15-110 Principles of Computing – S19

Lecture 16:
Files I/O 2

Teacher:
Gianni A. Di Caro

Files so far

- Open a file:

```
f = open(file_name, mode)    # mode: r, w, a, +, t, b
```

- Read at most nbytes:

```
data_string = f.read(nbytes)
```

- Read one line record:

```
data_string = f.readline()
```

- Read all remaining line records from current position:

```
data_string = f.readlines()
```

- Get current position in file:

```
pos = f.tell()
```

- Go to position pos in file:

```
f.seek(pos)
```

- Iterate through records:

```
for line in f:
    # do something with line
```

- Write a string of data:

```
f.write(data_string)
```

- Close a file:

```
f.close()
```

Output / string formatting

```
f = open('personal_data.txt', 'w+')

name = 'John'
title = 'Mr.'
age_y = 30
height = 500 / 2.8
weight = 86 * 0.94
record = name + ' ' + title + ' ' + str(age) + ' ' + str(height) + ' ' + str(weight)
f.write(record + '\n')

name = 'Anne-Marie'
title = 'Ms.'
age_y = 26
height = 500 / 3
weight = 59 * 0.94
record = name + ' ' + title + ' ' + str(age) + ' ' + str(height) + ' ' + str(weight)
f.write(record + '\n')

f.close()
```

John Mr. 30 178.57142857142858 80.83999999999999
Anne-Marie Ms. 30 166.66666666666666 55.459999999999994

hard to read, misaligned fields, unnecessary digits…

Output / string formatting: `str.format()`

```
f = open('personal_data.txt', 'w+')

name = 'John'
title = 'Mr.'
age_y = 30
height = 500 / 2.8
weight = 86 * 0.94
record = '{:12s} {:4s} {:3d}  {:8.3f}  {:8.3f}'.format(name, title, age_y, height, weight)
f.write(record + '\n')

name = 'Anne-Marie'
title = 'Ms.'
age_y = 26
height = 500 / 3
weight = 59 * 0.94
record = '{:12s} {:4s} {:3d}  {:8.3f}  {:8.3f}'.format(name, title, age_y, height, weight)
f.write(record + '\n')

f.close()
```

```
John         Mr.  30  178.571  80.840
Anne-Marie   Ms.  26  166.667  55.460
```

Output / string formatting: `str.format()`

```
f = open('personal_data.txt', 'w+')

name = 'John'
title = 'Mr.'
age_y = 30
height = 500 / 2.8
weight = 86 * 0.94
record ='|{:12s}| |{:4s}| |{:3d}| |{:8.3f}| |{:8.3f}|'.format(name, title,age_y,height,weight)
f.write(record + '\n')

name = 'Anne-Marie'
title = 'Ms.'
age_y = 26
height = 500 / 3
weight = 59 * 0.94
record ='|{:12s}| |{:4s}| |{:3d}| |{:8.3f}| |{:8.3f}|'.format(name,title,age_y, height,weight)
f.write(record + '\n')

f.close()
```

```
|John        | |Mr. | | 30| | 178.571| |  80.840|
|Anne-Marie  | |Ms. | | 26| | 166.667| |  55.460|
```

String fields are left-aligned
Numeric fields are right-aligned

`str.format()`: positional substitution, formatting specifiers

```
'{:12s} {:4s} {:3d}  {:8.3f}  {:8.3f}'.format(name, title, age_y, height, weight)
```

the string being constructed with formatting specifications fields

the variables/values to be substituted in the formatting specifications fields

left-aligned string field of 4 chars

right-aligned float field of 8 chars, 3 after .

John	Mr.	30	178.571	80.840
Anne-Marie	Ms.	26	166.667	55.460

left-aligned string field of 12 chars

right-aligned int field of 3 chars

right-aligned float field of 8 chars, 3 after .

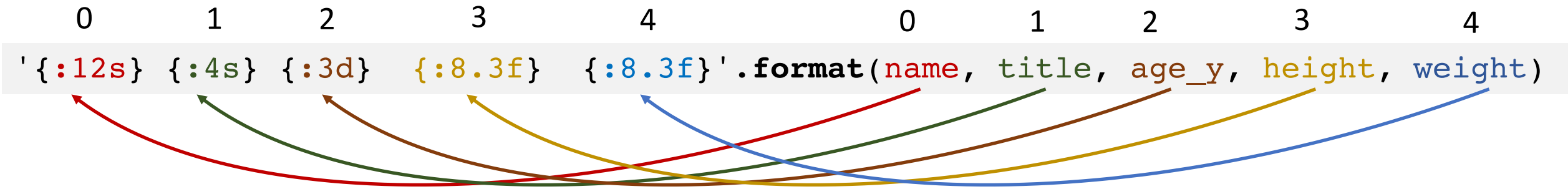
Formatting specifiers:

- `s`: for strings
- `d`: for decimal integers
- `f`: for floats
- b: for Booleans
- e/E: exponential notation
- g: general notation, rounds up to 6 significant digits

`str.format()`: positional substitution, formatting specifiers

- **Positional assignment** of variables to formatting specifications (in this case)

Argument index:



- Type of variable and formatting specifier must match!

```
name = 'John'
title = 'Mr.'
age_y = 30
height = 500 / 2.8
weight = 86 * 0.94
record = '{:12s} {:4s} {:3d}  {:3d}  {:3d}'.format(name, title, age_y, height, weight)
```

Throws an error since `height` and `weight` are float types while the specifier is `d` (integer)

`str.format()`: positional substitution, without formatting

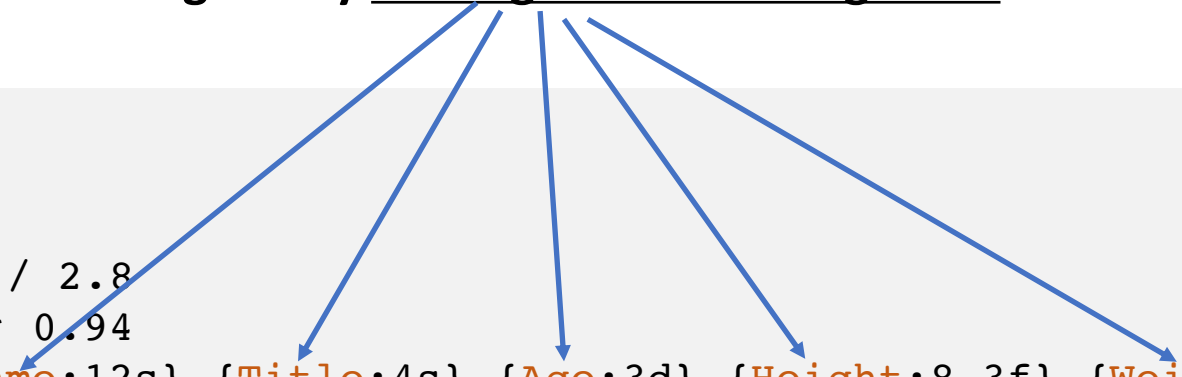
- **Formatting specifiers are <u>not required</u>**

```
name = 'John'
title = 'Mr.'
age_y = 30
height = 500 / 2.8
weight = 86 * 0.94
record = '{} {} {} {} {}'.format(name, title, age_y, height, weight)
print(record)
print(name, title, age_y, height, weight)
```

```
John Mr. 30 178.57142857142858 80.83999999999999
John Mr. 30 178.57142857142858 80.83999999999999
```

str.format(): substitution by name

- **Values can be assigned by <u>naming the formatting fields</u>**



```
name = 'John'
title = 'Mr.'
age_y = 30
height = 500 / 2.8
weight = 86 * 0.94
record = '{Name:12s} {Title:4s} {Age:3d} {Height:8.3f} {Weight:8.3f}'.format(Name=name,
                                    Title=title, Age=age_y, Height= height, Weight=weight)
print(record)
```

```
John         Mr.  30 178.571   80.840
```

