10.009 The Digital World

Term 3. 2016

Problem Set 6 (for Week 6)

Last update: January 12, 2016

Due dates:

- **Problems: Cohort sessions**: Following week: Monday 11:59pm.
- **Problems: Homework**: Same as for the cohort session problems.
- **Problems: Exercises**: These are practice problems and will not be graded. You are encouraged to solve these to enhance your programming skills. Being able to solve these problems will likely help you prepare for the midterm examination.

Objectives:

1. Learn to manipulate strings.
2. Learn to read and write files.

Note: Solve the programming problems listed below using the IDLE or Canopy editor. Make sure you save your programs in files with suitably chosen names and in an newly created directory. In each problem find out a way to test the correctness of your program. After writing each program, test it, debug it if the program is incorrect, correct it, and repeat this process until you have a fully working program. Show your working program to one of the cohort instructors.

Problems: Cohort sessions

1. *Strings:* Define a function reverse(s) that computes the reversal of a string. For example, `reverse('I am testing')` should return the string `'gnitset ma I'`. Use iteration for this exercise.

2. *Strings: Check password:* Some web sites impose certain rules for passwords. Write a function that checks whether a string is a valid password. Return True if it is a valid password and False otherwise. The password rules are as follows:

   - A password must have at least 8 characters.
   - A password must consist of only letters and digits.
   - A password must contain at least two digits.

   To test:

```
print "isValidPassword('test')"
ans=isValidPassword('test')
print ans

print "isValidPassword('testtest')"
ans=isValidPassword('testtest')
print ans

print "isValidPassword('testt22')"
ans=isValidPassword('testt22')
print ans

print "isValidPassword('testte22')"
ans=isValidPassword('testte22')
print ans
```

   The output should be:

```
isValidPassword('test')
False
isValidPassword('testtest')
False
isValidPassword('testt22')
False
isValidPassword('testte22')
True
```

3. *Strings: Common prefix:* Write a function that returns the longest common prefix of two strings. For example, the longest common prefix of `distance` and `disinfection` is `dis`. The header of the method is `def prefix(s1,s2)`. If the two strings have no common prefix, the method returns an empty string.

   To test:

```
print "prefix('distance','disinfection')"
ans=prefix('distance','disinfection')
print ans

print "prefix('testing','technical')"
ans=prefix('testing','technical')
print ans

print "prefix('drinking','drinker')"
ans=prefix('drinking','drinker')
print ans

print "prefix('rosses','crosses')"
ans=prefix('rosses','crosses')
print ans

print "prefix('distancetion','distance')"
ans=prefix('distancetion','distance')
print ans
```

The output should be:

```
prefix('distance','disinfection')
dis
prefix('testing','technical')
te
prefix('drinking','drinker')
drink
prefix('rosses','crosses')

prefix('distancetion','distance')
distance
```

4. *Files and I/O:* A file named `xy.dat` contains two columns of numbers, corresponding to the `x` and `y` coordinates on a curve. The start of the file looks as follows:

   -1.0000 -0.0000

   -0.9933 -0.0087

   -0.9867 -0.0179

   -0.9800 -0.0274

   -0.9733 -0.0374

   Write a function named `read2columns(f)` with argument `f` of file object (e.g., `f=open('xy.dat','r')`). The function should read the first column as an `x` coordinate and the second column as a `y` coordinate. Note that the two columns are separated by spaces. You then need to create a new data of the type Coordinate defined as follows:

```
class Coordinate:
    x=0
    y=0
```

   The function returns two coordinates, one has the maximum magnitude, and the other one has the minimum magnitude. The magnitude is defined as:

$$|p| = \sqrt{(p.x)^2 + (p.y)^2} \tag{1}$$

where p is of the type Coordinate as defined above.

To test:

```
f=open('xy.dat','r')
pmax,pmin=read2columns(f)
print 'max: (%f, %f)'%(pmax.x,pmax.y)
print 'min: (%f, %f)'%(pmin.x,pmin.y)
```

It should output the following:

```
max: (-1.000000, -0.000000)
min: (0.000000, 0.000000)
```

5. *File: Replace text* Write a function that reads in a file object and replaces text inside the opened file. The header of the function should takes in the file object, an old string, and a new string. The function should return the new text as a string where the new string input argument replaces the old string in the text. Use file 'replace.txt' to test.

