Python – General - 2

R. R. Maiti

Some operators

- ‘in’ -- 1
  - >>> 'a' in 'banana'
  - True
  - >>> 'seed' in 'banana'
  - False
- ‘in’ – 2
  - def any_lowercase1(s):
    - for c in s:
      - if c.islower():
        - return True
      - else:
        - return False

```
>>> s = 'abc'
>>> t = [0, 1, 2]
>>> zip(s, t)
[('a', 0), ('b', 1), ('c', 2)]
```

List

- List
  - nums = [10, 20, 30, 40]
  - strs = ['crunchy frog', 'ram bladder', 'lark vomit']
  - mixs = ['spam', 2.0, 5, [10, 20]]
  - empty = []
- Operations on list
  - cheeses = ['Cheddar', 'Edam', 'Gouda']
  - >>> 'Edam' in cheeses
  - True
  - >>> 'Brie' in cheeses
  - False

  - a = [1, 2, 3]
  - b = [4, 5, 6]
  - c = a + b
  - print c
  - print sum(a)

  - >>>[0] * 4
  - [0, 0, 0, 0]
  - >>> [1, 2, 3] * 3
  - [1, 2, 3, 1, 2, 3, 1

```
t = ['a', 'b', 'c']
>>> t.append('d')
>>> print t
['a', 'b', 'c', 'd']
>>> x = t.pop(1)
>>> y = t.remove('c') #if element is known
```

```
t = ['a', 'b', 'c', 'd', 'e', 'f']
>>> t[1:3]
['b', 'c']
>>> t[:4]
['a', 'b', 'c', 'd']
>>> t[3:]
['d', 'e', 'f']
```

```
t = ['a', 'b', 'c', 'd', 'e', 'f']
>>> t[1:3] = ['x', 'y']
>>> print t
['a', 'x', 'y', 'd', 'e', 'f']
```

```
t1 = ['a', 'b', 'c']
>>> t2 = ['d', 'e']
>>> t1.extend(t2)
>>> print t1
['a', 'b', 'c', 'd', 'e']
```

- Traverse a list
  - for cheese in cheeses:
    - print cheese
  - for i in range(len(numbers)):
    - numbers[i] = numbers[i] * 2

Dictionary

- eng2sp = dict()
- print eng2sp
- eng2sp['one'] = 'uno‘
- print eng2sp
- eng2sp = {'one': 'uno', 'two': 'dos', 'three': 'tres'}
- print eng2sp['two']
- len(eng2sp)
- vals = eng2sp.values()
- 'uno' in vals
- h = histogram('brontosaurus')

- def histogram(s):
  - d = dict()
  - for c in s:
    - if c not in d:
      - d[c] = 1
    - else:
      - d[c] += 1
  - return d

- def reverse_lookup(d, v):
  - for k in d:
    - if d[k] == v:
      - return k
  - raise ValueError

- def print_hist(h):
  - for c in h:
    - print c, h[c]

- def invert_dict(d):
  - inverse = dict()
  - for key in d:
    - val = d[key]
    - if val not in inverse:
      - inverse[val] = [key]
    - else:
      - inverse[val].append(key)
  - return inverse

- hist = histogram('parrot')
- print hist
- inverse = invert_dict(hist)
- print inverse

Tuple

- t = ('a', 'b', 'c', 'd', 'e')
- t = tuple('lupins')
- print t
- print t[1:3]

```
>>> temp = a
>>> a = b
>>> b = temp
```

```
>>> a, b = b, a
```

```
>>> addr = 'monty@python.org'
>>> uname, domain = addr.split('@')
```

- t = [('a', 0), ('b', 1), ('c', 2)]
- for letter, number in t:
- print number, letter

```
>>> t = [('a', 0), ('c', 2), ('b', 1)]
>>> d = dict(t)
>>> print d
```

```
>>> t = divmod(7, 3)
>>> print t
```

```
>>> d = dict(zip('abc', range(3)))
>>> print d
```

- for key, val in d.items():
- print val, key

```
>>> (0, 1, 2) < (0, 3, 4)
True
>>> (0, 1, 2000000) < (0, 3, 4)
True
```

Class Assignment - 1

Write a program that reads a word list from a file (see Section 9.1) and prints all the sets of
words that are anagrams.
Here is an example of what the output might look like:
['deltas', 'desalt', 'lasted', 'salted', 'slated', 'staled']
['retainers', 'ternaries']
['generating', 'greatening']
['resmelts', 'smelters', 'termless']

Files

- >>> fout = open('output.txt', 'w')
- >>> print fout
- >>> line1 = "This here's the wattle,\n"
- >>> fout.write(line1)
- >>> line2 = "the emblem of our land.\n"
- >>> fout.write(line2)
- >>> fout.close()

```
>>> camels = 42
>>> 'I have spotted %d camels.' % camels
'I have spotted 42 camels.'

>>> 'In %d years I have spotted %g %s.' % (3, 0.1, 'camels')
```

```
>>> import os
>>> cwd = os.getcwd()
>>> print cwd
```

```
>>> os.path.abspath('memo.txt')
'/home/dinsdale/memo.txt'
>>> os.path.exists('memo.txt')
True
```

- try:
  - fin = open('bad_file')
  - for line in fin:
    - print line
  - fin.close()
- except:
  - print 'Something went wrong.'

XOR – on binary string

```
def xor(a, b, n):
    ans = ""

    # Loop to iterate over the
    # Binary Strings
    for i in range(n):

        # If the Character matches
        if (a[i] == b[i]):
            ans += "0"
        else:
            ans += "1"
    return ans
```

```
# Driver Code
if __name__ == "__main__":
    a = "1010"
    b = "1101"
    n = len(a)
    c = xor(a, b, n)
    print(c)
```

Class Assignment-2

- Write a program to encrypt and decrypt a text file using simple XOR.
  - def encrypt(key, inputfile, outputfile):
    - #code to encrypt using XOR
    - # C = P^K
  - def decrypt(key, inputfile, outputfile):
    - #code to decrypt using XOR
    - # P = C ^ K
  - Def verify(file1, file2):
    - # verify if two files contain the same content
- Put appropriate guard within these functions to handle I/O exceptions.

Home Assignment 2

- Consider a network of IP addresses, both private and public addresses.
- Let IP addresses are given in a .csv file in the following format
  - IP1, IP2, Pkt-Size, Trans-Port
- Consider a graph representation of your choice in Python.
- Compute number of private addresses that connects to each of public IP addresses and store the information in the following format.
  - IP address, Pkt-Count
- Write a function to create a sample input file in the said format.