<u>General form:</u>

```
'{Field_name_1:FormattingSpecifier1} {Field_name_2:FormattingSpecifier2}'.format(
                        Field_name_1 = variable/value, Field_name_2 = variable/value)
```

`str.format()`: mixing up positional and named substitution

- **Positional fields must be placed <u>before named ones</u>**

```
name = 'John'
title = 'Mr.'
age_y = 30
height = 500 / 2.8
weight = 86 * 0.94
record = '{} {} {:3d} {Height:g} {Weight:e}'.format(name, title, age_y,
                                                    Height= height, Weight=weight)
print(record)
```

```
John Mr.  30 178.571 8.084e+01
```

`str.format()`: string formatting options

- **Align left**, with padding spaces

```
name = 'John'
record = '|{:10s}|'.format(name)
print(record)
```

```
name = 'John'
record = '|{:<10s}|'.format(name)
print(record)
```

```
|John      |
```

- **Align right**, with padding spaces

```
name = 'John'
record = '|{:>10s}|'.format(name)
print(record)
```

```
|      John|
```

`str.format()`: string formatting options

- **Align left,** with selected padding character

```
name = 'John'
record = '|{:_<10s}|'.format(name)
print(record)
```

|John_____|

- **Align right,** with selected padding character

```
name = 'John'
record = '|{:+>10s}|'.format(name)
print(record)
```

|++++++John|

`str.format()`: string formatting options

- **Center** the text in the field

```
name = 'John'
record = '|{:^10s}|'.format(name)
print(record)
```

```
|   John   |
```

- **Truncate** the string up to a given number of characters

```
name = 'John'
record = '|{:.2s}|'.format(name)
print(record)
```

```
|Jo|
```

Many more options for string formatting …

- The `str.format()` method offers a wide range of possibilities (much wider than presented here!)

  https://www.programiz.com/python-programming/methods/string/format

- Similar effects can be achieved using also other methods **(% strings**, and **f-strings**)

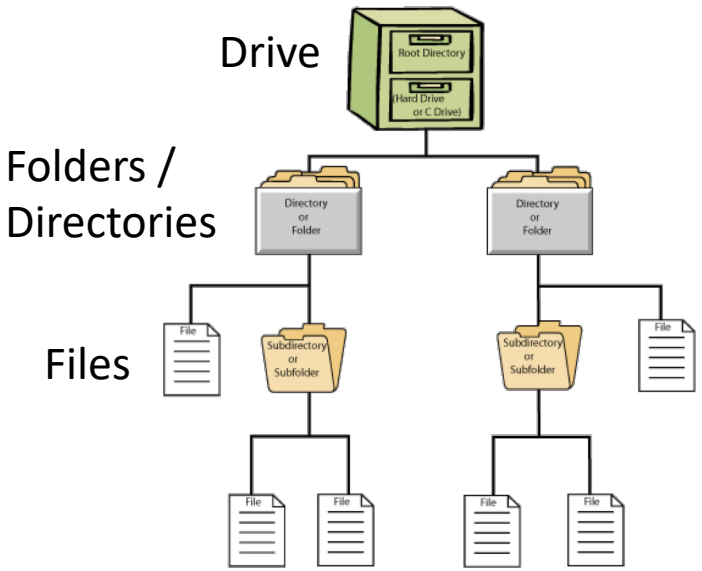
  https://realpython.com/python-string-formatting/

Common questions / challenges about files

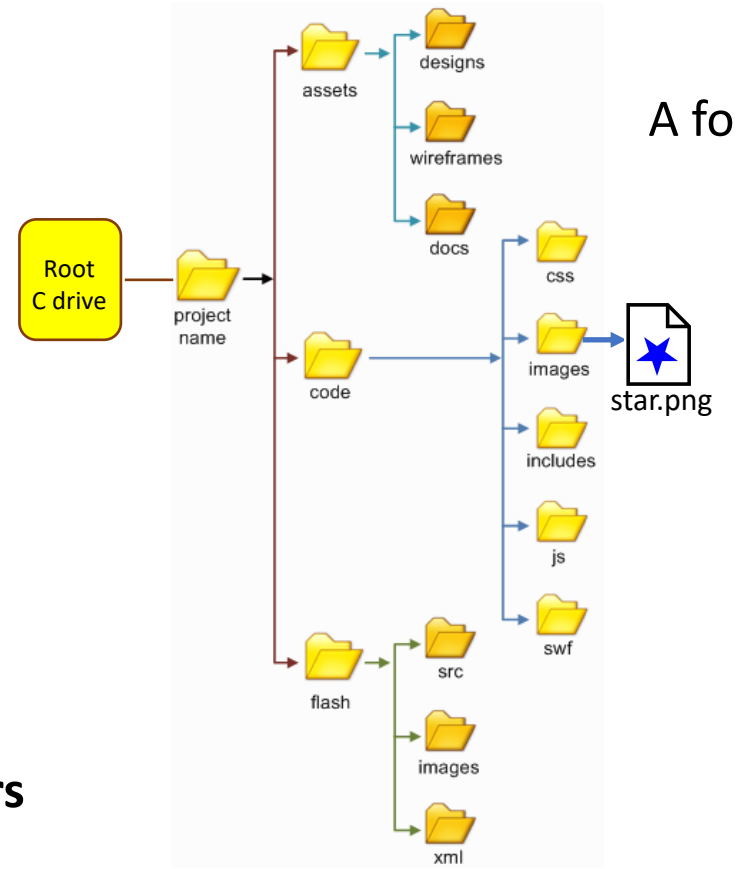
- Trying to open (with a read flag) a non-existing file results into an error: how do we **check if a file exists** or not in the file system prior any attempt to open it? (to avoid that the program crashes?)
- A file might be **located anywhere in the file system**: how do we refer to such a file?
- How do we **search for a named file in the file system**?
- Sometimes it is appropriate to create a new folder where to put newly created files: how do we **create a new folder** from a python program?
- Opening an existing file with the write flag erases file content: how can we **check file status first**?
- A file might be too large to read, or might be not accessible for read or write to my program, or might have been modified very recently: how do we get **information about a file**?

Operations on the file system

- Module `os` (**Operating System**) offers a complete set of functionalities to inspect and manipulate the elements of the **file system of the computer**



Tree (data) structure rooted at the drive: a **hierarchy of folders**



A folder can contain zero or more files and/or zero or more sub-folders

A folder / file is uniquely identified by:

- ✓ a **name**
- ✓ the **path** relative to the root of the drive

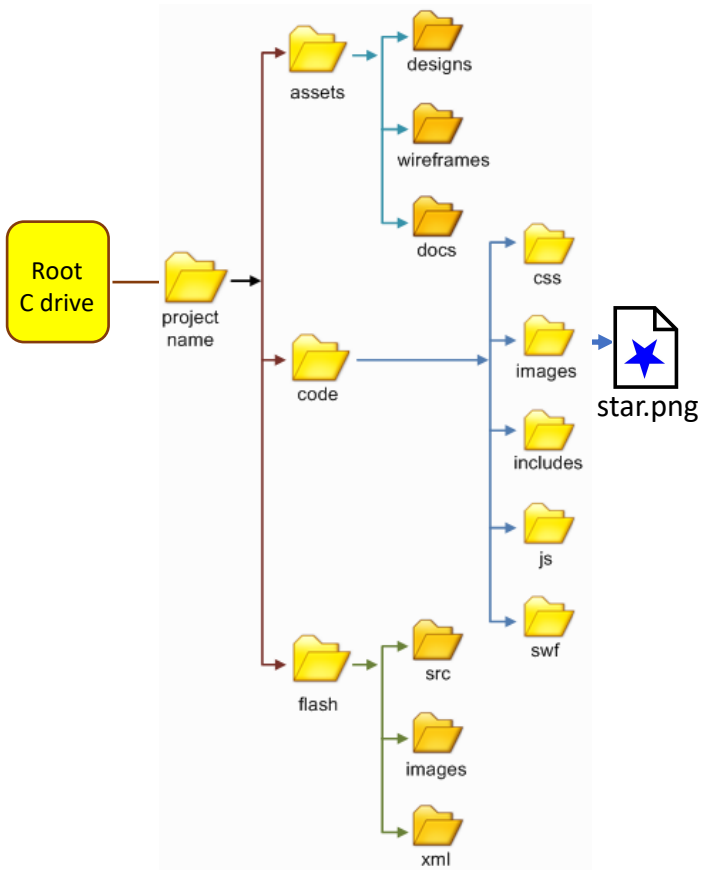
Windows:
C:\project name\code\images\star.png