Problems: Homework

1. *Strings: Binary to decimals:* Write a function that parses a binary number as a string into a decimal integer. Use the function header `def binaryToDecimal(binaryString):`. You can assume that the string is never empty.

```
>>> print binaryToDecimal('100')
4
>>> print binaryToDecimal('101')
5
>>> print binaryToDecimal('10001')
17
>>> print binaryToDecimal('10101')
21
```

2. *Strings:* Write a function named `uncompressed()` that takes a compressed string as input and outputs an uncompressed string, where each alphabetic character is preceded by a single digit indicating the number of times that the character should be entered in the uncompressed version of the string. For example:

   - The compressed string `2a5b1c` is uncompressed to `aabbbbbc`
   - The compressed string `1a1b2c` is uncompressed to `abcc`
   - The compressed string `1a9b3b1c` is uncompressed to `abbbbbbbbbbbbc`

3. *Strings: DNA:* Write a function named `getBaseCounts2()` by modifying `getBaseCounts()` that you wrote in homework problem set 4. `getBaseCounts2()` takes a string as input. The input string may contain letters other than A, C, G, and T. The function should return the counts of only A, C, G, and T in the form of a dictionary; it must ignore all letters other than A, C, G, and T. For any input string with lower case alphabets, the function will still return 'The input DNA string is invalid'.

4. *Files and I/O:* A file named `constants.txt` contains a table of the values and the dimensions of some fundamental constants from physics. The file looks as follows:

name of constant	value	dimension
speedoflight	299792458.0	m/s
gravitationalconstant	6.67259e-11	m**3/kg/s**2
Planckconstant	6.6260755e-34	J*s
elementarycharge	1.60217733e-19	C
Avogadronumber	6.0221367e23	1/mol
Boltzmannconstant	1.380658e-23	J/K
electronmass	9.1093897e-31	kg
protonmass	1.6726231e-27	kg

   We want to load this table into a dictionary named constants, where the keys are the names of the constants. For example, `constants['gravitational constant']` holds the value

of the gravitational constant ($6.67259 * 10^{-11}$). Write a function named `fundamentalConstants(f)` that reads and interprets the text in the file, and returns the dictionary.

Test Cases:

Input: f = open('constants.txt','r')
Output: {'Boltzmann constant': 1.380658e-23, 'speed of light': 299792458.0, 'proton mass': 1.6726231e-27, 'gravitational constant': 6.67259e-11, 'Avogadro number': 6.0221367e+23, 'elementary charge': 1.60217733e-19, 'Planck constant': 6.6260755e-34, 'electron mass': 9.1093897e-31}

5. *Files: Process scores:* Suppose that a text file contains an unspecified number of scores. Write a function that reads the scores from an object file and returns their total and average. Scores are separated by blanks. Use 'scores.txt' for testing.

Problems: Exercises

1. An anagram is a type of word play, the result of rearranging the letters of a word or phrase to produce a new word or phrase, using all the original letters exactly once; e.g., orchestra = carthorse. Using the word list at http://www.puzzlers.org/pub/wordlists/unixdict.txt, write a function `findAnagram(f)` that returns the sets of words in a nested list that share the same characters that contain the most words in them.

   Test Cases:

   Input: file= unixdict.txt
   Output: [['abel', 'able', 'bale', 'bela', 'elba'], ['alger', 'glare', 'lager', 'large', 'regal'], ['angel', 'angle', 'galen', 'glean', 'lange'], ['evil', 'levi', 'live', 'veil', 'vile'], ['caret', 'carte', 'cater', 'crate', 'trace'], ['elan', 'lane', 'lean', 'lena', 'neal']]

2. A sentence splitter is a program capable of splitting a text into sentences. The standard set of heuristics for sentence splitting includes (but is not limited to) the following rules: Sentence boundaries occur at one of "." (periods), "?" or "!", except that

   (a) Periods followed by whitespace followed by a lower case letter are not sentence boundaries.

   (b) Periods followed by a digit with no intervening whitespace are not sentence boundaries.

   (c) Periods followed by whitespace and then an upper case letter, but preceded by any of a short list of titles are not sentence

   (d) boundaries. Sample titles include Mr., Mrs., Dr., and so on.

   (e) Periods internal to a sequence of letters with no adjacent whitespace are not sentence boundaries (for example, www.aptex.com, or e.g).

   (f) Periods followed by certain kinds of punctuation (notably comma and more periods) are probably not sentence boundaries.

