Tuples and dictionaries

Ben Bolker

19:07 05 February 2017

Reference; reference

Tuples

- simple; **non-mutable** version of lists
- faster, safer
- can be expressed as `x, y, z` (or `(x,y,z)`, probably clearer)
- empty tuple: `()`
- tuple with one element: `(x,)`
- can do many of the same things as with lists

```
x = (1,4,"a",3)
print(x[1])   ## indexing
print(x[2:])  ## slicing
print(x+(3,)) ## appending
print(x[:2] + (3,) + x[2:]) ## insertion in the middle
x.index(4)    ## indexing
"z" in x      ## looking for stuff
x.count(4)    ## count
```

```
## 4
## ('a', 3)
## (1, 4, 'a', 3, 3)
## (1, 4, 3, 'a', 3)
```

- you *can't* modify the existing tuple at all (deletion, modification)
- unpacking: `x,y,z = t`
- swapping: `(a,b) = (b,a)`
- useful as the return value of functions; safe, and can be unpacked
- convert to/from lists (`tuple()`, `list()`)

```
x = (1,2,3)
def modify(x):
    y = list(x)
    y[0] = "a"
    return(tuple(y))

print(modify(x))
print(x)
```

```
## ('a', 2, 3)
## (1, 2, 3)
```

Dictionaries

- **unordered**
- efficient: *hashing*

```
d = {"A":1,"B":2,"C":3}
empty = {}  ## empty dictionary
print(d["A"])
## d[1] won't work
print("A" in d)
print(1 in d)
print(d.values())
print(d.keys())
x = (("A",1),("B",2))
dict(x)
```

```
## 1
## True
## False
## dict_values([1, 3, 2])
## dict_keys(['A', 'C', 'B'])
```

- can add and remove entries from a dictionary

```
d = {"A":1,"B":2,"C":3}
d["D"]=5
del d["A"]
d.pop("C")
```

- alternate syntax:

```
dict(A=1,B=2,C=3)
```

(**only** works if keys can be represented as a Python symbol) - loop over *keys* in a dictionary:

```
d = dict(A=1,B=2,C=3)
for i in d.keys():
    ## do something
```

dictionary inversions

1. Simplest/most straightforward way to invert a dictionary:

```
d = {"A":1,"B":2,"C":3}
inv = {}
for k in d.keys():
   inv[d[k]] = k
```

2. Alternative:

```
d = {"A":1,"B":2,"C":3}
inv = {}
for a,b in d.items():
   inv[b] = a
```

3. Fancier (build up a list):

```
d = {"A":1,"B":2,"C":3}
invtuple = ()
for a,b in d.items():
   invtuple += ((b,a),)
inv = dict(invtuple)
```

Benford's Law, Roman numerals examples?