10, 20, 30, 40, 50],
           [1, 2, 3, 4, 5],
           [15, 20, 25, 30, 35],
           [100, 200, 300, 400, 500]
           ]
```



Use cases

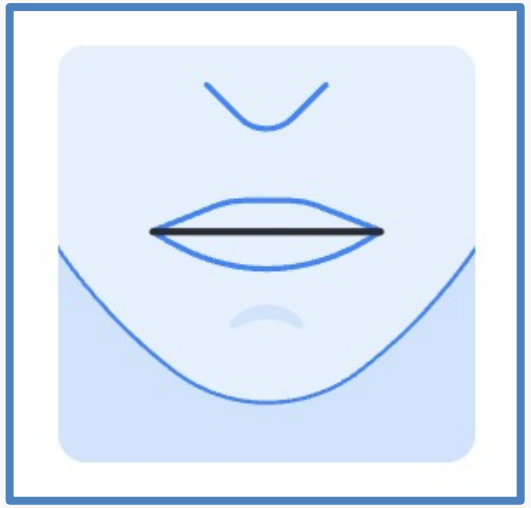
```
# Nest 3-D multiple lists
nested_a = [
           [[10, 20, 30, 40, 50],
           [1, 2, 3, 4, 5],
           [15, 20, 25, 30, 35],
           [100, 200, 300, 400, 500]],
           ...
           ...]
```



Iteration, Iterator, Iterable

❌

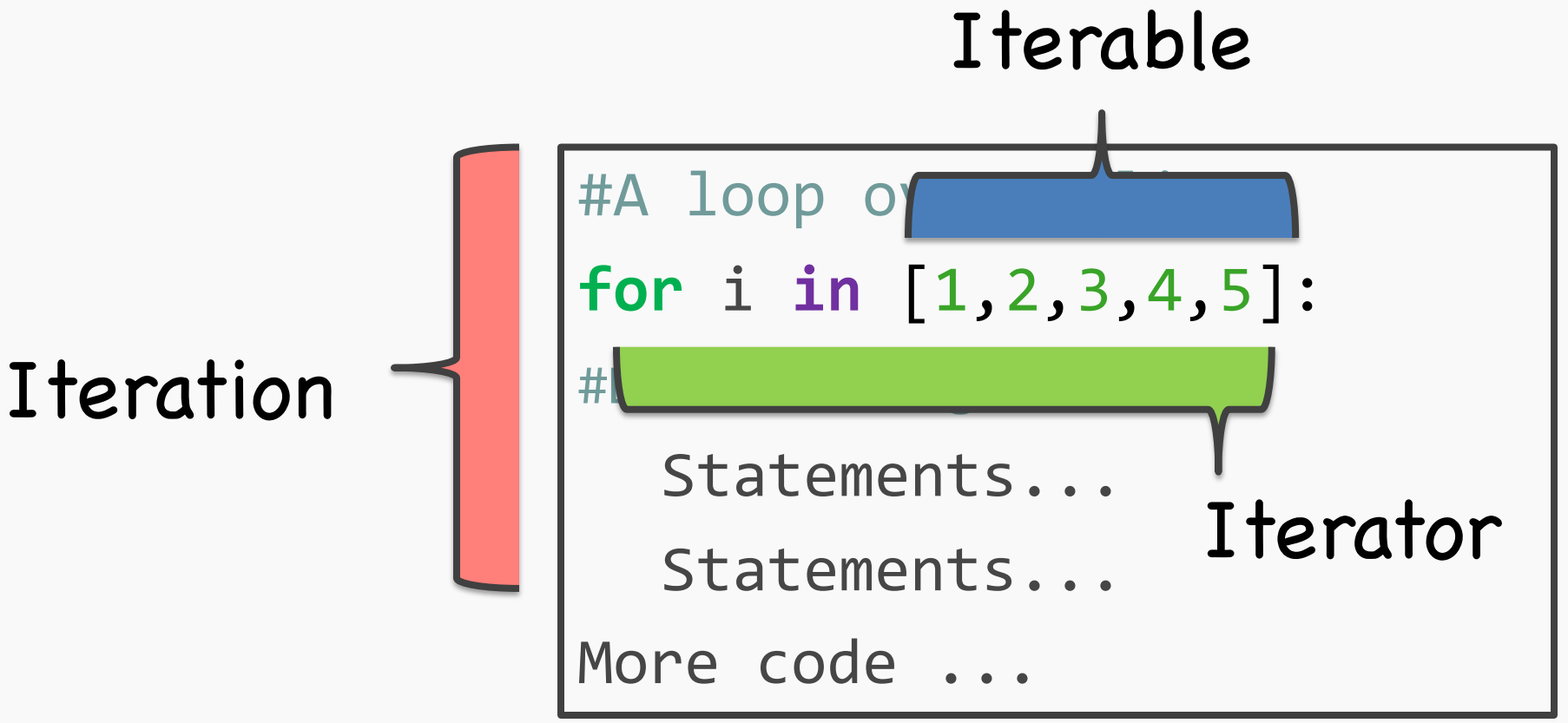
"I" terable



✅

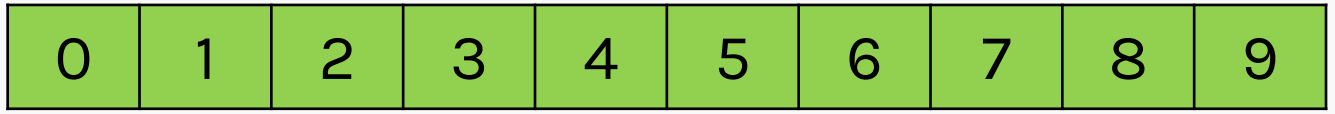
"It" erable

Iteration, Iterators Iterable



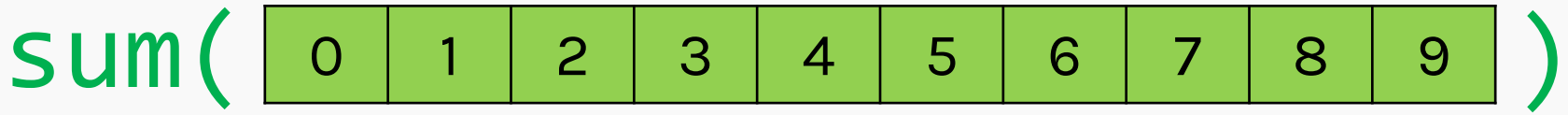
Range function

