

## 7.5: SEARCHING THROUGH A FILE



Contributed by [Chuck Severance](#)  
Clinical Associate Professor (School of Information) at [University of Michigan](#)

When you are searching through data in a file, it is a very common pattern to read through a file, ignoring most of the lines and only processing lines which meet a particular condition. We can combine the pattern for reading a file with string methods to build simple search mechanisms.

For example, if we wanted to read a file and only print out lines which started with the prefix "From:", we could use the string method *startswith* to select only those lines with the desired prefix:

```
fhand = open('mbox-short.txt')
count = 0
for line in fhand:
    if line.startswith('From:'):
        print(line)

# Code: http://www.py4e.com/code3/search1.py
```

When this program runs, we get the following output:

```
From: stephen.marquard@uct.ac.za

From: louis@media.berkeley.edu

From: zqian@umich.edu

From: rjlowe@iupui.edu
...
```

The output looks great since the only lines we are seeing are those which start with "From:", but why are we seeing the extra blank lines? This is due to that invisible *newline* character. Each of the lines ends with a newline, so the `print` statement prints the string in the variable *line* which includes a newline and then `print` adds *another* newline, resulting in the double spacing effect we see.

We could use line slicing to print all but the last character, but a simpler approach is to use the *rstrip* method which strips whitespace from the right side of a string as follows:

```
fhand = open('mbox-short.txt')
for line in fhand:
    line = line.rstrip()
    if line.startswith('From:'):
        print(line)

# Code: http://www.py4e.com/code3/search2.py
```

When this program runs, we get the following output:

```
From: stephen.marquard@uct.ac.za
From: louis@media.berkeley.edu
From: zqian@umich.edu
From: rjlowe@iupui.edu
From: zqian@umich.edu
From: rjlowe@iupui.edu
From: cwen@iupui.edu
...
```

As your file processing programs get more complicated, you may want to structure your search loops using `continue`. The basic idea of the search loop is that you are looking for "interesting" lines and effectively skipping "uninteresting" lines. And then when we find an interesting line, we do something with that line.

We can structure the loop to follow the pattern of skipping uninteresting lines as follows:

```
fhand = open('mbox-short.txt')
for line in fhand:
    line = line.rstrip()
    # Skip 'uninteresting lines'
    if not line.startswith('From:'):
        continue
    # Process our 'interesting' line
    print(line)

# Code: http://www.py4e.com/code3/search3.py
```

The output of the program is the same. In English, the uninteresting lines are those which do not start with "From:", which we skip using `continue`. For the "interesting" lines (i.e., those that start with "From:") we perform the processing on those lines.

We can use the `find` string method to simulate a text editor search that finds lines where the search string is anywhere in the line. Since `find` looks for an occurrence of a string within another string and either returns the position of the string or -1 if the string was not found, we can write the following loop to show lines which contain the string "@uct.ac.za" (i.e., they come from the University of Cape Town in South Africa):

```
fhand = open('mbox-short.txt')
for line in fhand:
    line = line.rstrip()
    if line.find('@uct.ac.za') == -1: continue
    print(line)

# Code: http://www.py4e.com/code3/search4.py
```

Which produces the following output:

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
X-Authentication-Warning: set sender to stephen.marquard@uct.ac.za using -f
From: stephen.marquard@uct.ac.za
Author: stephen.marquard@uct.ac.za
From david.horwitz@uct.ac.za Fri Jan  4 07:02:32 2008
X-Authentication-Warning: set sender to david.horwitz@uct.ac.za using -f
From: david.horwitz@uct.ac.za
Author: david.horwitz@uct.ac.za
...
```