Unix:
/project name/code/images/star.png

path name

Operations on the file system: check if a file exists

- `os.path.isfile(file_name)`: return `True` if file `file_name` exists, `False` otherwise;
  if `file_name` is not specified through a path, the existence of the file is referred to the current folder

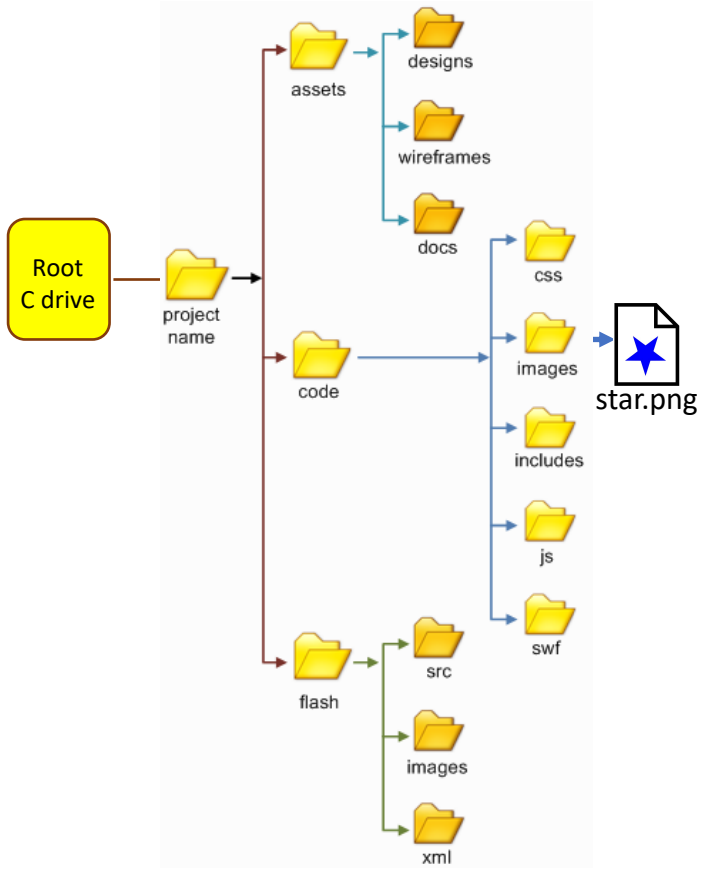



```
import os

if os.path.isfile('star.png'):
    print('File star.png is in the current folder')
elif os.path.isfile('/project name/assets/star.png'):
    print('File star.png is in folder assets')
else:
    print('File star.png is not in the file system')
```

Operations on the file system: check if a folder exists

- `os.path.isdir(folder_name)`: return `True` if folder `folder_name` exists, `False` otherwise; `folder_name` needs to include the full path, this holds also for the current path

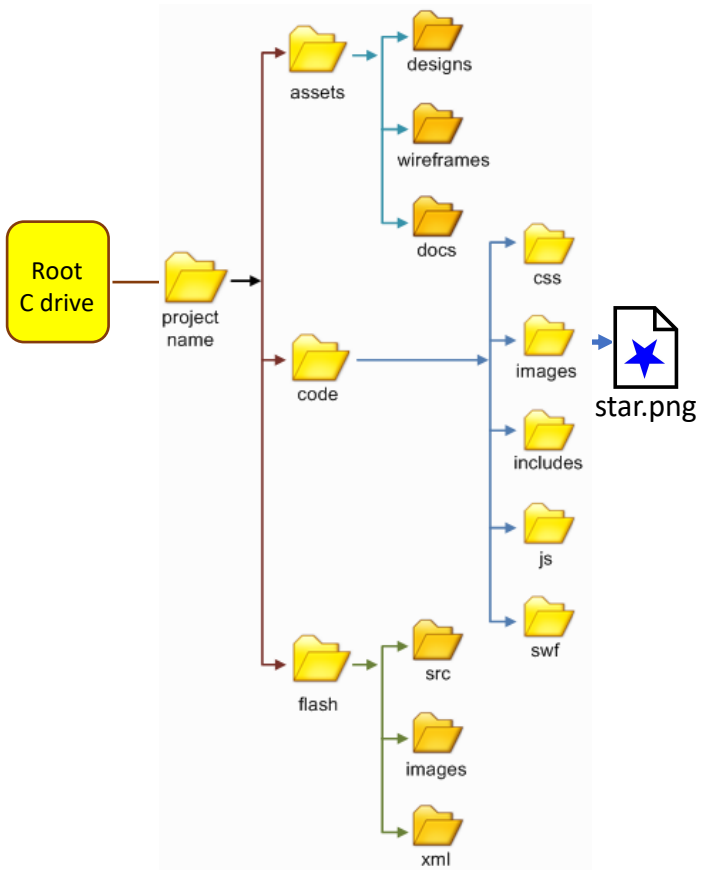



```
import os

if os.path.isdir('/project name/code/images'):
    print('Folder exists in the file system')
else:
    print('Folder is not in the file system')
```

Operations on the file system: check if a named file or folder exists

- `os.path.exists(name)`: return `True` if `name` exists, and it can be either a file or a folder, `False` otherwise; no distinction is made between the two types for checking

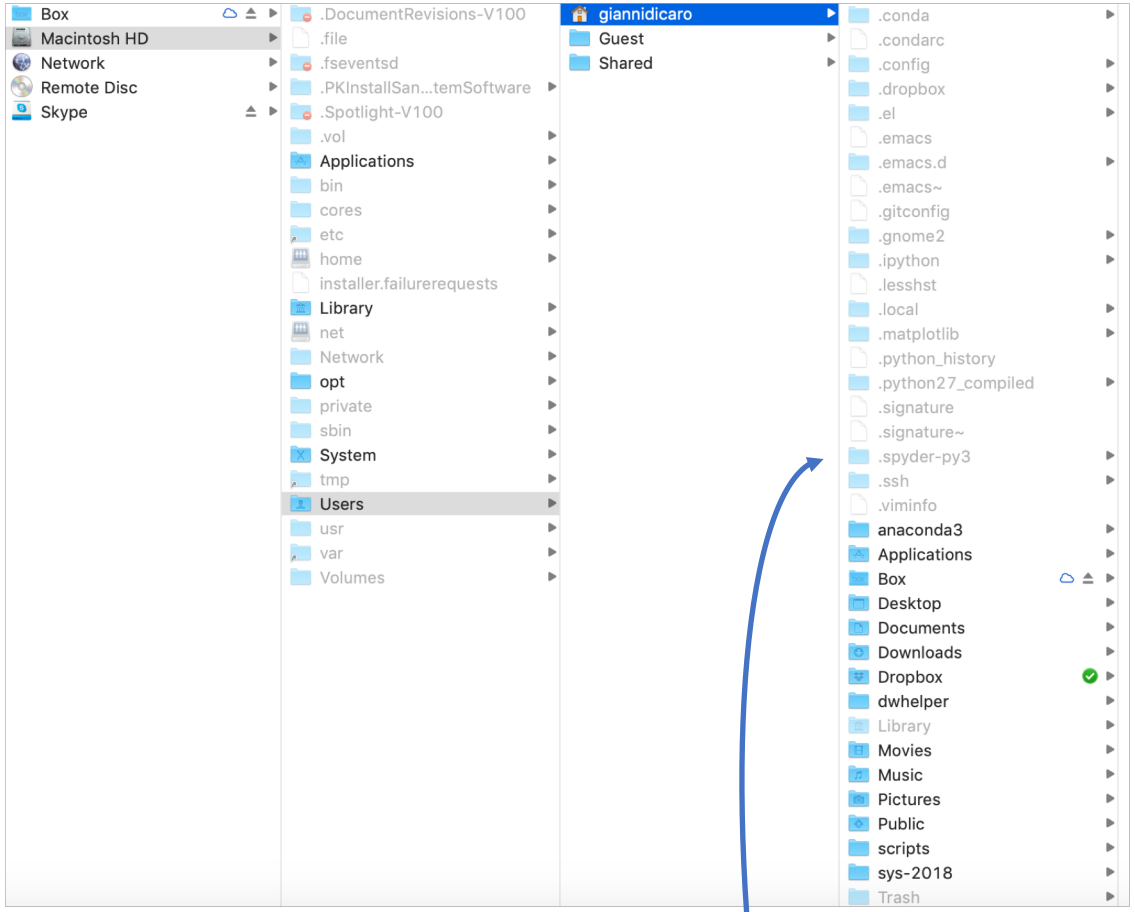



```
import os

if os.path.exists('/project name/code/images'):
    print('Folder exists in the file system')
else:
    print('Folder is not in the file system')
```

