# Lecture 7 - Counting Words, Using a file and a List

## Feb 8, 2022

## What is a file

A file is a name with a set of data. Quite often this data is a set of lines of text. (Programs are an order set of lines of text)

```
"Gunter, Dolly R",(072) 123-4760
"Polk, Hattie S",(563) 404-0792
```

If we had this inside the program as a list:

```
lines = [
    "\"Gunter, Dolly R\",(072) 123-4760",
    "\"Polk, Hattie S\",(563) 404-0792"
    ]
```

## First let's read in a file

```
 1:
 2: def readFileIntoList(fn):
 3:
 4:     f = open(fn,"r",encoding="utf8")
 5:     if f == None:
 6:         print ( f"Invalid file {fn} - failed to open" )
 7:         return None
 8:     dt = f.readlines()
 9:     f.close()
10:     for i in range (len(dt)):
11:         s = dt[i].rstrip()
12:         dt[i] = s
13:
14:     return dt
15:
16:
17: # Automated Test
18: if __name__ == "__main__":
19:     n_err = 0
20:
21:     got = readFileIntoList("2names.txt")
22:     expect = [
23:         "\"Gunter, Dolly R\",(072) 123-4760",
24:         "\"Polk, Hattie S\",(563) 404-0792"
25:         ]
26:     if got[0] != expect[0]:
27:         n_err = n_err + 1
28:         print ( "Error: Test 1: file read error expected {} got {}".
29:                 format (  expect[0], got[0] ) )
```

```
30:     if got[1] != expect[1]:
31:         n_err = n_err + 1
32:         print ( "Error: Test 2: file read error expected {} got {}".
33:                 format (  expect[1], got[1] ) )
34:     if len(got) != len(expect):
35:         n_err = n_err + 1
36:         print ( "Error: Test 3: file read error expected {} lines got {}".
37:                 format (  len(expect[1]), len(got[1]) ) )
38:
39:     if n_err == 0 :
40:         print ( "PASS" )
41:     else:
42:         print ( "FAILED" )
43:
```

Let's use this:

```
1: import readNameList
2: data = readNameList.readFileIntoList("50000phone.csv")
3: print ( data[0] )
```

Now let's use this to search the list.

```
1: import readNameList
2:
3: phone_list = readNameList.readFileIntoList("50000phone.csv")
4:
5: print ( "Enter a Name to Lookup" )
6: lookFor = input()
7:
8: found = False
9: for v in phone_list:
10:     if lookFor in v:
11:         found = True
12:         print ( "Found {}".format(v) )
13: if not found:
14:     print ( "no names found" )
```

For Loop Pattern:

```
1: found = False
2: for v in phone_list:                # variable in List
3:     if lookFor in v:                # match criteria
4:         found = True                # Ok Found it
5:         print ( "Found {}".format(v) )
6: if not found:                       # Never did find it
7:     print ( "no names found" )
```

Or with a while loop:

```
found = False
i = 0
while ( i < len(phone_list) ):
    if lookFor in phone_list[i]:    # match criteria
        found = True                # Ok Found it
        print ( "Found {}".format(phone_list[i]) )
    i = i + 1
if not found:                       # Never did find it
    print ( "no names found" )
```

Line 4 has `in` - let's take a look at the `in` operator.

```
a = "Fox in Socks"
if "Socks" in a :
    print ( "Good! Found '{}' in '{}'".format("Socks","Fox in Socks") )
else:
    print ( "Bad... did not find '{}' in '{}'".format("Socks","Fox in Socks") )

if "Tweetle" in a:
    print ( "'{}' should not be foudn in '{}'".format("Tweetle","Fox in Socks") )
else:
    print ( "This is good! '{}' is not in '{}'".format("Tweetle","Fox in Socks") )
```

Suppose you want to split a string up into the words that make it up.

Let's start with a string.

```
a = "Fox in Socks by Dr. Suess"
```

We can use "split" to break it up on the blanks. This will give us a list of blank separate words.

```
a = "Fox in Socks by Dr. Suess"
w = a.split()
print ( "words ->{}<- length {}".format(w,len(w)) )
```

We can combine this in a loop and find out the number of words in a list of lists.

```
list_of_list = [
    "Fox Socks",
    "Box Knox",
    "Knox in Box",
    "Fox in Socks",
]
cnt = 0
for s in list_of_list:
    t = s.split()
    cnt = cnt + len(t)
print ( "total {}".format(cnt) )
```

## Read in A File Name:

Our tool for reading a file into a list takes a file name. What happens if we read the file name from the user.

```
1: import readNameList
2:
3: print ( "Enter a File Name")
4: fn = input()
5:
6: a_list = readNameList.readFileIntoList(fn)
7:
8: flen = len(a_list)
9: print ( "# of lines in file {}".format(flen) )
```