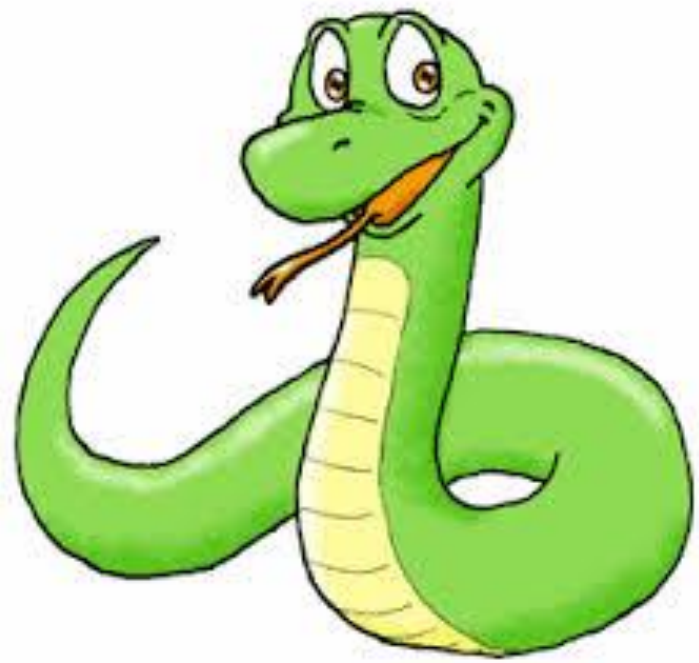
Application of GIS with Python

Chapter 6:Dictionaries



https://www.python.org/
http://www.tutorialspoint.com/python/

Table of Content

- Built in data type as a mapping between a set of indices and a set of values.
- Set of indices are called **keys** which are to be unique, can be any immutable type.
- Each key maps to a **value,** this association of a key to a value is called a **key-value pair** or an item. Unordered set of key: value pairs
- A pair of braces creates an empty dictionary: {}.
- Placing a comma-separated list of key: value pairs within the braces {} adds initial key: value pairs to the dictionary; this is also the way dictionaries are written on output

```
>>> eng2sp = dict()
>>> print (eng2sp)
{}
```

```
>>> tel = {'jack': 4098, 'sape': 4139}
>>> tel['guido'] = 4127
>>> tel
{'sape': 4139, 'guido': 4127, 'jack': 4098}
>>> tel['jack']
4098
```

Dictionaries

- To illustrate, the following examples all return a dictionary equal to {"one": 1, "two": 2, "three": 3}:

```
>>> a = dict(one=1, two=2, three=3)   #using dict function
>>> b = {'one': 1, 'two': 2, 'three': 3}
>>> c = dict(zip(['one', 'two', 'three'], [1, 2, 3])) #using zip function
>>> d = dict([('two', 2), ('one', 1), ('three', 3)])    #using tuples
>>> e = dict({'three': 3, 'one': 1, 'two': 2})
>>> a == b == c == d == e
True
```

Operators on Dictionaries

Operator		Explanation
len(d)	>>>dict = {'Name': 'Zara', 'Age': 7} >>>print "Length : %d" % len (dict) Length : 2	returns the number of stored entries, i.e. the number of (key,value) pairs.

del d[k]	>>> del dict['Age']
	>>> dict
	{'Name': 'Zara'}
		deletes the key k together with his value

k in d	>>> 'Name' in dict True	True, if a key k exists in the dictionary d
k not in d	>>> 'Age' not in dict True	True, if a key k doesn't exist in the dictionary d
cmp(dict1, dict2)		returns 0 if both dictionaries are equal, -1 if dict1 < dict2 and 1 if dict1 > dic2

Dictionary methods

- **dict.clear( )** :Removes all elements of dictionary dict
  - Syntax: dict.clear()

```
>>>dict = {'Name': 'Zara', 'Age': 7};
>>>print "Start Len : %d" %  len(dict)
Start Len : 2

>>>dict.clear()
>>>print "End Len : %d" %  len(dict)
End Len : 0
```

- **dict.copy()** :Returns a shallow copy of dictionary dict

  Syntax: dict.copy()

- This method returns a shallow copy of the dictionary

```
>>>dict1 = {'Name': 'Zara', 'Age': 7};
>>>dict2 = dict1.copy()
>>>print "New Dictinary : %s" %  str(dict2)
New Dictinary : {'Age': 7, 'Name': 'Zara'}
```

- **dict.fromkeys()** :Create a new dictionary with keys from seq and values set to value.
  - Syntax: dict.fromkeys(seq, [value])

```
>>> seq = ('name', 'age', 'sex')
>>> dict = dict.fromkeys(seq)
>>> dict
{'age': None, 'name': None, 'sex': None}
>>> dict = dict.fromkeys(seq, 10)
>>> dict
{'age': 10, 'name': 10, 'sex': 10}
```

- The method **get()** returns a value for the given key. If key is not available then returns default value None.
  - Syntax :dict.get(key, default=None)

```
>>> dict = {'Name': 'Zara', 'Age': 27}
>>> dict.get('Age')
27
```

- **dict.items() :**Returns a list of dict's (key, value) tuple pairs
  - Syntax: dict.items()

```
>>> dict.items()
[('Age', 27), ('Name', 'Zara')]
```

- **dict.keys()** :Returns list of dictionary dict's keys
- **dict.values()** :Returns list of dictionary dict's values

```
>>> dict.keys()
['Age', 'Name']
>>> dict.values()
[27, 'Zara']
```

- **dict.update(dict2)** :Adds dictionary dict2's key-values pairs to dict

```
>>> w={"house":"Haus","cat":"Katze","red":"rot"}
>>> w1 = {"red":"rouge","blau":"bleu"}
>>> w.update(w1)
>>> print w
{'house': 'Haus', 'blau': 'bleu', 'red': 'rouge', 'cat': 'Katze'}
```

Updating Dictionary

- can update a dictionary by adding a new entry or a key-value pair, modifying an existing entry, or deleting an existing entry

```
>>> dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
>>> dict['Age'] = 8; # update existing entry
>>> dict['School'] = "DPS School" # Add new entry
>>> dict
{'School': 'DPS School', 'Age': 8, 'Name': 'Zara', 'Class': 'First'}
```

- Dictionary can be updating operating through its functions and methods