Python Basics

dictionary, mutable arguments

CS101 Lecture #11

2016-10-31

Administrivia

Midterm instructions +

- Practice midterm on blackboard.

Library Functions

import

- Python has built-in functions
  - abs, type, len
- There are also specialized libraries
  - math, numpy, scipy, matplotlib

```
import math
math.sin(2*math.pi)

from math import sin, pi
sin(2*pi)
```

import

- Python has built-in functions
  - abs, type, len
- There are also specialized libraries
  - math, numpy, scipy, matplotlib

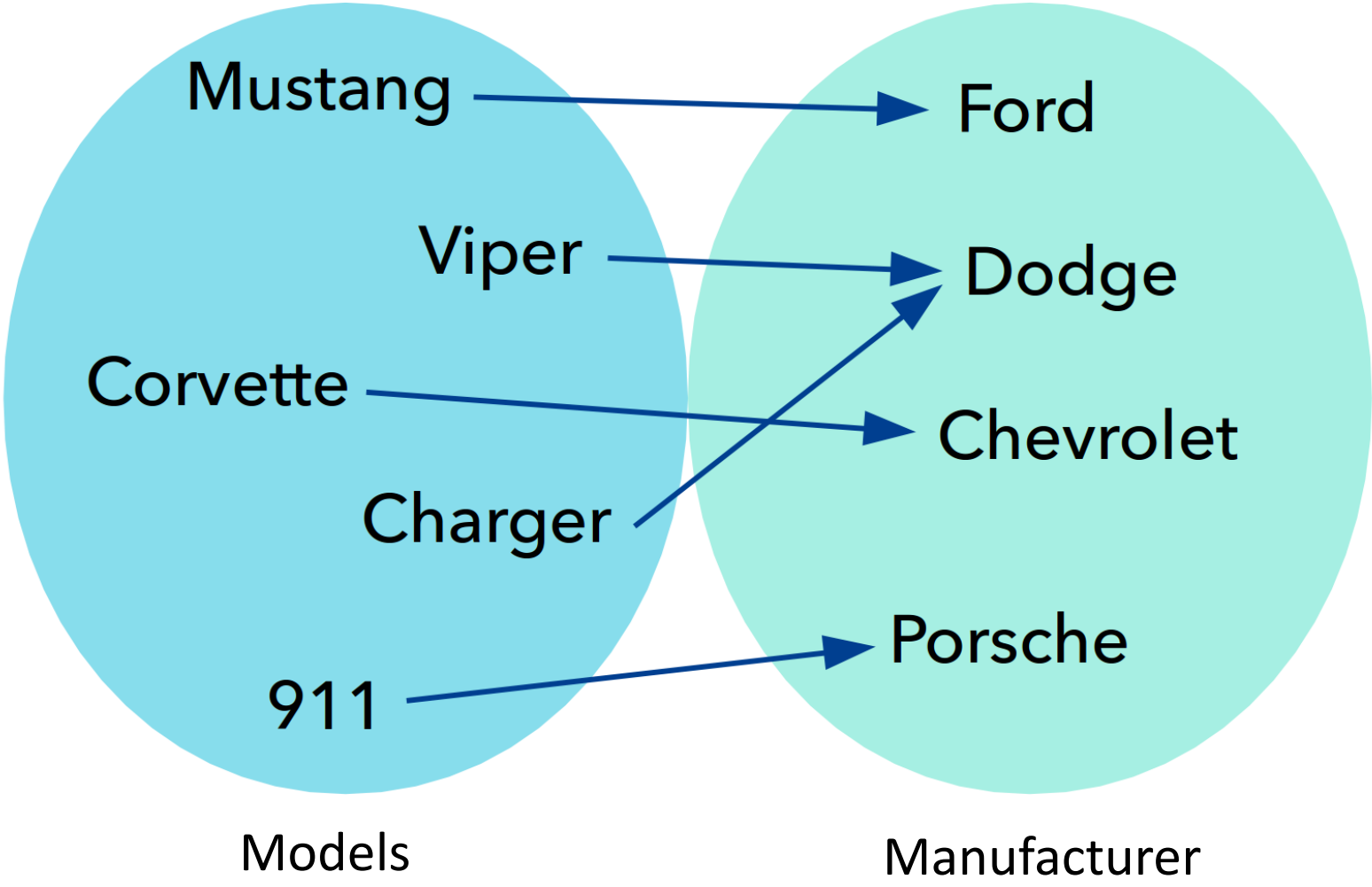
```
import numpy as np
np.random.rand(3,3)
np.random.randint(0,10)
```