Your task here is to write a function `senSplit(f)` that given a file object is able to write its content in a list with each sentence as a string item in a list. Test your function with the following short text: *Mr. Smith bought cheapsite.com for 1.5 million dollars, i.e. he paid a lot for it. Did he mind? Adam Jones Jr. thinks he didn't. In any case, this isn't true... Well, with a probability of .9 it isn't.* The result should be:

```
['Mr.  Smith bought cheapsite.com for 1.5 million dollars, i.e.  he paid a lot
for it.',
'Did he mind?',
'Adam Jones Jr.  thinks he didn't.',
'In any case, this isn't true...',
'Well, with a probability of .9 it isn't.']
```

Test Cases:

Test case 1

Input: filename= fileEx2_1.txt

Output: outp = ['Mr. Smith bought cheapsite.com for 1.5 million dollars, i.e. he paid a lot for it.', 'Did he mind?', 'Adam Jones Jr. thinks he didn\xe2\x80\x99t.', 'In any case, this isn\xe2\x80\x99t true...', 'Well, with a probability of .9 it isn\xe2\x80\x99t.']

Test case 2

Input: filename= fileEx2_2.txt

Output: ['Lorem ipsum dolor sit amet, consectetur adipiscing elit.', 'Donec sed nisl eu nulla scelerisque fermentum.', 'Nulla facilisi.', 'Suspendisse hendrerit quam in dui sollicitudin egestas.', 'Nullam euismod massa at nisi luctus dictum tristique nibh hendrerit.', 'Fusce eleifend sollicitudin sapien, nec imperdiet erat vestibulum et.', 'Phasellus vitae leo neque, ut bibendum orci.', 'Nullam a velit sit amet erat auctor adipiscing.', 'Maecenas sollicitudin dui sagittis nisi dapibus a rhoncus nisi dictum.', 'Suspendisse potenti., Phasellus nec risus eget nisl sodales lobortis.', 'Lorem ipsum dolor sit amet, consectetur adipiscing elit.', 'Aliquam sodales, justo vitae blandit viverra, est est lacinia tellus, eget euismod sem dui non felis.', 'In est lorem, eleifend eget blandit ac, lobortis et neque.', 'Etiam porttitor, justo quis cursus semper, elit odio semper augue, a suscipit lacus mauris et ipsum.', 'Nulla nec quam ante.', 'Phasellus diam quam, porta id eleifend sit amet, vestibulum eu ligula.']

Test case 3

Input: filename= fileEx2_3.txt

Output: outp = ['[32] But I must explain to you how all this mistaken idea of denouncing of a pleasure and praising pain was born and I will give you a complete account of the system, and expound the actual teachings of the great explorer of the truth, the master- builder of human happiness.', 'No one rejects, dislikes, or avoids pleasure itself, because it is pleasure, but because those who do not know how to pursue pleasure rationally encounter consequences that are extremely painful.', 'Nor again is there anyone who loves or pursues or desires to obtain pain of itself, because it is pain, but occasionally circumstances occur in which toil and pain can procure him some great pleasure.', 'To take a trivial example, which of us ever undertakes laborious physical exercise, except to obtain some advantage from it?', 'But who has any right to find fault with a man who chooses to enjoy a pleasure that has no annoying consequences, or one who avoids a pain that produces no resultant pleasure?']

Test case 4

Input: filename= fileEx2_4.txt

Output: outp = ['[33] On the other hand, we denounce with righteous indignation and dislike men who are so beguiled and demoralized by the charms of pleasure of the moment, so blinded by desire, that they cannot foresee the pain and trouble that are bound to ensue; and equal blame belongs to those who fail in their duty through weakness of will, which is the same as saying through shrinking from toil and pain.', 'These cases are perfectly simple and easy to distinguish.', 'In a free hour, when our power of choice is untrammeled and when nothing prevents our being able to do what we like best, every pleasure is to be welcomed and every pain avoided.', 'But in certain circumstances and owing to the claims of duty or the obligations of business it will frequently occur that pleasures have to be repudiated and annoyances accepted.', 'The wise man therefore always holds in these matters to this principle of selection: he rejects pleasures to secure other greater pleasures, or else he endures pains to avoid worse pains.']

Test case 5

Input: filename= fileEx2_5.txt

Output: outp = ['In publishing and graphic design, lorem ipsum[1] is placeholder text (filler text) commonly used to demonstrate the graphic elements of a document or visual presentation, such as font, typography, and layout, by removing the distraction of meaningful content.', 'The lorem ipsum text is typically a section of a Latin text by Cicero with words altered, added and removed that make it nonsensical in meaning and not proper Latin.']

End of Problem Set 6.