Fil

We found more than one version of Functions in a Nutshell. You can view the alternate here. ✕

A text file is basically referred to as a sequence of lines. **In order to make use of this function, the file which is being read must be in the same path or folder as your python code.** If it is in a different path you must make reference to that path by using a path, ironically.

A path looks like this

```
path = C:\Users\salim\.PyCharmCE2018.2\config\scratches"
infile = open(path)
```

Open Function

We use the function "open" to access a file in python. We attach that to a variable name. E.G

```
variable_Name = open("Example.txt", "r")
```

Example_txt is our file name which we put in quotations. The 'r' that follows is to specify that this is the file that's being read.

What can we do with a file?

Well, with a file + python we can read the contents of that file line by line by using a for loop. We can also write on an empty file as well(aka appending values).

Check Reading file.py for examples

.write()

Suppose you'd like to append some fancy words to an empty file. The .write method has you covered. By using the variable name attached to our file and .write() we can easily append a new value for that.

```
outfile = open("Example.txt", "w")
##Note we use "w" instead of "r" because we are writing on this file, not reading.
outfile.write("Steal your heart")
```

This will append "Steal your heart" to the first line. You can also do the same with variables as well. If s = "Steal your heart" then, outfile.write(s) will give the same output as the code above.

.close

.close method is used when you're done reading from and/or writing from the files you accessed. It should come at the end of your code, theoretically, of course.

```
variable_name.close()
outfile.close()
```

check Reading Files.py for more examples