In [ ]:

```
Python Tuples
```

In [ ]:

```
Tuple

Tuples are used to store multiple items in a single variable.

Tuple is one of 4 built-in data types in Python used to store collections of data, the othe
3 are List, Set, and Dictionary, all with different qualities and usage.

A tuple is a collection which is ordered and unchangeable.

Tuples are written with round brackets.
```

In [ ]:

```
Tuple Items

Tuple items are ordered, unchangeable, and allow duplicate values.

Tuple items are indexed, the first item has index [0], the second item has index [1] etc.

Ordered

When we say that tuples are ordered, it means that the items have a defined order, and that
order will not change.

Unchangeable

Tuples are unchangeable, meaning that we cannot change, add or remove items after the tuple
been created.

Allow Duplicates

Since tuples are indexed, they can have items with the same value:
```

In [ ]:

```
Tuple Methods

Python has two built-in methods that you can use on tuples.

Method  Description
count() Returns the number of times a specified value occurs in a tuple
index() Searches the tuple for a specified value and returns the position of where it was f
```

In [2]:

```
thistuple = ("apple", "banana", "cherry")
print(thistuple)
```

```
('apple', 'banana', 'cherry')
```

In [2]:

```
thistuple = tuple(("apple", "banana", "cherry"))
print(thistuple)
```

```
('apple', 'banana', 'cherry')
```

In [3]:

```
thistuple = tuple(("apple", "banana", "cherry"))
print(thistuple)
```

```
('apple', 'banana', 'cherry')
```

In [1]:

```
mytuple = ("apple", "banana", "cherry")

print(type(mytuple))
```

```
<class 'tuple'>
```

In [5]:

```
tuple1 = ("abc", 34, True, 40, "male")

print(tuple1)
```

```
('abc', 34, True, 40, 'male')
```

In [6]:

```
tuple1 = ("apple", "banana", "cherry")
tuple2 = (1, 5, 7, 9, 3)
tuple3 = (True, False, False)

print(tuple1)
print(tuple2)
print(tuple3)
```

```
('apple', 'banana', 'cherry')
(1, 5, 7, 9, 3)
(True, False, False)
```

In [6]:

```
thistuple = ("apple",)
print(type(thistuple))

#NOT a tuple
thistuple = ("apple")
print(type(thistuple))
```

```
<class 'tuple'>
<class 'str'>
```

In [8]:

```
thistuple = tuple(("apple", "banana", "cherry"))
print(len(thistuple))
```

```
3
```

In [9]:

```
thistuple = ("apple", "banana", "cherry", "apple", "cherry")
print(thistuple)
```

```
('apple', 'banana', 'cherry', 'apple', 'cherry')
```

In [10]:

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

```
banana
```

In [11]:

```
thistuple = ("apple", "banana", "cherry")
print(thistuple[-1])
```

```
cherry
```

In [9]:

```
thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")
print(thistuple[2:5])

#This will return the items from position 2 to 5.

#Remember that the first item is position 0,
#and note that the item in position 5 is NOT included
```

```
('cherry', 'orange', 'kiwi')
```

In [13]:

```
thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")

print(thistuple[:4])
```

```
('apple', 'banana', 'cherry', 'orange')
```

In [14]:

```
thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")

print(thistuple[2:])
```

```
('cherry', 'orange', 'kiwi', 'melon', 'mango')
```

In [15]:

```
thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")
print(thistuple[-4:-1])

#Negative indexing means starting from the end of the tuple.

#This example returns the items from index -4 (included) to index -1 (excluded)

#Remember that the last item has the index -1,
```

```
('orange', 'kiwi', 'melon')
```

In [16]:

```
thistuple = ("apple", "banana", "cherry")
if "apple" in thistuple:
  print("Yes, 'apple' is in the fruits tuple")
```

```
Yes, 'apple' is in the fruits tuple
```

In [13]:

```
x = ("apple", "banana", "cherry")
y = list(x)
print(y,type(y))
print(type(x))
y[1] = "kiwi"
x = tuple(y)

print(x)
```

```
['apple', 'banana', 'cherry'] <class 'list'>
<class 'tuple'>
('apple', 'kiwi', 'cherry')
```

In [18]:

```
thistuple = ("apple", "banana", "cherry")
y = list(thistuple)
y.append("orange")
thistuple = tuple(y)

print(thistuple)
```

```
('apple', 'banana', 'cherry', 'orange')
```

In [17]:

```
thistuple = ("apple", "banana", "cherry")
y = ("orange",)
thistuple += y

print(thistuple)
```

```
('apple', 'banana', 'cherry', 'orange')
```

In [5]:

```
thistuple = ("apple", "banana", "cherry")
y = list(thistuple)
print(y,type(y))
y.remove("apple")
thistuple = tuple(y)

print(thistuple)
print(type(y))
```

```
['apple', 'banana', 'cherry'] <class 'list'>
('banana', 'cherry')
<class 'list'>
```

In [21]:

```
thistuple = ("apple", "banana", "cherry")
del thistuple
print(thistuple) #this will raise an error because the tuple no longer exists
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-21-8a57ed92fc5d> in <module>
      1 thistuple = ("apple", "banana", "cherry")
      2 del thistuple
----> 3 print(thistuple) #this will raise an error because the tuple no long
er exists

NameError: name 'thistuple' is not defined
```

In [22]:

```
fruits = ("apple", "banana", "cherry")

print(fruits)
```

```
('apple', 'banana', 'cherry')
```

In [23]:

```
fruits = ("apple", "banana", "cherry")

(green, yellow, red) = fruits

print(green)
print(yellow)
print(red)
```

```
apple
banana
cherry
```

In [23]:

```
fruits = ("apple", "banana", "cherry", "strawberry", "raspberry")

(green, yellow,red,*orange) = fruits

print(green)
print(yellow)
print(red)
print(orange)
```

```
apple
banana
cherry
['strawberry', 'raspberry']
```

In [25]:

```
fruits = ("apple", "mango", "papaya", "pineapple", "cherry")

(green, *tropic, red) = fruits

print(green)
print(tropic)
print(red)
```

```
apple
['mango', 'papaya', 'pineapple']
cherry
```

In [6]:

```
thistuple = ("apple", "banana", "cherry")
for x in thistuple:
  print(x,end=" ")
```

```
apple banana cherry
```

In [29]:

```
thistuple = ("apple", "banana", "cherry")
for i in range(len(thistuple)):
  print(thistuple[i])
```

```
apple
banana
cherry
```

In [28]:

```
thistuple = ("apple", "banana", "cherry")
i = 0
while i < len(thistuple):
  print(thistuple[i])
  i = i + 1
```

```
apple
banana
cherry
```

In [30]:

```
tuple1 = ("a", "b" , "c")
tuple2 = (1, 2, 3)

tuple3 = tuple1 + tuple2
print(tuple3)
```

```
('a', 'b', 'c', 1, 2, 3)
```

In [31]:

```
fruits = ("apple", "banana", "cherry")
mytuple = fruits * 2

print(mytuple)
```

```
('apple', 'banana', 'cherry', 'apple', 'banana', 'cherry')
```