Operations on the file system: get the current path / folder

- `os.getcwd()`: Returns a string with the current folder name (**c**urrent **w**orking **d**irectory), including the prefix path



/Users/giannidicaro/.spyder-py3

```
import os

full_path = os.getcwd()
```

/Users/giannidicaro/.spyder-py3

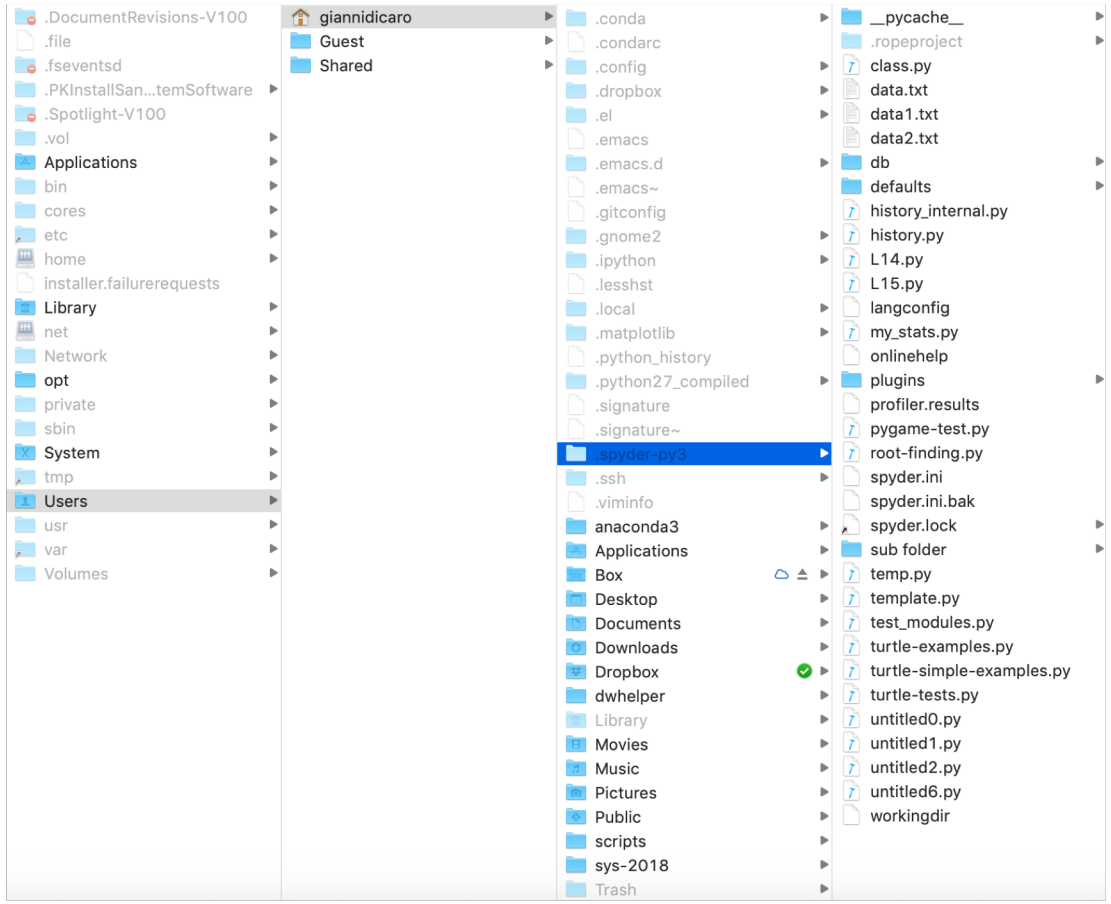
Name of current working folder?

```
folder_start = full_path.rfind('/')
folder_name = full_path[folder_start+1:]
print(folder_name)
```

.spyder-py3

Operations on the file system: get the list of files in current folder

- `os.listdir(path=None)`: Returns a list containing the names (as strings) of the files in the directory `path`. If `path` is `None`, the current directory, `'./'`, is listed




```
import os

file_list = os.listdir()
print(file_list)
```

```
['untitled0.py', 'untitled1.py', 'turtle-examples.py', 'plugins', 'workingdir',
'onlinehelp', '__pycache__', 'defaults', 'pygame-test.py', 'spyder.ini', 'turtle-
tests.py', '.ropeproject', 'temp.py', 'class.py', 'profiler.results', 'template.py', 'db',
'my_stats.py', 'turtle-simple-examples.py', 'data1.txt', 'untitled2.py', 'spyder.lock',
'langconfig', 'L15.py', 'data2.txt', 'history_internal.py', 'untitled6.py',
'test_modules.py', 'spyder.ini.bak', 'L14.py', 'data.txt', 'sub folder', 'root-
finding.py', 'history.py']
```

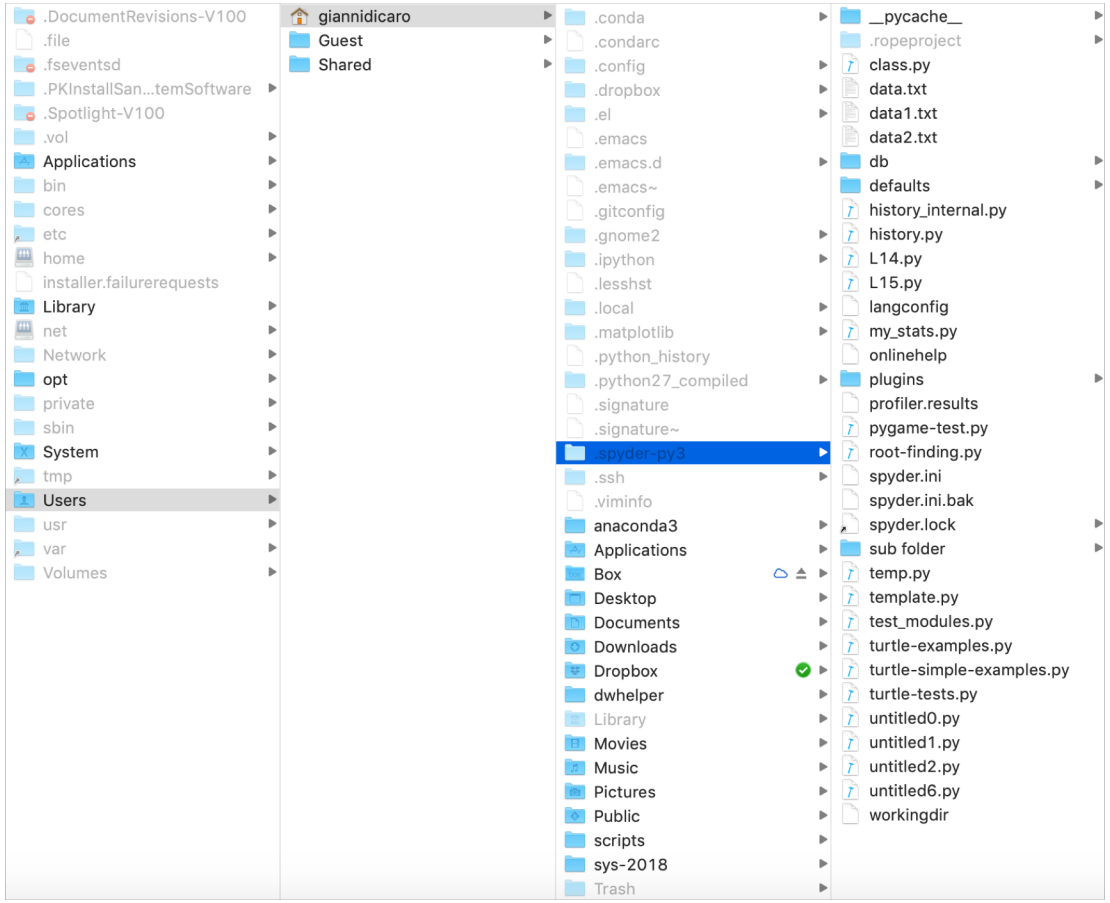
Is a specific file in the folder?

```
if 'L15.py' in file_list:
    print('File L15.py is in current folder')
```

```
File L15.py is in the current folder
```