```
# Create a list of integers
new_list = list(range(10))
print(new_list)
>>> [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```




```
range[start:stop:(Optional)step]
```

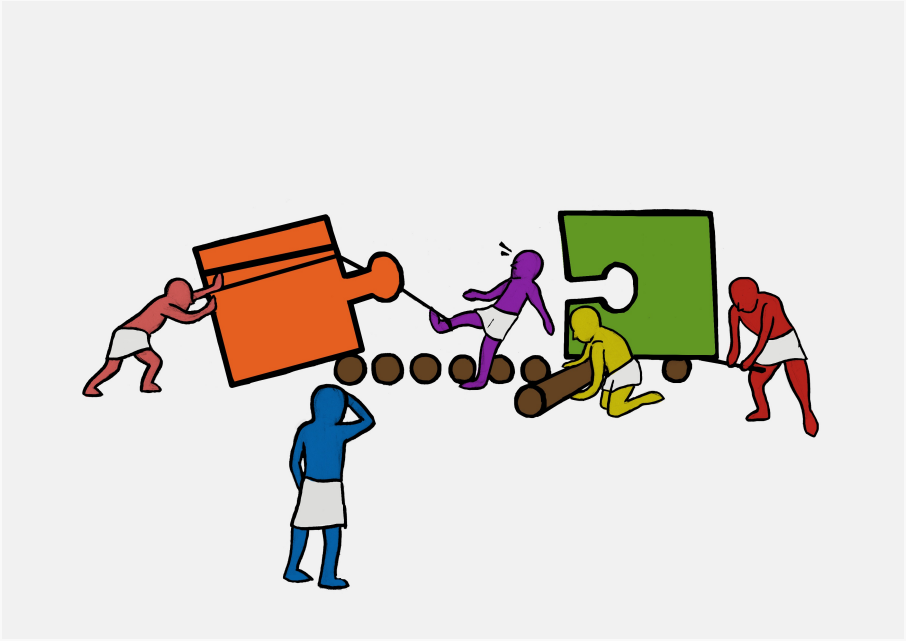
Built-in method – **Sum()**



Optional value to be added to the sum of the sequence; zero if not specified

```
sum(listlike,[,start])
```

Exercise #2



Description

Exercise: Lists

The goal of this exercise is to manipulate lists.

Forward direction indexing

	0	1	2	3	4
List	MON	TUE	WED	THU	FRI
	-5	-4	-3	-2	-1

Backward direction indexing