Tips for E4

by Deborah R. Fowler

open

close

Back in Class 5 we talked about opening and closing file – you have done this - both in:

- Exercise 2 where you opened a data file
- Exercise 3 where you wrote an data file – let’s review

```
fileVar = open(filename,'r')

OR

fileVar = open(filename,'w')

fileVar.close()
```

We talked about using relative paths as well as they are better!

```
>>>
>>> kermit = open("C:/Users/Deborah/Desktop/testdata.txt",'r')
>>>
```

Shown is an absolute path
Better is to use RELATIVE paths …

```python
kermit = open("testdata.txt",'r')
for line in kermit:
    print line

kermit.close()
```

```
10.6    11.5   40.6

20.0     50.6    50.0

10.0   50.8 45.7
>>>
```

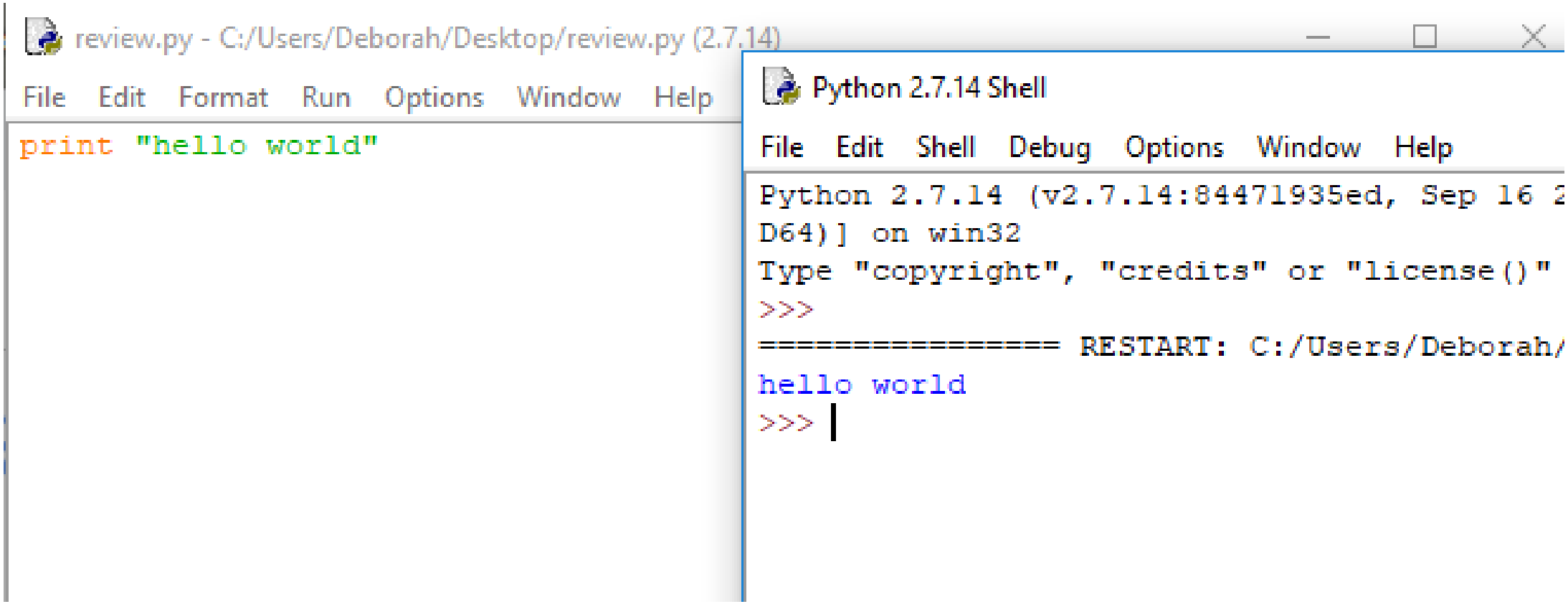
Other reminders

split() breaks up a string into smaller chunks (by default it will base it on whitespace)

