CS 150 - Dictionaries & Sets - Ch. 12

Announcements

- Email with HW6 grades
  (email only went to 1 person
   in a group)

- HW9 will be up after class
covers Ch10, recursion (Ch11),
  + 1 question on dictionaries

- Final exam: May 9 (?)
Containers

Any object which provides support for managing a collection.

In Python, each supports:

- for element in data
- element in data
- len(data)

Ex: lists & tuples, SortedSet
Issues

* Order - is the data ordered?

* Mutability - will the container be modified? Will the objects inside be modified?

* Other associated data
- language helper was just words
- storing a dictionary - need definitions

* Heterogeneous versus homogeneous
- lists are heterogeneous
Dictionaries

Maps keys to associated values.
If the keys are integers, this is a list (or tuple).

Ex: groceries = ["milk", "eggs", "tea"]

groceries[0] ➝ "milk"

groceries[1] ➝ "eggs"

key ➝ value
More general examples

key or movie

director ['Star Wars'] → 'George Lucas'
director ['The Godfather'] → 'Francis Ford Coppola'
director ['The Princess Bride'] → 'Rob Reiner'

Here, the keys are more general identifiers.
Keys
- Required to be unique.

- Can be a tuple to allow overlap:
  Ex.: director [ ("Shaft", 1971) ]
       versus
director [ ("Shaft", 2000) ]

- Often a unique # such as SS/N or ISBN.

- Keys must be immutable.
  (data can be mutable)
Python's dict class

Ex: `director = dict()
director["Star Wars"] = "George Lucas"
director["The Godfather"] = "Francis Ford Coppola"
director["The Princess Bride"] = "Rob Reiner"
print director["Star Wars"]
print director["The Hobbit"]
Another way

can initialize with \( \mathcal{E} \):

director = \( \mathcal{E} \)

Also put pairs in when initializing:

dnaToRna = \{ A: \{ U \}, C: \{ G \}, G: \{ C \}, T: \{ A \} \}

key value
Syntax
See p. 484
- d[k]
- d[k] = value
- k in d
- len(d)
- d.clear()
- d.pop(k)

- d.keys() (returns lists)
- d.values()
- d.items() (returns a list of tuples)
for k in d:
Ex:

```python
titles = director.keys()
# returns a list of movie titles
```
```
titles. sort()   # couldn't sort dict
```
```
for move in titles:
  print move, 'was directed by',
  director [move]
```
Note:

Dictionaries are many-to-one:

A single movie has only one director,

(but a director can direct many movies)

Dictionaries look up based on the unique key only.

(Going the other way is called a reverse dictionary - see 12.3.3)
Another example: Sets

An unordered collection of unique elements

Allows

- containment queries: ‘red’ in colors (which is very efficient)
- order is arbitrary
- elements in set are immutable
Ex:  colors = set()
colors.add('red')
colors.add('blue')

Also - can send in a starter value:
myset = set([1, 2, -3, 5, 2])

letters = set('this is a test')
Operations (p. 411 - 413)

- `s.add(v)`
- `s.remove(v)`
- `len(s)`
- `v in s`
- `for v in s`
- `s == t`
- `s <= t`
- `s.union(t)`
- `s.intersection(t)`
- `s - t`
- `s -= t`
- `s.update(t)`

return new set

set1: 5, 1, 3
set2: 2, 6, 3

3 lexicographical comparison

Intersection
Practice 12.1

Assume dict. values didn’t exist.

Write a code fragment that produces a list of all the values in a dictionary.