Operations on the file system: change the working folder

- `os.chdir(path)`: Changes the current path to `path`




```
import os
print(os.getcwd())
os.chdir('/Applications')   #move anywhere in the fs
print(os.getcwd())
os.chdir('/Users/giannidicaro')  #go back to the home folder
print(os.getcwd())
#print(os.listdir())
os.chdir('./anaconda3') #move down to a specific sub-folder
print(os.getcwd())
os.chdir('../')    #move to a folder up in the hierarchy
print(os.getcwd())
os.chdir('./.spyder-py3')
print(os.getcwd())
```

```
/Users/giannidicaro/.spyder-py3
/Applications
/Users/giannidicaro
/Users/giannidicaro/anaconda3
/Users/giannidicaro
/Users/giannidicaro/.spyder-py3
```

Operations on the file system: get status / stats of a file

- `os.stat(file_name)`: Returns a data structure with various information about `file_name`, information is stored in separate fields of the structure, that can be accessed individually with the dot notation

- **st_mode** – protection bits.
- **st_ino** – inode number.
- **st_dev** – device.
- **st_nlink** – number of hard links.
- **st_uid** – user id of owner.
- **st_gid** – group id of owner.
- **st_size** – size of file, in bytes.
- **st_atime** – time of most recent access.
- **st_mtime** – time of most recent content modification.
- **st_ctime** – time of most recent metadata change.

```
import os
stat_info = os.stat('L15.py')
print(stat_info)
print('Size:', stat_info.st_size)
print('Last modification time:', stat_info.st_mtime)
```

```
~/.spyder-py3$ ls -al L15.py
-rw-r--r--@ 1 giannidicaro  staff  3007 Mar  6 14:22 L15.py
~/.spyder-py3$
```

os.stat_result(st_mode=33188, st_ino=5499079, st_dev=16777220, st_nlink=1, st_uid=501, st_gid=20, st_size=2810, st_atime=1551870605, st_mtime=1551870605, st_ctime=1551870605)
Size: 2810
Last modification time: 1551872218.209256

```
from datetime import datetime
print('Last modification date: ', end='')
print(datetime.fromtimestamp(stat_info.st_mtime).strftime('%Y-%m-%d %H:%M:%S'))
```

Last modification date: 2019-03-06 14:36:58

Operations on the file system: create an `ls` command

```
~/.spyder-py3$ ls -al
total 2584
drwxr-xr-x  39 giannidicaro  staff     1248 Mar  7 16:52 .
drwxr-xr-x+ 62 giannidicaro  staff     1984 Mar  6 13:09 ..
drwxr-xr-x   3 giannidicaro  staff       96 Feb 10 09:25 .ropeproject
-rw-------@  1 giannidicaro  staff     8431 Feb 27 00:01 L14.py
-rw-r--r--@  1 giannidicaro  staff     5395 Mar  7 16:52 L15.py
drwxr-xr-x   4 giannidicaro  staff      128 Feb 17 12:15 __pycache__
-rw-r--r--@  1 giannidicaro  staff      286 Feb 19 14:23 class.py
-rw-------@  1 giannidicaro  staff    78706 Mar  6 13:52 cmu_logo.png
-rw-------   1 giannidicaro  staff     2766 Mar  7 16:52 data.txt
-rw-r--r--   1 giannidicaro  staff        0 Mar  3 23:02 data1.txt
-rw-r--r--   1 giannidicaro  staff        0 Mar  3 23:02 data2.txt
drwxr-xr-x   3 giannidicaro  staff       96 Dec 18 21:00 db
drwxr-xr-x   3 giannidicaro  staff       96 Dec 18 21:00 defaults
-rw-r--r--   1 giannidicaro  staff     5105 Mar  7 12:20 history.py
-rw-r--r--   1 giannidicaro  staff       67 Feb 28 00:33 history_internal.py
-rw-r--r--   1 giannidicaro  staff        2 Feb 10 09:25 langconfig
-rw-r--r--@  1 giannidicaro  staff      694 Feb 17 12:02 my_stats.py
-rw-r--r--   1 giannidicaro  staff        0 Feb 27 23:49 onlinehelp
-rw-r--r--   1 giannidicaro  staff       84 Mar  7 16:52 personal_data.txt
drwxr-xr-x   2 giannidicaro  staff       64 Dec 18 21:00 plugins
-rw-r--r--   1 giannidicaro  staff  1064163 Feb  4 17:44 profiler.results
-rw-r--r--@  1 giannidicaro  staff      493 Mar  3 13:40 pygame-test.py
-rw-r--r--@  1 giannidicaro  staff     6838 Feb 17 11:54 root-finding.py
-rw-r--r--   1 giannidicaro  staff    27994 Mar  3 23:01 spyder.ini
-rw-r--r--   1 giannidicaro  staff    27993 Feb 28 04:47 spyder.ini.bak
lrwxr-xr-x   1 giannidicaro  staff        5 Feb 28 00:33 spyder.lock -> 18341
drwx------   2 giannidicaro  staff       64 Mar  6 13:06 sub folder
drwxr-xr-x   3 giannidicaro  staff       96 Mar  7 16:52 temp
-rw-r--r--   1 giannidicaro  staff     4461 Feb 16 16:35 temp.py
-rw-r--r--   1 giannidicaro  staff       98 Feb 10 09:25 template.py
-rw-r--r--@  1 giannidicaro  staff     1725 Feb 19 14:58 test_modules.py
-rw-r--r--@  1 giannidicaro  staff     2287 Jan 25 18:29 turtle-examples.py
-rw-r--r--@  1 giannidicaro  staff      456 Jan 25 19:09 turtle-simple-examples.py
-rw-r--r--@  1 giannidicaro  staff     2471 Jan 25 18:40 turtle-tests.py
-rw-r--r--@  1 giannidicaro  staff     6943 Feb 26 23:31 untitled0.py
-rw-r--r--@  1 giannidicaro  staff     3992 Feb  9 22:48 untitled1.py
-rw-r--r--@  1 giannidicaro  staff      144 Feb 13 22:50 untitled2.py
-rw-r--r--@  1 giannidicaro  staff     3753 Feb 22 12:54 untitled6.py
-rw-r--r--   1 giannidicaro  staff       88 Mar  7 16:52 workingdir
```

```
Directory /Users/giannidicaro/.spyder-py3:
Total 37 files
D  temp                            96   2019-03-07 17:22:10
f  untitled0.py                  6943   2019-02-26 23:31:09
f  untitled1.py                  3992   2019-02-09 22:48:47
f  turtle-examples.py            2287   2019-01-25 18:29:13
D  plugins                         64   2018-12-18 21:00:21
f  workingdir                      88   2019-03-07 17:19:21
f  onlinehelp                       0   2019-02-27 23:49:54
D  __pycache__                    128   2019-02-17 12:15:17
f  personal_data.txt               84   2019-03-07 17:22:09
D  defaults                        96   2018-12-18 21:00:21
f  pygame-test.py                 493   2019-03-03 13:40:57
f  spyder.ini                   27994   2019-03-03 23:01:47
f  turtle-tests.py               2471   2019-01-25 18:40:17
D  .ropeproject                    96   2019-02-10 09:25:46
f  temp.py                       4461   2019-02-16 16:35:20
f  class.py                       286   2019-02-19 14:23:04
f  profiler.results           1064163   2019-02-04 17:44:20
f  template.py                     98   2019-02-10 09:25:45
D  db                              96   2018-12-18 21:00:34
f  my_stats.py                    694   2019-02-17 12:02:52
f  turtle-simple-examples.py      456   2019-01-25 19:09:46
f  data1.txt                        0   2019-03-03 23:02:17
f  untitled2.py                   144   2019-02-13 22:50:01
f  cmu_logo.png                 78706   2019-03-06 13:52:12
f  langconfig                       2   2019-02-10 09:25:41
f  L15.py                        6202   2019-03-07 17:22:09
f  data2.txt                        0   2019-03-03 23:02:29
f  history_internal.py             67   2019-02-28 00:33:15
f  untitled6.py                  3753   2019-02-22 12:54:00
f  test_modules.py               1725   2019-02-19 14:58:07
f  spyder.ini.bak               27993   2019-02-28 04:47:06
f  L14.py                        8431   2019-02-27 00:01:33
f  data.txt                      3021   2019-03-07 17:22:09
D  sub folder                      64   2019-03-06 13:06:04
f  root-finding.py               6838   2019-02-17 11:54:44
f  history.py                    5105   2019-03-07 12:20:37
```