Individual elements can be accessed with subscripts

kermit[0] gives the zeroth element of a list called kermit

Let’s go back to class one and start with our hello world



We then moved on to define function so we could organize our code and perhaps call some segment of code multiple times, reducing repetition of code

```
def printhello():
    print "hello world"

printhello()
```

Python 2.7.14 Shell

File Edit Shell Debug Optio

```
Python 2.7.14 (v2.7.14:
D64)] on win32
Type "copyright", "cred
>>>
================ RESTAR
hello world
>>>
```

We then added information sent to the function in the form of parameters (makes the function more useful)

File Edit Format Run Options Window Help

```
def printhello(phrase):
    print phrase

printhello("hello world")
```

Python 2.7.14 Shell

File Edit Shell Debug Options

```
Python 2.7.14 (v2.7.14:8447
D64)] on win32
Type "copyright", "credits"
>>>
================= RESTART: (
hello world
>>>
```

The function can be called as many times as we need it

```
def printhello(phrase):
    print phrase

printhello("hello world")
printhello("hope this helps")
printhello("this is all review")
```

Python 2.7.14 Shell

File Edit Shell Debug Options Winc

```
Python 2.7.14 (v2.7.14:8447193
D64)] on win32
Type "copyright", "credits" or
>>>
================= RESTART: C:\U
hello world
>>>
================= RESTART: C:\U
hello world
hope this helps
this is all review
>>>
```

Loops could be used to repeat

```
def printhello(phrase):
    print phrase

for i in range(0,10):
    printhello("hello world")
    print i+1
```

Python 2.7.14 Shell

File Edit Shell Debug

```
Python 2.7.14 (v2.7
D64)] on win32
Type "copyright", "
>>>
================ RE
hello world
1
hello world
2
hello world
3
hello world
4
hello world
5
hello world
6
hello world
7
hello world
8
hello world
9
hello world
10
>>>
```

Make our code cleaner: Wrap our code in main
Comment with top block, Comment intent

```python
# Review
#
# Author: Deborah R. Fowler
# Date: 10/28/2018
# Description: reviewing concepts learned early in the quarter
# input: none
# output: phrases

# print a phrase to the consule window
def printhello(phrase):
    print phrase

# print multiple times
def main():
    for i in range(0,10):
        printhello("hello world" + " " + str(i))

# call to main (consistent with other programming languages
main()
```

Python 2.7.14 Shell

File Edit Shell Debug Op

```
Python 2.7.14 (v2.7.1
D64)] on win32
Type "copyright", "cre
>>>
================ REST
hello world 0
hello world 1
hello world 2
hello world 3
hello world 4
hello world 5
hello world 6
hello world 7
hello world 8
hello world 9
>>>
```

We could rewrite our for loop as a while loop

```python
# Review
#
# Author: Deborah R. Fowler
# Date: 10/28/2018
# Description: reviewing concepts learned early in the quarter
# input: none
# output: phrases

# print a phrase to the consule window
def printhello(phrase):
    print phrase

# print multiple times
def main():
    i = 0
    while (i < 10):
        printhello("hello world" + " " + str(i))
        i = i + 1

# call to main (consistent with other programming languages
main()
```

