C: Tuples in more detail

1. Can **pull out components** with a **tuple on left** of an assignment statement:

```
names = "Potter,Harry".split(",")
(last, first) = names
```

Creates a list

Extracts components to individual variable

names ← ["Potter", "Harry"]
last ← "Potter"
first ← "Harry"

Ok to omit the parentheses:

```
last, first = names
```

2. Many functions return **tuples** in order to pass back **multiple values**:

```
fig1, ax1 = plt.subplots()
```

subplots() returns two values:
a Figure object and an Axes object.

3. Can use to **retrieve keys and values** from dictionaries using **.items()**:

Three ways to extract or loop through parts of dictionaries:

List of keys:	dname.keys()
List of values:	dname.values()
List of (key,value) tuples:	dname.items()

```
names = { "36067":"Onondaga County", … }
```

```
for key,value in names.items():
```

key ⬅ "36067"
value ⬅ "Onondaga County"

4. Can combine **several lists** into **single list of tuples** using **zip()**:

```
list1 = ["a", "b", "c"]
list2 = [1, 2, 3]

tlist = zip(list1, list2)
```

tlist 👉 [("a",1),
("b",2),
("c",3)]

5. Can **build dictionaries** from lists of tuples containing key, value pairs:

```
d1 = dict( tlist )
```

d1 👉 { "a":1,
"b":2,
"c":3 }

6. Can **number lists** via **enumerate()**:

```
list3 = ["a", "b", "c"]

tuples = enumerate(list3)

for n,v in tuples:
   print(f"item {n} is {v}")
```

tuples 👉 [(0,"a"),
(1,"b"),
(2,"c")]