**Text Files**

There is a ton of interesting data to be found on the internet stored in text files. In this chapter we

will learn how to work with data stored in text files.

**Reading from files**

Suppose we have a text file called example.txt whose contents are shown below, and we want to

read its contents into Python. There are several ways to do so. We will look at two of them.

Hello.

This is a text file.

Bye!

1. The first way to read a text file uses a list comprehension to load the file line-by-line into a

list:

lines = [line.strip() **for** line **in open**('example.txt')]

The list lines is now

['Hello.', 'This is a text file.', 'Bye!']

The string method strip removes any whitespace characters from the beginning and end of

a string. If we had not used it, each line would contain a newline character at the end of the

line. This is usually not what we want.

Note: strip removes whitespace from both the beginning and end of the line. Use rstrip

if you need to preserve whitespace at the beginning of the line.

2. The second way of reading a text file loads the entire file into a string:

s = **open**('example.txt').read()

The string s is now

'Hello.\nThis is a text file.\nBye!'

**Directories**

Say your program opens a file, like below:

s = **open**('file.txt').read()

The file is assumed to be in the same directory as your program itself. If it is in a different directory,

then you need to specify that, like below:

s = **open**('c:/users/heinold/desktop/file.txt').read()

**Writing to files**

There are also several ways to write to files. We will look at one way here. We will be writing to a

file called writefile.txt.

f = **open**('writefile.txt', 'w')

**print**('This is line 1.', file=f)

**print**('This is line 2.', file=f)

f.close()

We first have to open the file. That is what the first line does, with the 'w' indicating that we want

to be able to write to the file. Python creates what is called a file object to represent the file, and we

give that object the name f. This is what we use to refer to the file. To write to the file, we use the

print statement with the optional file argument that specifies the file to write to. When we are

done writing, we should close the file to make sure all of our changes take. Be careful here because

if writefile.txt already exists, its contents will be overwritten.

**Examples**

**Example 1** Write a program that reads a list of temperatures from a file called temps.txt, converts

those temperatures to Fahrenheit, and writes the results to a file called ftemps.txt.

file1 = **open**('ftemps.txt', 'w')

temperatures = [line.strip() **for** line **in open**('temps.txt')]

**for** t **in** temperatures:

**print**(t\*9/5+32, file=file1)

file1.close()

**Example 2** In earlier sections we wrote a simple quiz game. The questions and answers were both

contained in lists hard-coded into the program. Instead of that, we can store the questions and

answers in files. That way, if you decide to change the questions or answers, you just have to

change their files. Moreover, if you decide to give the program to someone else who doesn’t know

Python, they can easily create their own lists of questions and answers. To do this, we just replace

the lines that create the lists with the following:

questions = [line.strip() **for** line **in open**('questions.txt')]

answers = [line.strip() **for** line **in open**('answers.txt')]

**Example 3** Say you have a text file that contains the results of every 2009-10 NCAA basketball

game. (You can find such a file at www.kenpom.com.) A typical line of the file looks like this:

02/27/2010, Robert Morris, 61, Mount St. Mary's, 63

Below is a program that scans through the file to find the most lopsided game, the one where the

winning team had the largest margin of victory.

lines = [line.strip() **for** line **in open**('scores.txt')]

games = [line.split(',') **for** line **in** lines]

**print**(**max**([**abs**(**int**(g[2])-**int**(g[4])) **for** g **in** games]))

We use the split method to break each line into a lists of its component parts. The scores are at

indices 2 and 4. To find the maximum difference, we can use a list comprehension to create a list of

all the margins of victories and use **max** to find the maximum.

The maximum turns out to be 84. Unfortunately, the method above does not tell us anything else

about the game. In order to do that, we resort to the longer way to find maximums, described in

earlier Sections. This allows us to store information about the game as we search for the largest margin

of victory.

lines = [line.strip() **for** line **in open**('scores.txt')]

games = [line.split(',') **for** line **in** lines]

biggest\_diff = 0

**for** g **in** games:

diff = **abs**(**int**(g[2])-**int**(g[4]))

**if** diff>biggest\_diff:

biggest\_diff = diff

game\_info = g

**print**(game\_info)

['12/03/2009', ' SalemInternational', '35', ' Marshall', '119']

**Wordplay**

If you like words, you can have a lot of fun with a wordlist, which is a text file where each line

contains a different word. A quick web search will turn up a variety of different wordlists, ranging

from lists of common English words to lists containing practically every English word.

Assuming the wordlist file is wordlist.txt, we can load the words into a list using the line

below.

wordlist = [line.strip() **for** line **in open**('wordlist.txt')]

**Example 1** Print all three letter words.

**for** word **in** wordlist:

**if len**(word)==3:

**print**(word)

Note that this and most of the upcoming examples can be done with list comprehensions:

**print**([word **for** word **in** wordlist **if len**(word)==3])

**Example 2** Print all the words that start with gn or kn.

**for** word **in** wordlist:

**if** word[:2]=='gn' **or** word[:2]=='kn':

**print**(word)

**Example 3** Determine what percentage of words start with a vowel.

count = 0

**for** word **in** wordlist:

**if** word[0] **in** 'aeiou':

count=count+1

**print**(100\*count/**len**(wordlist))

**Example 4** Print all 7-letter words that start with th and end in ly. Things like this are good for

cheating at crosswords.

**for** word **in** wordlist:

**if len**(word)==7 **and** word[:2]=='th' **and** word[-2:]=='ly':

**print**(word)

**Example 5** Print the first ten words that start with q.

i=0

**while** wordlist[i]!='q':

i=i+1

**print**(wordlist[i:i+10]

12.5. EXERCISES 113

Note this is not a very efficient way of doing things since we have to scan through most of the list.

A binary search would be more efficient, but the above approach still runs almost instantly even

for large files.

**Example 6** Find the longest word that can be made using only the letters a, b, c, d, and e.

largest = 0

**for** word **in** wordlist:

**for** c **in** word:

**if** c **not in** 'abcde':

**break**

**else**:

**if len**(word)>largest:

largest=**len**(word)

largest\_word=word

**print**(largest\_word)

The way this program works is for every word in the wordlist, we use a for/else loop (Section 9.4)

to scan through the word looking checking each character to see if it is an a, b, c, d, or e. If any

letter isn’t one of these, then we break out of the loop and move on to the next word. On the other

hand, if we get all the way through the loop, then we go to else block. In that block, we use a

modification of the technique from earlier section for finding a maximum.