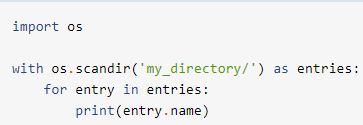
**OS/Pathlib Info**

**os.listdir()** ……..lists every file in the current directory. Iterable i.e.

* Entry = os.listdir()
* For entry in entries:
  + Print(entry)……will print each file one at a time

**Can also use os.scandir(‘path here’)** to do the same as above

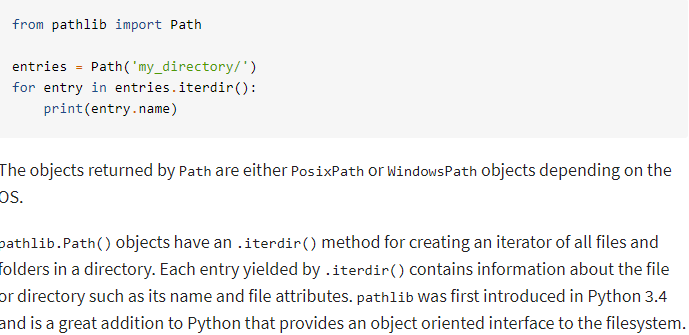
* os.scandir() returns an iterator rather than a list, so to see each file, you’d have to iterate over it via **for** loop



Use of context manager above means that once it has been used, Python will close it, and it will no longer take up any space in memory. Not necessary, though.

**List an Entire Directory Using Pathlib**

It is similar to the context manager we used above with **os.scandir()**.

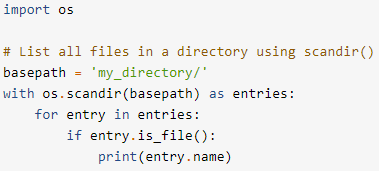


**List all Files in a Directory:**

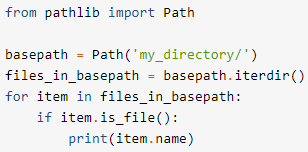


The call to os.listdir(basepath) or os.scandir(basepath) returns all items in the path, and then the call to os.path.isfile() filters them by whether or not they are a file (i.e. it won’t print out folders/directories)

**Example with os.scandir():**

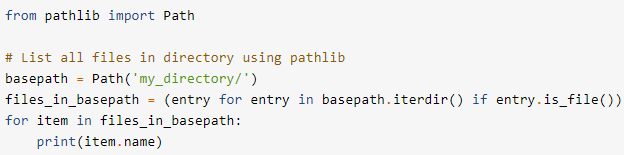


**Example with pathlib.Path():**



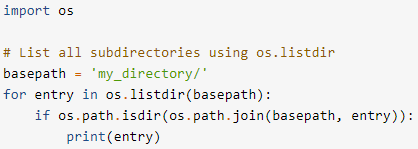
**NOTE:** you can just add “.iterdir()” to the end of the Path call in the basepath variable, rather than creating a new variable **files\_in\_basepath**

**Example with pathlib.Path() using a Generator Expression:**

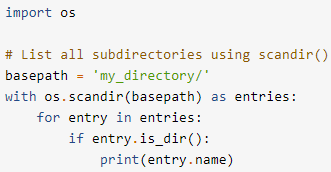


**Listing Subdirectories Instead of Files:**

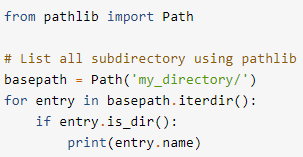
**Example w/os.listdir() and os.path.isdir():**



**Example w/os.scandir():**



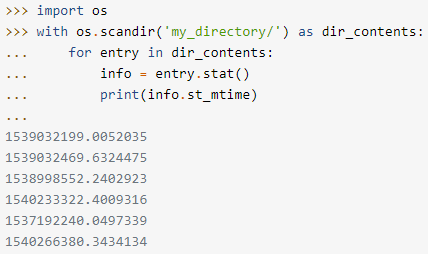
**Example w/pathlib.Path():**



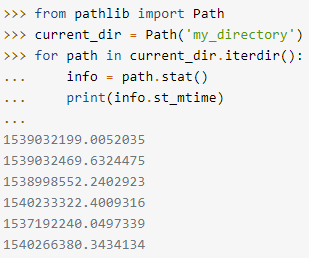
**NOTE:** you could just add “.iterdir()” to the end of the **basepath** variable

**Getting File Attributes:**

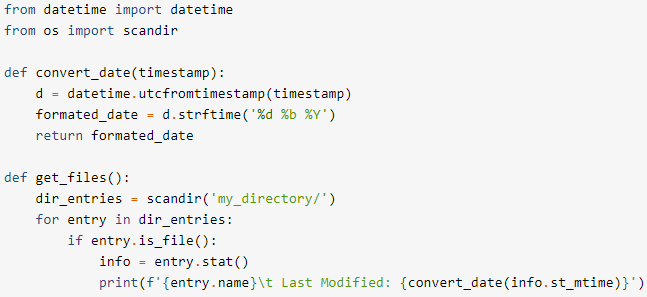
**Example w/ os.scandir():**



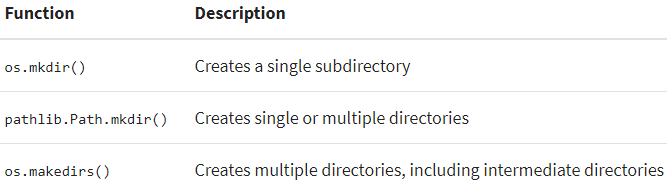
**Example w/pathlib.Path():**



**Example of a helper function used to convert the Unix timestamp from os.scandir():**



**How to create directories:**



**Choices:**

**1).** Pass a path to **os.mkdir()**

**2).** Use pathlib.Path.mkdir()

x = Path(‘my\_directory’)

x.mkdir()