Operations on the file system: create an `ls` command

```
Directory /Users/giannidicaro/.spyder-py3:
Total 37 files
D  temp                            96   2019-03-07 17:22:10
f  untitled0.py                  6943   2019-02-26 23:31:09
f  untitled1.py                  3992   2019-02-09 22:48:47
f  turtle-examples.py            2287   2019-01-25 18:29:13
D  plugins                         64   2018-12-18 21:00:21
f  workingdir                      88   2019-03-07 17:19:21
f  onlinehelp                       0   2019-02-27 23:49:54
D  __pycache__                    128   2019-02-17 12:15:17
f  personal_data.txt               84   2019-03-07 17:22:09
D  defaults                        96   2018-12-18 21:00:21
f  pygame-test.py                 493   2019-03-03 13:40:57
f  spyder.ini                   27994   2019-03-03 23:01:47
f  turtle-tests.py               2471   2019-01-25 18:40:17
D  .ropeproject                    96   2019-02-10 09:25:46
f  temp.py                       4461   2019-02-16 16:35:20
f  class.py                       286   2019-02-19 14:23:04
f  profiler.results           1064163   2019-02-04 17:44:20
f  template.py                     98   2019-02-10 09:25:45
D  db                              96   2018-12-18 21:00:34
f  my_stats.py                    694   2019-02-17 12:02:52
f  turtle-simple-examples.py      456   2019-01-25 19:09:46
f  data1.txt                        0   2019-03-03 23:02:17
f  untitled2.py                   144   2019-02-13 22:50:01
f  cmu_logo.png                 78706   2019-03-06 13:52:12
f  langconfig                       2   2019-02-10 09:25:41
f  L15.py                        6202   2019-03-07 17:22:09
f  data2.txt                        0   2019-03-03 23:02:29
f  history_internal.py             67   2019-02-28 00:33:15
f  untitled6.py                  3753   2019-02-22 12:54:00
f  test_modules.py               1725   2019-02-19 14:58:07
f  spyder.ini.bak               27993   2019-02-28 04:47:06
f  L14.py                        8431   2019-02-27 00:01:33
f  data.txt                      3021   2019-03-07 17:22:09
D  sub folder                      64   2019-03-06 13:06:04
f  root-finding.py               6838   2019-02-17 11:54:44
f  history.py                    5105   2019-03-07 12:20:37
```

```
def ls(path):
    file_list = []
    if path == '.':
        folder = os.getcwd()
    else:
        folder = path
    header = 'Directory ' + folder + ':\n'
    header += 'Total ' + str(len(os.listdir(path))) + ' files'
    file_list.append(header)
    for ff in os.listdir(path):
        if os.path.islink(ff):
            continue
        info = os.stat(ff)
        size = info.st_size
        updated = datetime.fromtimestamp(info.st_mtime).strftime('%Y-%m-%d %H:%M:%S')
        if os.path.isdir(ff):
            file_type = 'D'
        else:
            file_type = 'f'
        file_info = '{type:2s} {name:25s} {size:9d}   {date:18s}'.format(type=file_type,
                    name=ff, size=size, date=updated)
        file_list.append(file_info)
    return file_list
```

Operations on the file system: create an `ls` command

```
for ff in ls('.'):
    print(ff)
```

```
Directory /Users/giannidicaro/.spyder-py3:
Total 37 files
D  temp                          96   2019-03-07 17:22:10
f  untitled0.py                6943   2019-02-26 23:31:09
f  untitled1.py                3992   2019-02-09 22:48:47
f  turtle-examples.py          2287   2019-01-25 18:29:13
D  plugins                       64   2018-12-18 21:00:21
f  workingdir                    88   2019-03-07 17:19:21
f  onlinehelp                     0   2019-02-27 23:49:54
D  __pycache__                  128   2019-02-17 12:15:17
f  personal_data.txt             84   2019-03-07 17:22:09
D  defaults                      96   2018-12-18 21:00:21
f  pygame-test.py               493   2019-03-03 13:40:57
f  spyder.ini                 27994   2019-03-03 23:01:47
f  turtle-tests.py             2471   2019-01-25 18:40:17
D  .ropeproject                  96   2019-02-10 09:25:46
f  temp.py                     4461   2019-02-16 16:35:20
f  class.py                     286   2019-02-19 14:23:04
f  profiler.results         1064163   2019-02-04 17:44:20
f  template.py                   98   2019-02-10 09:25:45
D  db                            96   2018-12-18 21:00:34
f  my_stats.py                  694   2019-02-17 12:02:52
f  turtle-simple-examples.py    456   2019-01-25 19:09:46
f  data1.txt                      0   2019-03-03 23:02:17
f  untitled2.py                 144   2019-02-13 22:50:01
f  cmu_logo.png               78706   2019-03-06 13:52:12
f  langconfig                     2   2019-02-10 09:25:41
f  L15.py                      6202   2019-03-07 17:22:09
f  data2.txt                      0   2019-03-03 23:02:29
f  history_internal.py           67   2019-02-28 00:33:15
f  untitled6.py                3753   2019-02-22 12:54:00
f  test_modules.py             1725   2019-02-19 14:58:07
f  spyder.ini.bak             27993   2019-02-28 04:47:06
f  L14.py                      8431   2019-02-27 00:01:33
f  data.txt                    3021   2019-03-07 17:22:09
D  sub folder                    64   2019-03-06 13:06:04
f  root-finding.py             6838   2019-02-17 11:54:44
f  history.py                  5105   2019-03-07 12:20:37
```

```
def ls(path):
    file_list = []
    if path == '.':
        folder = os.getcwd()
    else:
        folder = path
    header = 'Directory ' + folder + ':\n'
    header += 'Total ' + str(len(os.listdir(path))) + ' files'
    file_list.append(header)
    for ff in os.listdir(path):
        if os.path.islink(ff):
            continue
        info = os.stat(ff)
        size = info.st_size
        updated = datetime.fromtimestamp(info.st_mtime).strftime('%Y-%m-%d %H:%M:%S')
        if os.path.isdir(ff):
            file_type = 'D'
        else:
            file_type = 'f'
        file_info ='{type:2s} {name:25s} {size:9d} {date:18s}'.format(type=file_type,
                    name=ff, size=size, date=updated)
        file_list.append(file_info)
    return file_list
```

