# Programming Thinking

## Session 13

Pepe García jgarciah@faculty.ie.edu

SCHOOL OF SCIENCE & TECHNOLOGY

# Plan for today

- Handling files in Python

SCHOOL OF SCIENCE & TECHNOLOGY

# Plan for today

- Handling files in Python
- CSV files

SCHOOL OF SCIENCE & TECHNOLOGY

# the **open** function

We can use **open()** to open a file in Python, we only need to pass the path of the file we want to open. Let's say there's a file named hello.txt in my desktop that I want to open and read from Python, I can do it as follows:

SCHOOL OF SCIENCE & TECHNOLOGY

# the **open** function

We can use **open()** to open a file in Python, we only need to pass the path of the file we want to open. Let's say there's a file named hello.txt in my desktop that I want to open and read from Python, I can do it as follows:

```
file = open("/Users/pepe/Desktop/hello.txt")
```

SCHOOL OF SCIENCE & TECHNOLOGY

# closing files

Something very important we must do if we're going to deal with files is to close them. Once we've opened them, we can directly call **close()** on them at any point to close them.

# closing files

Something very important we must do if we're going to deal with files is to close them. Once we've opened them, we can directly call **close()** on them at any point to close them.

```
file = open("/Users/pepe/Desktop/hello.txt")

# deal with the file

file.close()
```

SCHOOL OF SCIENCE & TECHNOLOGY

# Reading the contents of a file

Now that we know how to open and close files, we can read the contents of a file. Let's do that line by line.

# Reading the contents of a file

Now that we know how to open and close files, we can read the contents of a file. Let's do that line by line.

```
file = open("/Users/pepe/Desktop/hello.txt")

for line in file:
    print(line)

file.close()
```

# Reading the contents of a file

Now that we know how to open and close files, we can read the contents of a file. Let's do that line by line.

```
file = open("/Users/pepe/Desktop/hello.txt")

for line in file:
    print(line)

file.close()
```

As you can see, we're treating `file` as a list of lines.

SCHOOL OF SCIENCE & TECHNOLOGY

## Be careful with closed files

If you try to operate with a file that has already been closed, you'll see an error.

```
file = open("/Users/pepe/Desktop/hello.txt")

# In this line we're closing the file
file.close()

# trying to do this will cause an error
for line in file:
    print(line)
```

SCHOOL OF SCIENCE & TECHNOLOGY

## Be careful with closed files

If you try to operate with a file that has already been closed, you'll see an error.

```
file = open("/Users/pepe/Desktop/hello.txt")

# In this line we're closing the file
file.close()

# trying to do this will cause an error
for line in file:
    print(line)
```

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

SCHOOL OF SCIENCE & TECHNOLOGY

# Interlude, **with**

there is a useful Python keyword that one can use to make sure that the file will always be closed, **with**:

SCHOOL OF SCIENCE & TECHNOLOGY

# Interlude, **with**

there is a useful Python keyword that one can use to make sure that the file will always be closed, **with**:

```
with open("file_path") as file:

    for line in file:
        #do something with line
        print(line)
```

SCHOOL OF SCIENCE & TECHNOLOGY

# Handling files. modes

When opening a file, we can choose in which **mode** we open it depending on how we're going to use it.

| I/O Mode | Syntax | Behavior |
|---|---|---|
| Read | 'r' | Opens the contents of a file for reading into the file interface, allowing for lines to be read-in successively. |
| Write | 'w' | Creates a file with the specified name and allows for text to be written to the file; note that specifying a pre-existing filename will overwrite the existing file. |
| Append | 'a' | Opens an existing file and allows for text to be written to it, starting at the conclusion of the original file contents. |
| Read and Write | 'r+' | Opens a file such that its contents can be both read-in and written-to, thus offering great versatility. |

Python's available file-access modes are summarized here.

doi:10.1371/journal.pcbi.1004867.t004

Figure 1: file modes

SCHOOL OF SCIENCE & TECHNOLOGY

# Writing files

We can write into files in a way similar to the one used for reading them.

# Writing files

We can write into files in a way similar to the one used for reading them.

```
with open('/Users/pepe/Desktop/goodbye.txt', 'w') as file
    file.write("goodbye y'all!")
```

SCHOOL OF SCIENCE & TECHNOLOGY

# Checkpoint

## Checkpoint

Is everything clear so far? do you have any question?

# CSV files

Python comes with a **CSV** library that we can use out of the box. We use it by **importing** it. **Imports** are commonly added at the top of the file.

```
import csv
```

SCHOOL OF SCIENCE & TECHNOLOGY

# CSV files

The **csv** library is based on the idea of readers and writers. One can read all lines in a file like so:

```
with open("file.csv") as f:
    reader = csv.reader(f)
    for line in reader:
        print(line)  #line will be a list here
```

SCHOOL OF SCIENCE & TECHNOLOGY

# CSV files

The **csv** library is based on the idea of readers and writers. One can read all lines in a file like so:

```
with open("file.csv") as f:
    reader = csv.reader(f)
    for line in reader:
        print(line)  #line will be a list here
```

first we open the file normally

SCHOOL OF SCIENCE & TECHNOLOGY

# CSV files

The **csv** library is based on the idea of readers and writers. One can read all lines in a file like so:

```
with open("file.csv") as f:
    reader = csv.reader(f)
    for line in reader:
        print(line)  #line will be a list here
```

first we open the file normally

Then we create a reader using **csv.reader()**

SCHOOL OF SCIENCE & TECHNOLOGY

# CSV files

The **csv** library is based on the idea of readers and writers. One can read all lines in a file like so:

```
with open("file.csv") as f:
    reader = csv.reader(f)
    for line in reader:
        print(line)  #line will be a list here
```

first we open the file normally

Then we create a reader using **csv.reader()**

Finally, we operate with the reader

# CSV files

writing is not very different from reading:

```
lines = [
   ["asdf", "qwer"],
   ["hello", "world"]
]

with open("file.csv", "a") as f:
    writer = csv.writer(f)
    for line in lines:
        writer.writerow(line)
```

SCHOOL OF SCIENCE & TECHNOLOGY

# CSV files

writing is not very different from reading:

```
lines = [
   ["asdf", "qwer"],
   ["hello", "world"]
]

with open("file.csv", "a") as f:
    writer = csv.writer(f)
    for line in lines:
        writer.writerow(line)
```

First we need some data to put in the csv file

SCHOOL OF SCIENCE & TECHNOLOGY

# CSV files

writing is not very different from reading:

```
lines = [
   ["asdf", "qwer"],
   ["hello", "world"]
]

with open("file.csv", "a") as f:
    writer = csv.writer(f)
    for line in lines:
        writer.writerow(line)
```

First we need some data to put in the csv file

Then we open the file with the append mode

. . . Later, we create a **csv.writer**

SCHOOL OF SCIENCE & TECHNOLOGY

# CSV files

writing is not very different from reading:

```
lines = [
   ["asdf", "qwer"],
   ["hello", "world"]
]

with open("file.csv", "a") as f:
    writer = csv.writer(f)
    for line in lines:
        writer.writerow(line)
```

First we need some data to put in the csv file

Then we open the file with the append mode

. . . Later, we create a **csv.writer**

SCHOOL OF SCIENCE & TECHNOLOGY