Python 2.7.14 Shell

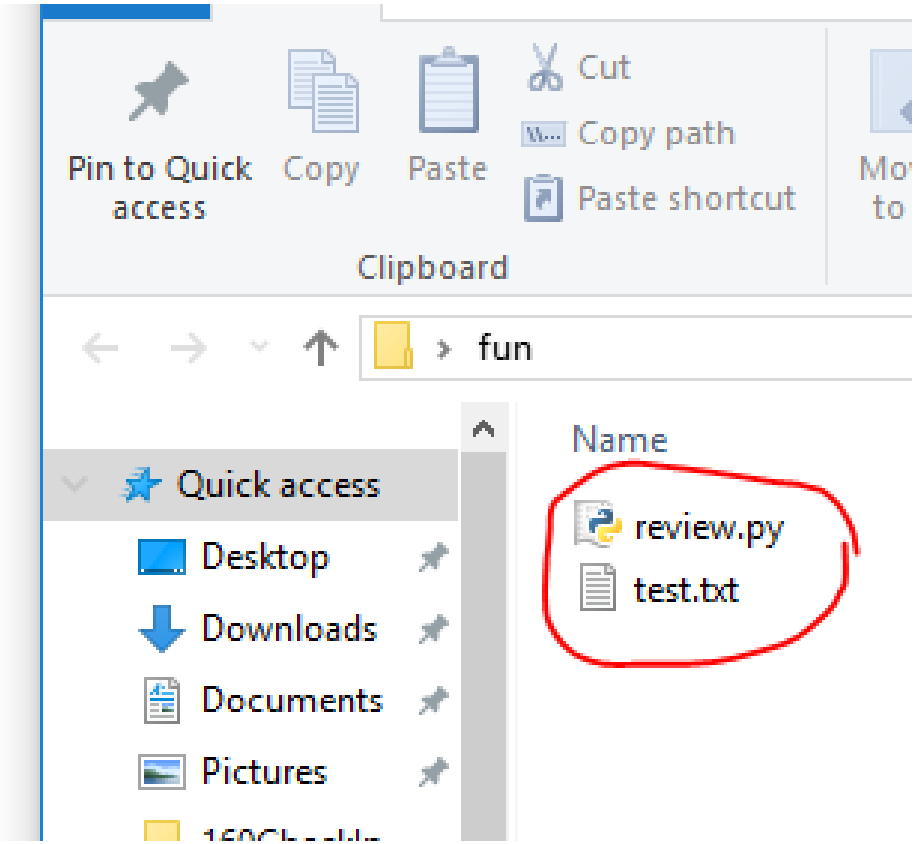
File Edit Shell Deb

```
Python 2.7.14 (v
D64)] on win32
Type "copyright"
>>>
================
hello world 0
hello world 1
hello world 2
hello world 3
hello world 4
hello world 5
hello world 6
hello world 7
hello world 8
hello world 9
>>>
```

Now what if we wanted to put our output into a file?

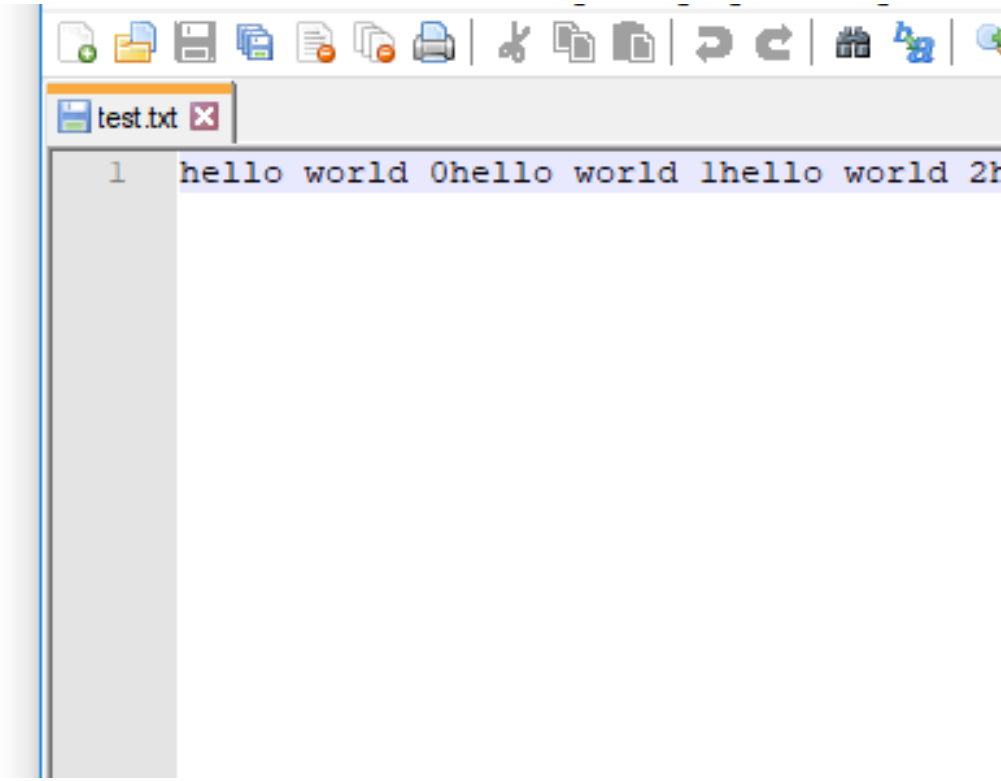
First, we need to open a file

```
def printhello(phrase):
    return phrase

def main():
    i = 0
    filename = "test.txt"
    fileVar = open(filename,'w')
    while (i < 10):
        temp = printhello("hello world" + " " + str(i))
        i = i + 1
    fileVar.close()

main()
```