Dictionary

`dict` data type

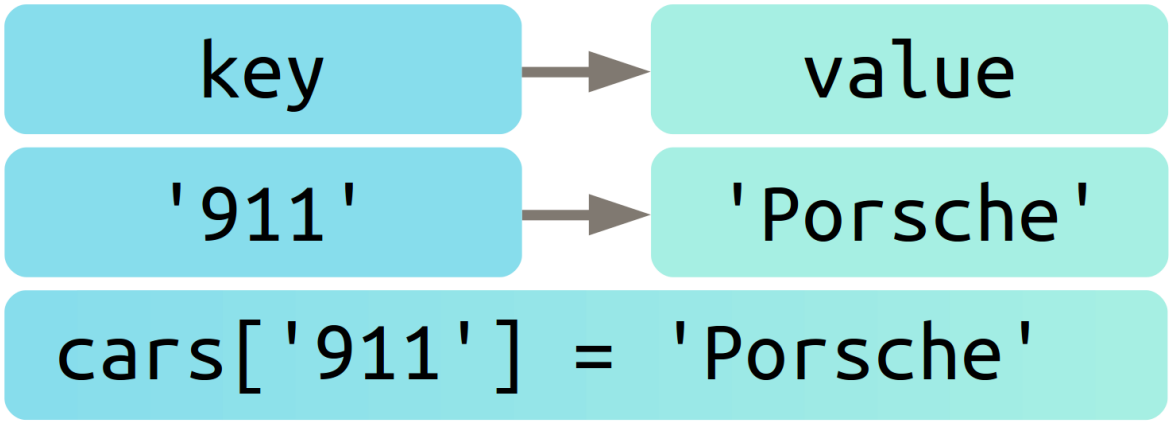
- How do we index a list?
- Lists and tuples have implicit indexing scheme
  - 0,1,2,…
- How else would you like to organize data?

Example


Mustang
Viper
Corvette
Charger
911
Ford
Dodge
Chevrolet
Porsche
Models
Manufacturer

`dict` data type

- Indexes data by specified data type (key)
- Analogy of a real dictionary (word: explanation), but can use data types besides string
- The data type maps *key* to *value*
- It is a *many-to-one* mapping



`dict` data type

```
cars = {}
cars[ 'Mustang' ]  = 'Ford'
cars[ 'Viper' ]    = 'Dodge'
cars[ 'Corvette' ] = 'Chevrolet'
cars[ 'Charger' ]  = 'Dodge'
cars[ '911' ]      = 'Porsche'
```

`dict` data type

- Create multiple items at once
- Syntax as follows:
  - Opening brace {
  - `key: value` pairs, separated by commas
  - Closing brace }

```
cars = {
    'Mustang': 'Ford',
    'Viper': 'Dodge',
    'Corvette': 'Chevrolet',
    'Charger': 'Dodge'
    '911':   'Porsche'
}
```

`dict` operations and methods

```
d = {'one':1, 'two':2, 'three':3}
print(d['one'])
d['four'] = 4          # add a new key-value pair if key
                       # doesn't exist
d['three'] = 3.0       # modify an existing pair
del d['four']
'five' in d            # False
for key in d:          # no guarantee on order
    print(key, d[key])
d.keys()
d.values()
```

Example

```
d = {1:'a', 2:'b', 3:'c', 4:'d'}
x = d[2] + d[3]
```

What is the value of x?

A 5

B 7

C 'cd'

D 'bc'

Example

```
d = {}
words = [‘red’, ‘orange’, ‘yellow’]
for word in words:
    d[word] = words.index(word)
```

What is stored in d?

`dict` Applications

- Link data based on a common field

```
zipcode = {'Bill': 60644,
           'Jill': 41073,
           'Jack': 61820,
           'Tony': 60801}
zip2city = {41073: 'Cincinnati',
            60644: 'Chicago',
            61801: 'Urbana',
            61820: 'Champaign'}

for name in zipcode:
    print(name, zip2city[ zipcode[name] ])
```

