Week 4
Python

22 October 2018

CS 35L Lab 4

Jeremy Rotman

Announcements

- Assignment #3 is due Friday by 11:55pm
- For Assignment #10
  - You should begin to choose stories
  - Email me to tell me what you are choosing
  - Here is the link to see what stories people have signed up for already
    - Choose a story at least one week before you present
  - Here is the link to sign up to present
    - Sign up to present by Friday Oct 26
- Happy Birthday Franz Liszt

Questions?

Outline

- ➔ Python
- ➔ Homework #3 Hints

Python

- What is Python?

What is Python?

- A scripting language
- Also an object-oriented language
    - Allows classes and member functions
- Easier to read than C
- Very popular language

Optparse Library

- Library to help with parsing command-line options
- Argument
  - String entered on the command line and passed into the script
  - Arguments are elements of sys.argv[1:]
    - sys.argv[0] is the name of the program being executed
- Option
  - An argument used to supply information to guide or customize the execution of a program
  - Usually, one dash followed by a single letter (-x or -F)
  - OR, two dashes followed by word(s) (--filename or --dry-run)

Optparse Library

- Option Argument
  - An argument that follows an option and is closely associated to that option
  - It is consumed from the argument list when the option is
  - It can be a separate argument, or part of the option:
    - E.g. --file foo.txt
    - OR --file=foo.txt

Python Quirks

- ➔ Whitespace matters!
  - ◆ There are no curly braces or closing keywords (e.g. fi, done, etc.)
  - ◆ The indentation of a line makes a difference
- ➔ Tabs vs. Spaces
  - ◆ Some text editors include tab characters, some use spaces
  - ◆ Sometimes Python can run into errors since those are not equal
  - ◆ Just make sure to be consistent!

Lists

- A Python list is similar to a C++ array
  - Dynamic
    - It expands as needed when new items are added
  - Heterogeneous
    - It can hold objects of different type
    - Ie it can hold both integers and strings
- Accessing elements
  - List_name[*index*]
  - List_name[*start*:*end*]
    - Start is included, but end is not

Lists

- Adding elements to a list
  - List1 = [7, 8, “nine”]
  - List1.append(“ten”)
  - print List1
    - [7, 8, ‘nine’, ‘ten’]
- Merging Lists
  - List2 = [“this”, “that”]
  - List3 = [“these”, “those”]
  - List2 + List3
    - [‘this’, ‘that’, ‘these’, ‘those’]

Lists

➔ Looping over Python list

```
For element in List1:
    #Do stuff with the element
```

Dictionaries

- ➔ Similar to hash tables
- ➔ Stores data as key-value pairs
- ➔ Dict = {}
  - ◆ Creates an empty dictionary called Dict
- ➔ Keys are unique
  - ◆ Values are not necessarily unique
  - ◆ Keys must also be immutable

Dictionaries

➔
```
Dict1 = {}
Dict1[“ten”] = 10
print Dict1[“ten”]
   10
```

➔
```
Meaning = {}
Meaning[42] = [“life”, “universe”]
Meaning[42].append(“everything”)
print Meaning[42]
   [‘life’, ‘universe’, ‘everything’]
```

Dictionaries

- ➔ Testing within a dictionary:

```
if key in dict:
    dict[key].append(val)
else:
    dict[key] = [val]
```

- ➔ Iterating over a dictionary

```
for key in dict:    # only gives you keys
for key,value in dict.iteritems():  # Python 2
for key,value in dict.items():#Python 3
```

For Loops

- Python for loops generally iterate over an object
  - Such as a list
- If you need to iterate over indexes:

```
for i in range(len(list)):
   print i
```

Homework 3

- randline.py
  - You can run it with
    - `./randline.py -n N filename`
  - This takes *N* random lines from *filename*
- Options and Arguments:
  - -n specifies number of lines to write
    - This is an option
  - *N* is the number of lines we want
    - This is an option argument
  - *filename* is the file we are taking lines from
    - This is an argument

`#!/usr/bin/python`	Tells the shell which interpreter to use
`import random, sys`	Import statements, similar to include statements
`from optparse import OptionParser`	Import OptionParser class from optparse module
`class randline:`	The beginning of the class statement: randline
`def __init__(self, filename):`	The constructor
`f = open (filename, 'r')`	Creates a file handle
`self.lines = f.readlines()`	Reads the file into a list of strings called lines
`f.close ()`	Close the file
`def chooseline(self):`	The beginning of a function belonging to randline
`return random.choice(self.lines)`	Randomly select a number between 0 and the size of lines and returns the line corresponding to the randomly selected number
`def main():`	The beginning of main function
`version_msg = "%prog 2.0"`	version message
`usage_msg = """%prog [OPTION]... FILE Output randomly selected lines from FILE."""`	usage message

```
parser = OptionParser(version=version_msg,
                      usage=usage_msg) parser.add_option("-n",
"--numlines",              action="store", dest="numlines",
            default=1, help="output NUMLINES  lines (default
1)")

options, args = parser.parse_args(sys.argv[1:])

try:
    numlines = int(options.numlines)
except:
    parser.error("invalid NUMLINES: {0}".
            format(options.numlines))
if numlines < 0:
    parser.error("negative count: {0}".
format(numlines))
if len(args) != 1:
    parser.error("wrong number of operands")
input_file = args[0]
try:
    generator = randline(input_file)
    for index in range(numlines):
        sys.stdout.write(generator.chooseline())
except IOError as (errno, strerror):
    parser.error("I/O error({0}): {1}". format(errno, strerror))

if __name__ == "__main__":
    main()
```

Creates OptionParser instance

Start defining options, action “store” tells optparse to take next argument and store to the right destination which is “numlines”. Set the default value of “numlines” to 1 and help message.

options: an object containing all option args
args: list of positional args leftover after parsing options
Try block
  get numline from options and convert to integer
Exception handling
  error message if numlines is not integer type, replace {0} w/ input

If numlines is negative
  error message

If length of args is not 1 (no file name or more than one file name)
  error message
Assign the first and only argument to variable input_file
Try block
  instantiate randline object with parameter input_file
  for loop, iterate from 0 to numlines – 1
    print the randomly chosen line
Exception handling
  error message in the format of “I/O error (errno):strerror

In order to make the Python file a standalone program

Homework 3

- You will be creating shuf.py
  - This should function essentially the same way as GNU shuf
  - Including the options:
    - --input-range (-i), --head-count (-n), --repeat (-r), and --help
  - Support any number (including zero) of non-option arguments, as well as the argument "-" meaning standard input
  - You will have to port your shuf.py to Python 3

Homework 3 Hints

- For Q4:
  - Lookup “automatic tuple unpacking”
- If you’re unsure how shuf.py should output something
  - Try it on GNU shuf
- Use randline.py as a starting point
  - There are still plenty more to look up about arguments, options, and option arguments
- If you have troubles with optparse under python 3, you can use argparse instead.