Complex Python data structures

Reuven M. Lerner, PhD
reuven@lerner.co.il

1

Data structures

- When presented with a problem, carefully consider how you will structure your data
  - This can mean the difference between your work being incredibly easy or incredibly difficult
- Lists of dictionaries and dictionaries of dictionaries are common — use them!

Lists of lists

```
p1 = ['Reuven', 'Lerner', '054-496-8405']

p2 = ['Atara', 'Lerner-Friedman',
'054-123-4567']

people = [p1, p2]
```

3

Lists of tuples

```
p1 = ('Reuven', 'Lerner', '054-496-8405')

p2 =('Atara', 'Lerner-Friedman',
'054-123-4567')

people = [p1, p2]
```

Lists?

- My general rule: If I need to store or retrieve with a numeric index, then a list is probably the wrong data structure

- (I’m all in favor of using loops to iterate over the elements of a list, of course!)

Dict to the rescue!

- Better semantics
- Easier searching
- Easier to add and remove fields
- Easier to make records different
- But yes, it consumes more memory

6

Lists of dicts

```
p1 = {'first_name':'Reuven',
'last_name':'Lerner',
'phone':'054-496-8405'}

p2 = {'first_name':'Atara',
'last_name':'Lerner-Friedman',
'phone':'054-123-4567'}

people = [p1, p2]
```

Dicts of dicts

```
p1 = {'first_name':'Reuven',
'last_name':'Lerner',
'phone':'054-496-8405'}

p2 = {'first_name':'Atara',
'last_name':'Lerner-Friedman',
'phone':'054-123-4567'}

people = {123: p1, 456: p2}
```