Operations on the file system: `ls` with sorting option

```
for ff in ls('.', True):
    print(ff)
```

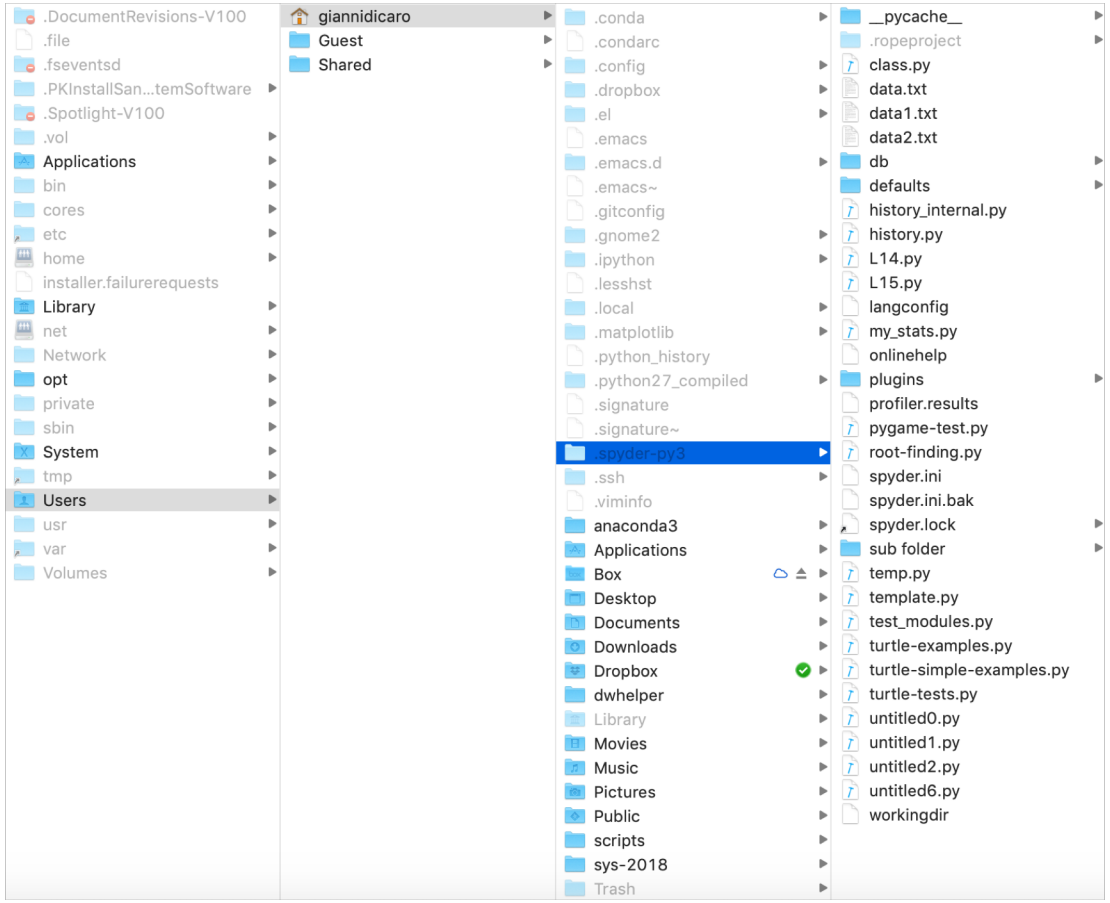
```
Directory /Users/giannidicaro/.spyder-py3:
Total 37 files
D  __pycache__                     128   2019-02-17 12:15:17
D  db                               96   2018-12-18 21:00:34
D  defaults                         96   2018-12-18 21:00:21
D  plugins                          64   2018-12-18 21:00:21
D  sub folder                       64   2019-03-06 13:06:04
D  temp                             96   2019-03-07 17:54:15
f  L14.py                         8431   2019-02-27 00:01:33
f  L15.py                         6276   2019-03-07 17:54:15
f  class.py                        286   2019-02-19 14:23:04
f  cmu_logo.png                  78706   2019-03-06 13:52:12
f  data.txt                       3106   2019-03-07 17:54:15
f  data1.txt                         0   2019-03-03 23:02:17
f  data2.txt                         0   2019-03-03 23:02:29
f  history.py                     5105   2019-03-07 12:20:37
f  history_internal.py              67   2019-02-28 00:33:15
f  langconfig                        2   2019-02-10 09:25:41
f  my_stats.py                     694   2019-02-17 12:02:52
f  onlinehelp                        0   2019-02-27 23:49:54
f  personal_data.txt                84   2019-03-07 17:54:15
f  profiler.results            1064163   2019-02-04 17:44:20
f  pygame-test.py                  493   2019-03-03 13:40:57
f  root-finding.py                6838   2019-02-17 11:54:44
f  spyder.ini                    27994   2019-03-03 23:01:47
f  spyder.ini.bak                27993   2019-02-28 04:47:06
f  temp.py                        4461   2019-02-16 16:35:20
f  template.py                      98   2019-02-10 09:25:45
f  test_modules.py                1725   2019-02-19 14:58:07
f  turtle-examples.py             2287   2019-01-25 18:29:13
f  turtle-simple-examples.py       456   2019-01-25 19:09:46
f  turtle-tests.py                2471   2019-01-25 18:40:17
f  untitled0.py                   6943   2019-02-26 23:31:09
f  untitled1.py                   3992   2019-02-09 22:48:47
f  untitled2.py                    144   2019-02-13 22:50:01
f  untitled6.py                   3753   2019-02-22 12:54:00
f  workingdir                       88   2019-03-07 17:54:09
```

```
def ls(path, sort):
    file_list = []
    if path == '.':
        folder = os.getcwd()
    else:
        folder = path
    header = 'Directory ' + folder + ':\n'
    header += 'Total ' + str(len(os.listdir(path))) + ' files'
    for ff in os.listdir(path):
        if os.path.islink(ff):
            continue
        info = os.stat(ff)
        size = info.st_size
        updated = datetime.fromtimestamp(info.st_mtime).strftime('%Y-%m-%d %H:%M:%S')
        if os.path.isdir(ff):
            file_type = 'D'
        else:
            file_type = 'f'
        file_info = '{type:2s} {name:25s} {size:9d}  {date:18s}'.format(type=file_type,
                    name=ff, size=size, date=updated)
        file_list.append(file_info)
    if sort == True:
        file_list.sort()

    file_list[0] = header
    return file_list
```

Operations on the file system: create a new folder

- `os.mkdir(path)`: Create a new folder at the given `path`, if `path` only contains a name (and not a full path) the folder is created as a sub-folder of the current one




```
import os
os.mkdir('temp')
```

How do we check that the new folder is there?

```
for ff in os.listdir():
    if os.path.isdir(ff):
        print(ff)
```

```
temp
plugins
__pycache__
defaults
.ropeproject
db
sub folder
```

Operations on the file system: remove a folder

- `os.remove(path)`: Remove the folder at the given `path`, the folder must be empty, otherwise a `PermissionError` exception is generated

  - If the folder is empty (and exists) this works with no errors

```
import os
if os.path.isdir('temp'):
    os.rmdir('temp')
```

  - Check first whether the folder is empty or not, remove al folder files

```
import os
if os.path.isdir('temp'):
    os.chdir('temp')
    for file in os.listdir():
        os.remove(file)
    os.chdir('../')
    os.rmdir('temp')
```

Problems with file operations that can result into errors

- Trying to open (with a read flag) a non-existing file results into an error (we can check this first with os methods)
- Trying to perform operations in a file for which we don't have the right permissions result into an error (again, we can avoid this by using the os methods)
- Trying to perform a read / write operation on an already closed file results into an error (no way to overcome this with os methods)
- How do we deal "flexibly" with these situations that could generate errors?
- A more general question:

Can we **try out** operations that could generate an error

without having the program being aborted whenever the error is actually generated?

