Data Storage and File Handling

Objectives

To use some of the Python 2 file handling methods, as well as the pickle and gzip modules.

Reference Material

Primarily Chapter 7 Data Storage and File Handling, but also Chapter 3 Flow Control and Chapter 4 String Handling. Background information on pickle and gzip is available in the online documentation.

Questions

1. Write a Python script to list all the unused port numbers in the /etc/services file between 1 and 200

   Steps:

   Become familiar with the input file - view it first

   Write the main code to read the services file one line at a time

   Use string functions to:

   Ignore lines starting with a # comment character

   Ignore lines that just consist of "white-space"

   /etc/services has several columns separated by white-space

   - Use split or a regular expression to isolate the port/protocol field
   - Use another split or regular expression to isolate the port number
   - Create a nested loop to print out all the unused numbers between the previous port number and the current one
   - Don't forget to stop at port number 200!
   - Note that many port numbers have > 1 entry

   On Windows the file is in 'C:\WINDOWS\system32\drivers\etc\services', or in 'C:\WINNT\system32\drivers\etc\services'.

   Many port numbers have more than one entry in the file, but you may assume they are in order.

   Hints:

   open the file

   Read the file line-by-line using a for loop

   Test the first character of a string using a slice

Test for white-space using *string.*isspace()

Be careful of comparing strings and int - you will have to convert the port number to an int

Points will be deducted if you forget to close the file (only kidding)

2. Using the data in **country.txt**, construct a Python dictionary where the country name is the key and the other record details are stored in a list as the value. Store (pickle) this dictionary into a file (call it country.p).

   Notice the size of the file compare to the original, and then change the program to use gzip.

3. Now write a program which reads the pickled dictionary and displays it onto the console. If time allows, convert your pickle to use a shelve.

If time allows…

4. This exercise uses **messier.txt**, which was used in a previous optional exercise (you do not need to have completed that exercise to do this one).

   This file contains details of Messier celestial objects that are identified by a Messier number, the first field in the file.

   The aim is to access the records in the file randomly, using `seek()`. Note that (with Python 2) you should open the file for binary access.

   Construct an index (could be in a list or a dictionary) which consists of the file position (use `tell()`) of each record. The key is the first field, the Messier number, which is prefixed M (ignore any lines that do not start with 'M').

   Now prompt the use to enter a Messier number, with or without the 'M', and display the record for that celestial object.

Solutions

Question 1

This solution uses split, with a nested loop to determine unused ports:

```
file = r'C:\WINDOWS\system32\drivers\etc\services'

count = 1

for line in open(file, 'r'):
   if line[0:1] != '#' and not line.isspace():
       name, pp = line.split(None, 1)
       port, protocol = pp.split ('/', 1)

       port = int(port)
       while count < port:
           print "Unused port: " + str(count)
           count += 1
           if count > 200: break

       count = int(port) + 1
       if count > 200: break
```

This solution uses regular expressions and sets. A common mistake with this approach is to forget to convert the captured port number to an int, required since `range` returns ints.

```
import re

file = r'C:\WINDOWS\system32\drivers\etc\services'

ports = set()
for line in open(file):
   m = re.search(r'(\d+)/(udp|tcp)',line)
   if m:
       port = int(m.groups()[0])
       if port > 200: break
       ports.add(port)

print set(range(1,201)) - ports
```

Questions 2 & 3

```
import pickle
import gzip

country_dict = {}

for line in open('country.txt') :
    row = line.split(',')
    name = row.pop(0)
    country_dict[name] = row

outp = gzip.open('country.p', 'wb')
pickle.dump(country_dict, outp)
outp.close()

# Using a shelve
import shelve
db = shelve.open('country')
for country in country_dict.keys():
    db[country] = country_dict[country]
db.close()

db = shelve.open('country')
print db['Belgium']
db.close()
```

If time allows…

Question 4

```
# Construct an index
Index = []
fh = open('messier.txt','rb')

while True:
    line = fh.readline()
    if not line: break

    if line.startswith('M'):
        num = line[1:6].rstrip()
        Index.append(fh.tell() - len(line))

while True:
    num = raw_input("Enter a Messier number(0 to exit):
")
    if num.startswith("M"):
        num = int(num[1:])
    else:
        num = int(num)

    if num < 1: break
    num = num - 1

    fh.seek(Index[num])
    print fh.readline()
```