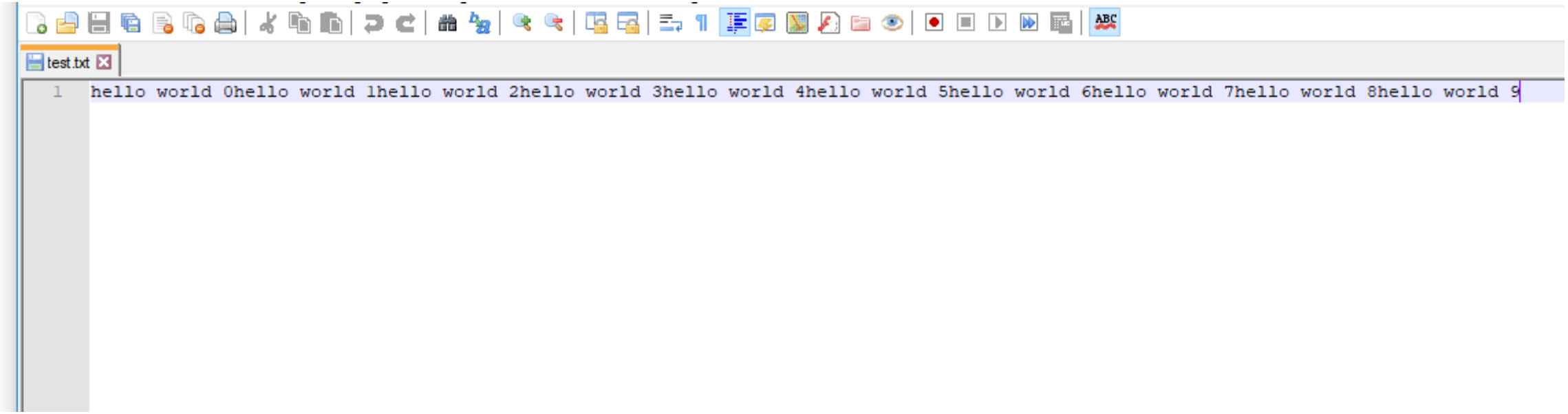


The previous statements opened a file – but the file was empty - we had not written anything into it yet – let's change that

```
def printhello(phrase):
    return phrase

def main():
    i = 0
    filename = "test.txt"
    fileVar = open(filename,'w')
    while (i < 10):
        temp = printhello("hello world" + " " + str(i))
        fileVar.write(temp)
        i = i + 1
    fileVar.close()

main()
```



Note that the write command runs the line all together whereas print adds a newline character “\n”



