Lesson objectives

1. Describe the characteristics of the *dictionary* data structure in Python
2. Perform basic operations with dictionaries including creation, copying, updating, and traversing
3. Use dictionaries in functions

The *dictionary* data structure

- In Python, a *dictionary* is mapping between a set of indices (**keys**) and a set of **values**
  - The items in a dictionary are key-value pairs

The *dictionary* data structure

- Keys can be any Python data type
  - Because keys are used for indexing, they should be immutable
- Values can be any Python data type
  - Values can be mutable or immutable

Creating a dictionary

```
eng2sp = dict()
print eng2sp

eng2sp['one'] = 'uno'
print eng2sp

eng2sp['two'] = 'dos'
print eng2sp
```

Creating a dictionary

```
eng2sp = {'one': 'uno', 'two': 'dos',
          'three': 'tres'}
print eng2sp
```

- In general, the order of items in a dictionary is unpredictable
- Dictionaries are indexed by keys, not integers

Dictionary indexing

```
print eng2sp['three']
```

```
print eng2sp['five']
```

* If the index is not a key in the dictionary, Python raises an error.

Dictionary indexing

```
if 'five' in eng2sp:
    print eng2sp['five']

print eng2sp.get('five')
```

The `in` operator

- Note that the `in` operator works differently for dictionaries than for other sequences
  - For offset indexed sequences (strings, lists, tuples), `x in y` checks to see whether `x` is an item in the sequence
  - For dictionaries, `x in y` checks to see whether `x` is a key in the dictionary

Keys and values

- The `keys` method returns a list of the keys in a dictionary

```
print eng2sp.keys()
```

- The `values` method returns a list of the values

```
print eng2sp.values()
```

Keys and values

- The `items` method returns a list of tuple pairs of the key-value pairs in a dictionary

```
print eng2sp.items()
```

Tuple data structure

- tuples are constructed by the comma operator (not within square brackets), with or without enclosing parentheses.

```
t = 4,5,6
print t
```

- A single element tuple must have a trailing comma, such as (d,).

Tuple data structure

- tuples are very similar to lists, but they are immutable: items in a tuple cannot be changed.
- tuple elements are accessed by index, or by simultaneous assignment:

```
print t[0]

a,b,c = t # unpacking a tuple
```