Dealing with errors: `try-except-else-finally` construct

- When an **error** occurs during the program, Python generates an **exception:** it generates an error type that identifies the exception and then **stops** the execution
- Exceptions can be handled using the **try statement** to avoid that the program does actually stop when an error occurs during the execution

- `try-except-else-finally` blocks:
  - ✓ The **try** block let executing a block of code that can potentially generate an exception
  - ✓ The **except** block let handling the error, if generated by the try block (i.e., what to do when an error occurs)
  - Optional
    - ✓ The **else** block let specifying a block of code that is executed if the try block *didn't generate any exception*
    - ✓ The **finally** block let executing the code, regardless of the result of the try- and except blocks.

```
y = 1
try:
    x /= 10
    y += x
except:
    print("x doesn't exist")
else:
    print('x:', x)
    del x
finally:
    print('y:', y)
```

Catching multiple exceptions

- ✓ **Multiple, different exceptions can be caught**

```
y = 1
x = 10
d = 0.0
try:
    x /= d
    y += x
except NameError:
    print(”Variable doesn't exist")
except ZeroDivisionError:
    print("Division by zero!")
```

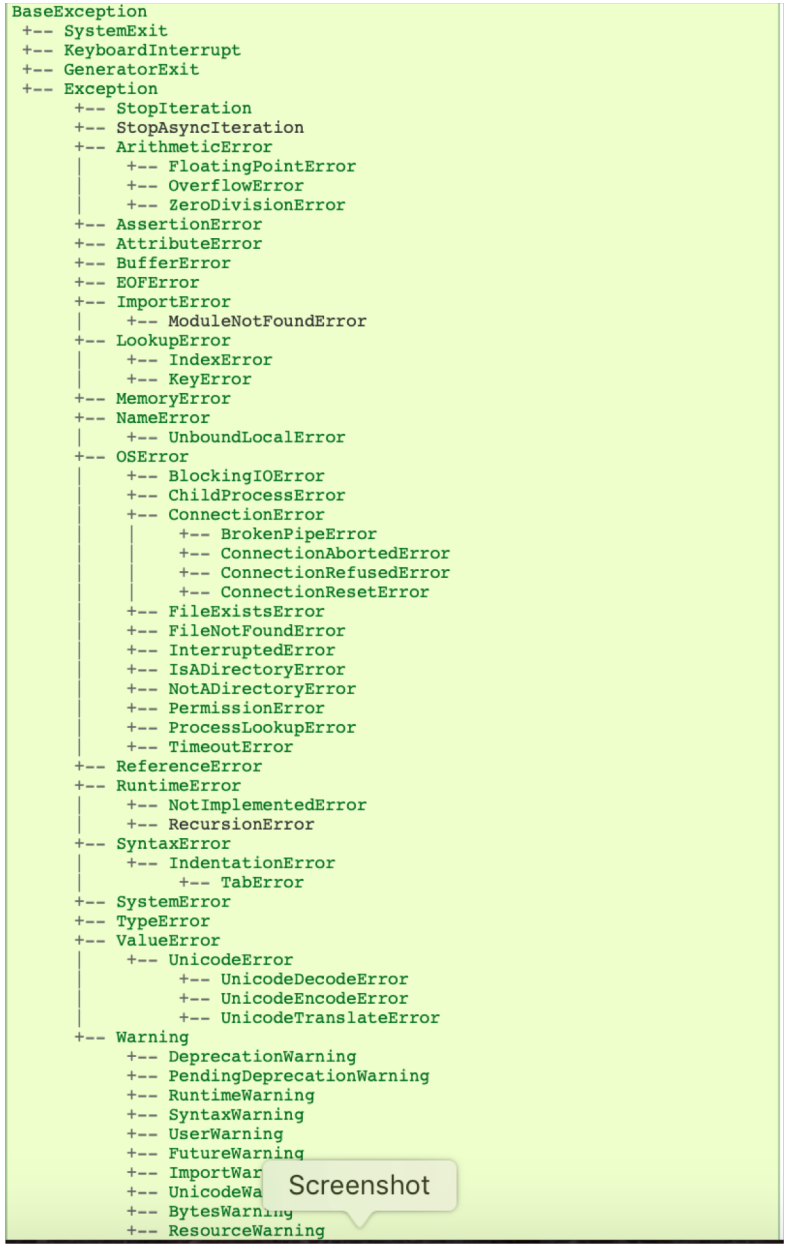
```
  File "/Users/giannidicaro/.spyder-py3/L15.py", line 63, in <module>
    a += w

NameError: name 'w' is not defined
```

```
  File "/Users/giannidicaro/.spyder-py3/L15.py", line 63, in <module>
    a = 1/0

ZeroDivisionError: division by zero
```

List of python’s exceptions



Exceptions list:

https://docs.python.org/3/library/exceptions.html

Catching multiple exceptions

- ✓ **Multiple, different exceptions can be caught, but only a few may need to be explicitly named**

```
y = 1
x = 10
d = 0.0
try:
    x /= d
    y += x
except ZeroDivisionError:
    print("Division by zero!")
except:
    print("Something went wrong!")
```

Try/Catch with files

- ✓ **Try to open a file for writing, otherwise open a different file if it fails, and at the end always issue a close()**

```
try:
    f = open('sales.dat', 'r')
except FileNotFoundError:
    print('File sales.dat does not exist in the current folder')
    print("I will open another file, that it's for sure in the system")
    f = open('all_sales.dat', 'r')
except:
    print('File sales.dat does exist but it is not readable')
else:
    print("Add data to the file")
    f.write ('New sale: 8000')
finally:
    print("Files must be closed, no error will be thrown if open failed")
    f.close()
```

Try/Catch with files

- ✓ **Try to write on a file, reopen it if it was previously closed**

```
try:
    nbytes = f.write('New data: 1 3 5')
except ValueError:
    print('File was previously closed! To write, I will reopen it')
    f = open('sales.dat', 'a+')
    nbytes = f.write('New data: 1 3 5')
except:
    print('File sales.dat does exist but it is not readable')
finally:
    print("Let's close the file anyway")
    f.close()
```

```
  File "/Users/giannidicaro/.spyder-py3/L15.py", line 65, in <module>
    nbytes = f.write('New data: 1 3 5')

ValueError: I/O operation on closed file.
```

Generating custom exceptions: `assert`

- **Raise an error if an expression is evaluated False**: `assert Expression<, argument>`
- The argument is optional, in its absence no custom message is generated
- ✓ **Sanity-check:** if something is wrong generates an AssertionException error with a user-defined argument
- ✓ The error can be dealt with `try-except`, otherwise it will just abort the program

```
def KelvinToCelsius(temperature):
    assert (temperature >= 0),"Negative Kelvin!"
    return (temperature-273.15)

print (KelvinToCelsius(273))
print (KelvinToFahrenheit(-5))
```

```
Celsius:   -0.150
Traceback (most recent call last):

  File "<ipython-input-182-d1dbd701d3ba>", line 7, in <module>
    print ("Celsius: {:8.3f} ".format(KelvinToCelsius(-5)))

  File "<ipython-input-182-d1dbd701d3ba>", line 2, in KelvinToCelsius
    assert (temperature >= 0),"Negative Kelvin! "

AssertionError: Negative Kelvin, colder than absolute zero!
```

```
try:
    print ("Celsius: {:8.3f} ".format(KelvinToCelsius(273)))
    print ("Celsius: {:8.3f} ".format(KelvinToCelsius(-5)))
except AssertionError:
    print ("Provide a Kelvin temperature >= 0")
```

```
Celsius:   -0.150
Provide a Kelvin temperature >= 0
```