We can do that as well …

```python
def printhello(phrase):
    return phrase

def main():
    i = 0
    filename = "test.txt"
    fileVar = open(filename,'w')
    while (i < 10):
        temp = printhello("hello world" + " " + str(i) + "\n")
        fileVar.write(temp)
        i = i + 1
    fileVar.close()

main()
```

File Edit Search View Enco

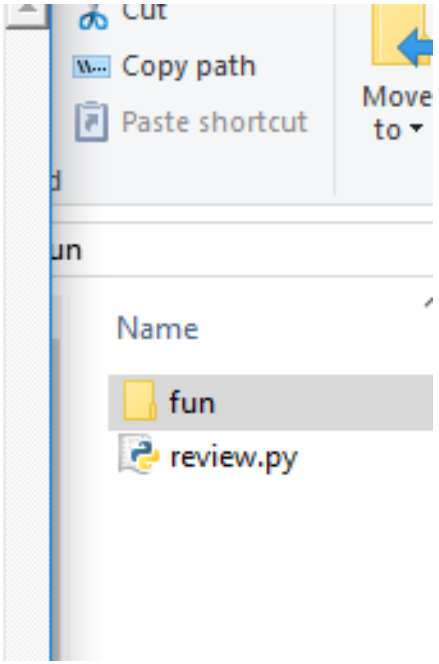
test.txt

```
hello world 0
hello world 1
hello world 2
hello world 3
hello world 4
hello world 5
hello world 6
hello world 7
hello world 8
hello world 9
```

Now consider in your assignment that you are to produce many files – that would quickly clutter up your working directory

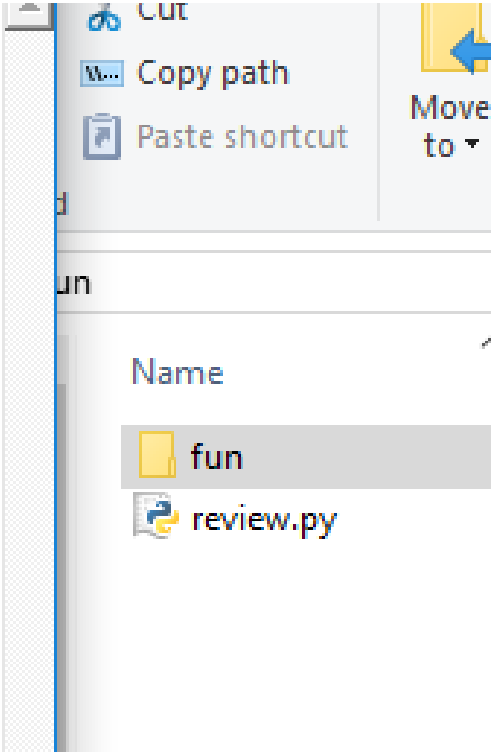
To put these into a directory you would simply create one manually and write to there

```
def printhello(phrase):
    return phrase

def main():
    i = 0
    filename = "fun/test.txt"
    fileVar = open(filename,'w')
    while (i < 10):
        temp = printhello("hello world" + " " + str(i) + "\n")
        fileVar.write(temp)
        i = i + 1
    fileVar.close()

main()
```



Referencing now the file with a path to fun, a directory we created …

```
def printhello(phrase):
    return phrase

def main():
    i = 0
    filename = "fun/test.txt"
    fileVar = open(filename,'w')
    while (i < 10):
        temp = printhello("hello world" + " " + str(i) + "\n")
        fileVar.write(temp)
        i = i + 1
    fileVar.close()

main()
```



