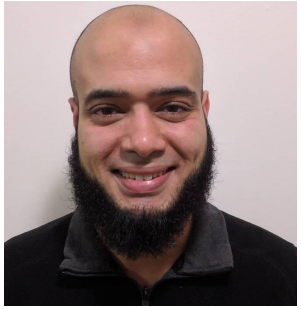
Python Programming

Writing to files

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



Let’s Write!

- The write method doesn’t add new line. If you want it, you have to provide it
- If the file doesn’t exist, it will be created
  - Error if not possible: e.g. invalid path or security permission issues!
- By default, the old content will be overwritten

```python
path = 'output1.txt'

with open(path, 'w') as file:
    file.write('Hey')
    file.write('Your name?')

# let's run this code twice.
# observe: file will be created if not exist
```

output1.txt × 01.py ×

```
HeyYour name?
```

Printing lines

- Just add \n to force printing new lines

```python
path = 'output2.txt'

lines = ['Hey', 'Your name?']

# w for write but overwrite
with open(path, 'w') as file:
    for line in lines:
        file.write(line + '\n')
```

output2.txt

```
Hey
Your name?
```

Appending mode

- The append mode just keep adding things to the end of the file
  - Each run will add new content, not overwriting

```python
path = 'output3.txt'

lines = ['Hey', 'Your name?']

# a for write but append
with open(path, 'a') as file:
    for line in lines:
        file.write(line + '\n')

# let's run this code twice.
```

output3.txt

```
Hey
Your name?
Hey
Your name?
```

Read and Write

- In same with statement, we can open several files

```
2   input_path = 'input.txt'
3   output_path = 'output.txt'
4
5   with open(input_path, 'r') as reader, \
6       open(output_path, 'w') as writer:
7       lines = reader.readlines()
8       writer.writelines(reversed(lines))
9       # writelines doesn't add \n
10
```

Fail if exists

- Sometimes, you want your code works well only if you are creating
  - Neither overwriting nor appending is expected

```python
path = 'output4.txt'

lines = ['Hey', 'Your name?']

# x: if exist = errio
with open(path, 'x') as file:
    for line in lines:
        file.write(line + '\n')

# let's run this code twice.
# second time error:
# FileExistsError: [Errno 17] File exists:
#     'output4.txt'
```

Mix reading and writing

- We can mix reading/writing use **r+** and **w+** but this might be problematic
- There is also .seek functionalities
  - You may study this later in **file structures** course

os.linesep

- Import os
- **os.linesep** is the line separator (e.g. \n linux or \r\n windows)
- One might think to add \r\n during writing for windows
- However, behind the scene these conversions in reading/writing are done
  - Specifically for the normal text mode
    - E.g. when you print the line, it will always have \n regardless the platform
  - Note: in binary mode such conversions doesn’t occur
- Tip: Stick to \n in writing in text mode

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."