When using mkdir(), it will throw a **FileExistsError** if the file already exists, so it may be a good idea, unless you are certain that it doesn’t exist, to handle the potential error in a try/except block:

Try:

p.mkdir()

Except FileExistsError:

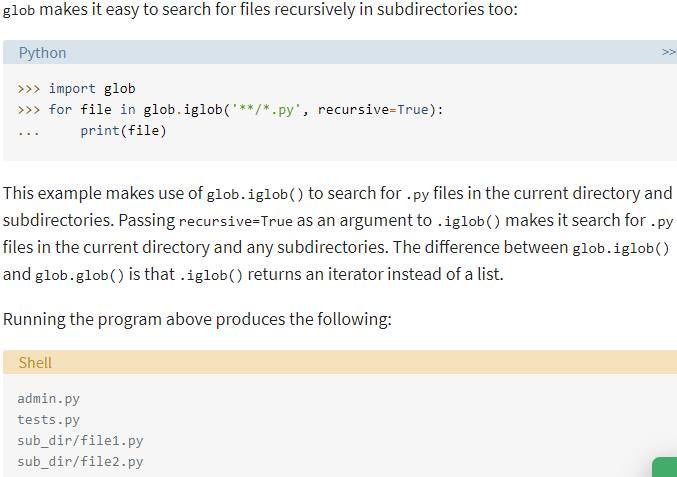
Pass

**OR**

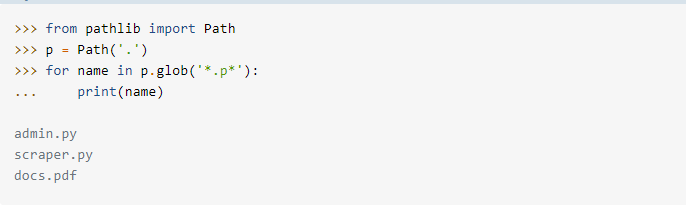
Pass in the “exist\_ok=True” argument into .mkdir()

**Using glob.glob() to do File Pattern Matching**

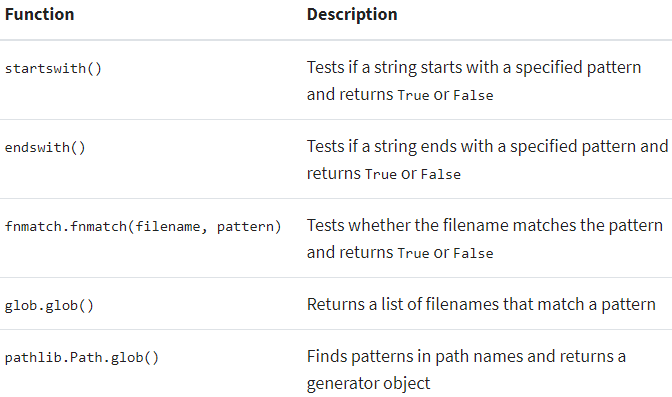




**Example of File Pattern Matching Using Pathlib**

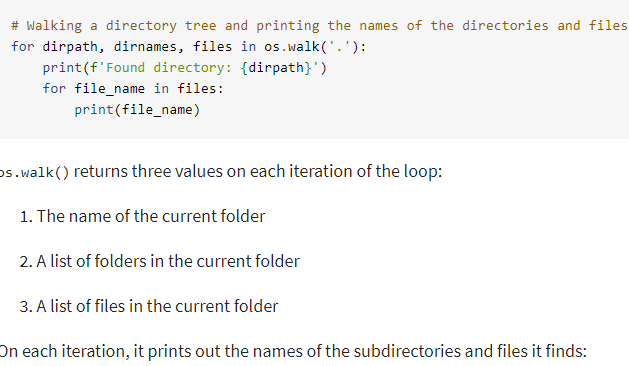


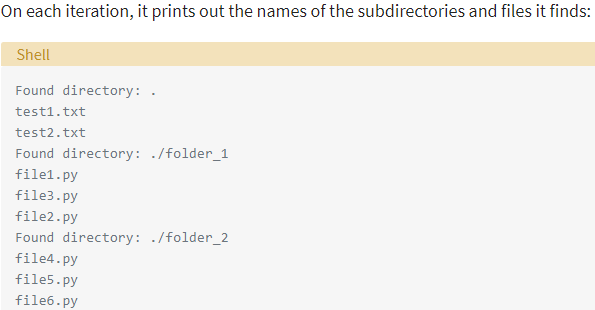
**Recap of File Pattern Matching Methods**



**Traversing File Directories and Processing Files with os.walk():**

**os.walk()** is used to generate a file name in a directory tree by walking the directory either top-down or bottom-up





**Use os.walk() to traverse the file path in a bottom-up manner:**