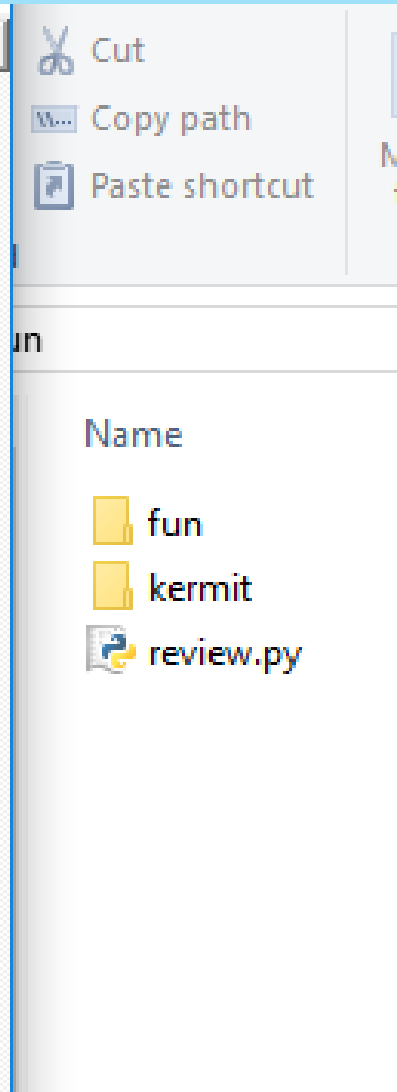
Referencing now the file with a path to fun, a directory we created OR we can create one with a python call to mkdir

```
import os

os.mkdir("myDirNameOrPath")
```

You do not have to do so
Ignore the following blue slides if you are struggling
you do not need to create a directory using code

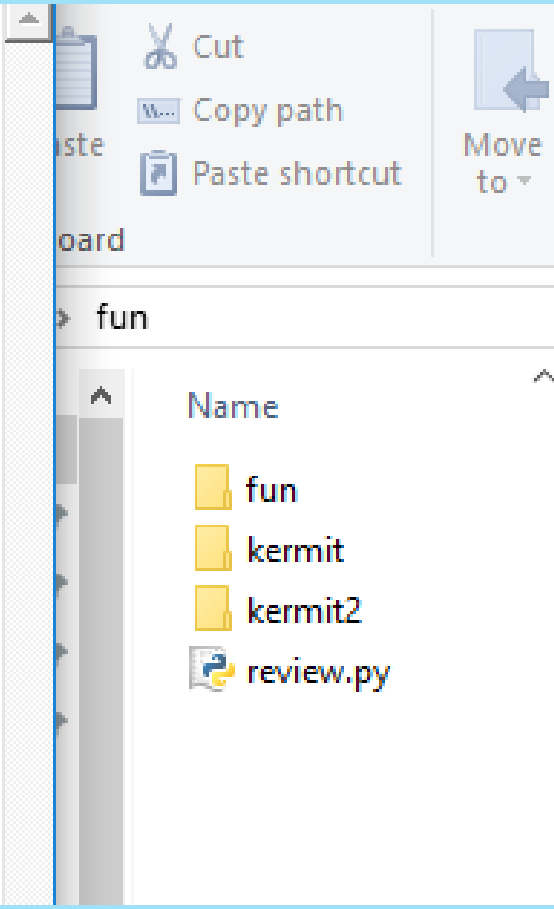
Referencing now the file with a path to fun, a directory we created OR …

```python
# gives you access to built-in operating system dependent modules of Python
import os

def printhello(phrase):
    return phrase

def main():
    i = 0
    os.makedirs("./kermit")
    filename = "kermit/test.txt"
    fileVar = open(filename,'w')
    while (i < 10):
        temp = printhello("hello world" + " " + str(i) + "\n")
        fileVar.write(temp)
        i = i + 1
    fileVar.close()

main()
```



mkdir or makedirs are both fine

```python
# gives you access to built-in operating system dependent modules of Python
import os

def printhello(phrase):
    return phrase

def main():
    i = 0
    os.mkdir("./kermit2")
    filename = "kermit2/test.txt"
    fileVar = open(filename,'w')
    while (i < 10):
        temp = printhello("hello world" + " " + str(i) + "\n")
        fileVar.write(temp)
        i = i + 1
    fileVar.close()

main()
```