`dict` Applications

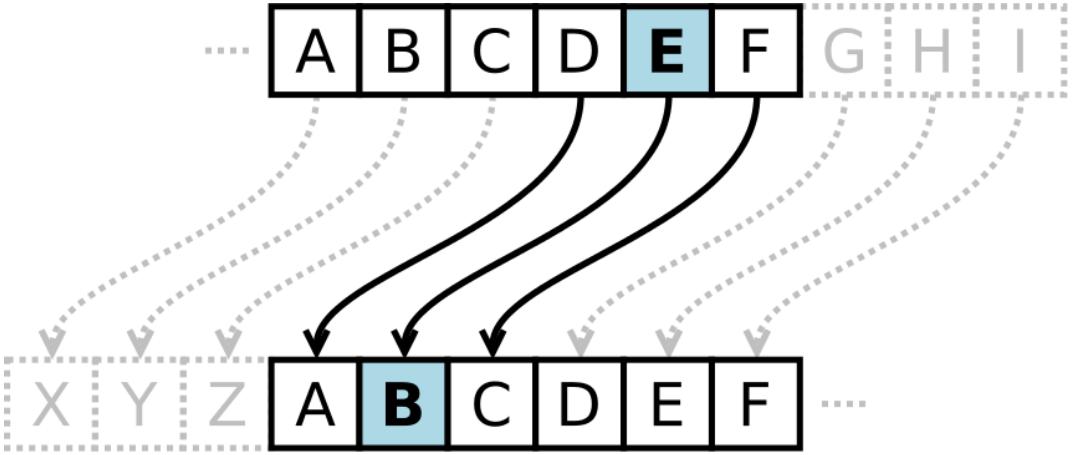
- Dictionaries can encode/decode data, or translate data from one representation to another

```
x = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
y = 'BCDEFGHIJKLMNOPQRSTUVWXYZA'

e = {}
for i in range(len(x)):
    e[x[i]] = y[i]
encoded = ''
for c in 'HELLO':
    encoded += e[c]
```

Exercise

- Encoder words in a file using a Caesar cipher



$$E_n(x) = (x + n)\%26$$

https://en.wikipedia.org/wiki/Caesar_cipher

Exercise

```
def encode_caesar(message, n):
    from string import ascii_uppercase as alphabet
    x = alphabet
    e = {}
    # initialize e (encoder)
    …

    message = message.upper()

    encoded = ''
    # encode the message
    …

    return encoded
```

Exercise

```
>>> message = 'the quick brown fox jump over the lazy dog'
>>> cipher.encode_caesar(message, 7)
"AOL XBPJR IYVDU MVE QBTWZ VCLY AOL SHGF KVF"
```

- How would you write a Caesar decoder?

`dict` Applications

- Dictionary can also be used as accumulators

```
x = 'ABBACABD'
d = {}

for c in x:
    if c not in d:
        d[c] = 1
    else:
        d[c] += 1
```

- What is the output of d by the above code?

Exercise

- Count the category frequencies in Jeopardy questions
- Find the category name with the highest frequency

```
myfile = open('jeopardy.csv')
lines = myfile.readlines()

# get category frequencies
…

# get most popular category name
…
```

Mutable Arguments

Example

```
def fun(x):
    x = x + 1
    return x

x = 5
fun(x)
```

What is the value of x in the end?

Mutable arguments

- Mutability causes lists to work differently when passed to a function
- Can be changed within a function
- Very useful

```
def fun(q,i):
    q.append(i)

a = []
for i in range(10)
    fun(a,i)
print(a)
```

Mutable arguments

```
def readfile(filename, a)
    for line in open(filename):
        a.append(line.strip())

all_lines = []
readfile('cipher.py', all_lines)
readfile('jeopardy.py', all_lines)
…
```

Mutable arguments

```
def readfile(filename, a)
    for line in open(filename):
        a.append(line.strip())

all_lines = []
for f in open('filenames.txt'):
    readfile(f.strip(), all_lines)
```

Copy mutable type by value

- Have a copy of a list, not as an alias
- Use the slice operator

```
x = [3,2,1]
y = x[:]      #slice everything
y.sort()
print(x)
```

Copy mutable type by value

- Have a copy of a list, not as an alias
- Use the slice operator

```
x = [3,2,1]
y = x[:]          #slice everything
z = x

y == x            #True
z == x            #True
```

test identity: `is`

- Have a copy of a list, not as an alias
- Use the slice operator

```
x = [3,2,1]
y = x[:]          #slice everything
z = x

y is x            #False
z is x            #True
```

Reminders

Reminders

- Practice midterm on blackboard.

- Q&A session on Tuesday (Oct 31st) afternoon
  - Venue and time to-be-determined
  - (?)Tentative