If you try to write over it, it will cause an error – you can get fancier by adding more code …

```python
# gives you access to built-in operating system dependent modules
import os

def printhello(phrase):
    return phrase

def main():
    i = 0
    os.mkdir("./kermit2")
    filename = "kermit2/test.txt"
    fileVar = open(filename,'w')
    while (i < 10):
        temp = printhello("hello world" + " " + str(i) + "\n")
        fileVar.write(temp)
        i = i + 1
    fileVar.close()

main()
```

```
Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:25:58) [MSC v.1500 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
============== RESTART: C:\Users\Deborah\Desktop\fun\review.py ==============

Traceback (most recent call last):
  File "C:\Users\Deborah\Desktop\fun\review.py", line 18, in <module>
    main()
  File "C:\Users\Deborah\Desktop\fun\review.py", line 9, in main
    os.mkdir("./kermit2")
WindowsError: [Error 183] Cannot create a file when that file already exists: '.
/kermit2'
>>>
```

If you try to write over it, it will cause an error – you can get fancier by adding more code …
A selection statement (if statement) check if the path exists

```
# gives you access to built-in operating system dependent modules of Python
import os

def printhello(phrase):
    return phrase

def main():
    i = 0
    if os.path.exists("./kermit2"):
        os.mkdir("./kermit3")
    filename = "kermit3/test.txt"
    fileVar = open(filename,'w')
    while (i < 10):
        temp = printhello("hello world" + " " + str(i) + "\n")
        fileVar.write(temp)
        i = i + 1
    fileVar.close()

main()
```

Making these variables makes it more general, robust, easier to extend, for example we could write over the directory if we wanted to (**do not need this for E4**)

```
# gives you access to built-in operating system dependent modules of P
import os
import shutil

def printhello(phrase):
    return phrase

def main():
    i = 0
    dirname = "kermit"
    if os.path.exists(dirname):
        shutil.rmtree(dirname)
    os.mkdir(dirname)
    filename = "kermit/test.txt"
    fileVar = open(filename,'w')
    while (i < 10):
        temp = printhello("something new" + " " + str(i) + "\n")
        fileVar.write(temp)
        i = i + 1
    fileVar.close()

main()
```

shutil offers high-level file manipulation operations

We will discuss this when we cover bash and compare it to python scripting

Going back to Exercise 4

You now know how to create a file and put lines of data into that file

What about creating multiple files?

For the moment, let’s assume we have a directory called kermit and that is where we will create our multiple files

Currently our code created a single file with 10 lines

```python
def printhello(phrase):
    return phrase

def main():
    filename = "kermit/test.txt"
    fileVar = open(filename,'w')
    i = 0
    while (i < 10):
        temp = printhello("something new" + " " + str(i) + "\n")
        fileVar.write(temp)
        i = i + 1
    fileVar.close()

main()
```

test.txt

```
something new 0
something new 1
something new 2
something new 3
something new 4
something new 5
something new 6
something new 7
something new 8
something new 9
```

Code so far (creating a directory manually)

```
def printhello(phrase):
    return phrase

def main():
    filename = "kermit/test.txt"
    fileVar = open(filename,'w')
    for i in range(0,10):
        temp = printhello("something new" + " " + str(i) + "\n")
        fileVar.write(temp)
    fileVar.close()

main()
```

If for loops are more comfortable you can use that as show below

Now how would you make say 25 files given this code?

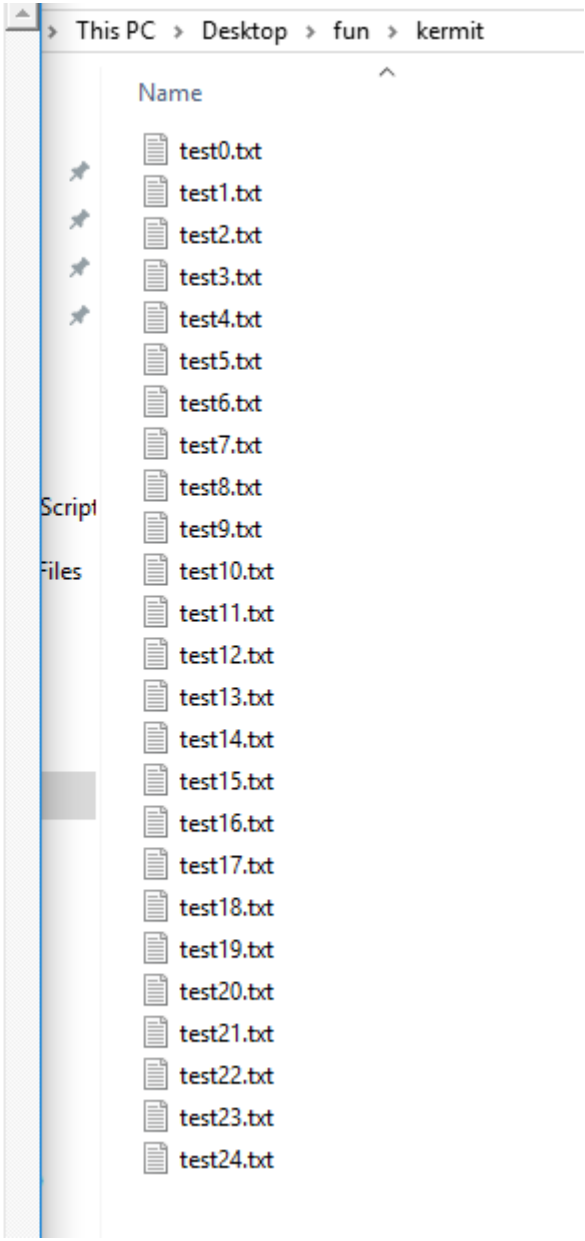
```
def printhello(phrase):
    return phrase

def main():
    filename = "kermit/test.txt"
    fileVar = open(filename,'w')
    for i in range(0,10):
        temp = printhello("something new" + " " + str(i) + "\n")
        fileVar.write(temp)
    fileVar.close()

main()
```

If for loops are more comfortable you can use that as show below

Now how would you make say 25 files given this code?

```
def printhello(phrase):
    return phrase

def singleFile(num):
    filename = "kermit/test" + str(num)+ ".txt"
    fileVar = open(filename,'w')
    for i in range(0,10):
        temp = printhello("something new" + " " + str(i) + "\n")
        fileVar.write(temp)
    fileVar.close()

def main():
    for filenum in range(0,25):
        singleFile(filenum)

main()
```

```python
def printhello(phrase):
    return phrase

def singleFile(num):
    filename = "kermit/test" + str(num)+ ".txt"
    fileVar = open(filename,'w')
    for i in range(0,10):
        temp = printhello("something new" + " " + str(i) + "\n")
        fileVar.write(temp)
    fileVar.close()

def main():
    for filenum in range(0,25):
        singleFile(filenum)

main()
```



Result of the code above (and on the previous slide) is that it produces 25 files (0 to 24) and each one of those files contains data – in this case 10 lines of what we have written

Now we did not have to split this up into functions – functions allow us to think of one task at a time

The following code does the same thing:

```
for filenum in range(0,25):
    filename = "kermit/test" + str(filenum)+ ".txt"
    fileVar = open(filename,'w')
    for i in range(0,10):
        temp = str(i) + " " + "all in one" + "\n"
        fileVar.write(temp)
    fileVar.close()
```

Now consider where you are getting your data from and re-write the previous slide into an algorithm, substituting the information you will need for Exercise 4 where appropriate