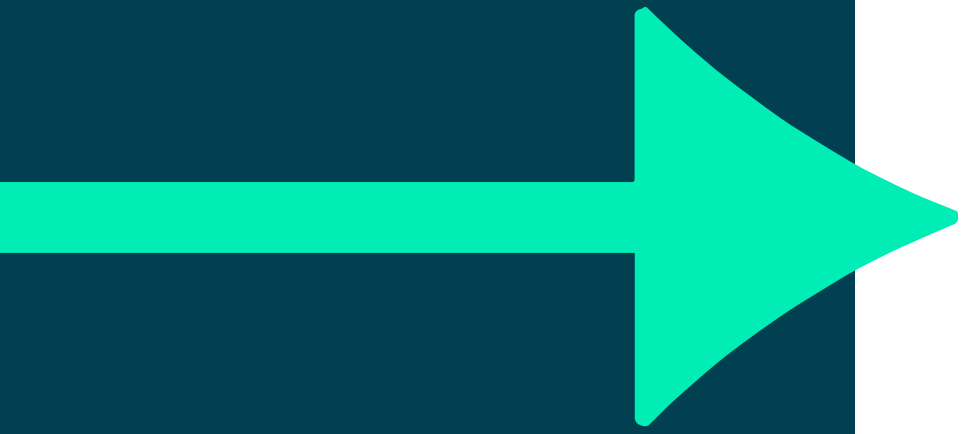
POWERING POTENTIAL

File Input and Output

LESSON OBJECTIVES



In this chapter, you'll learn how to:

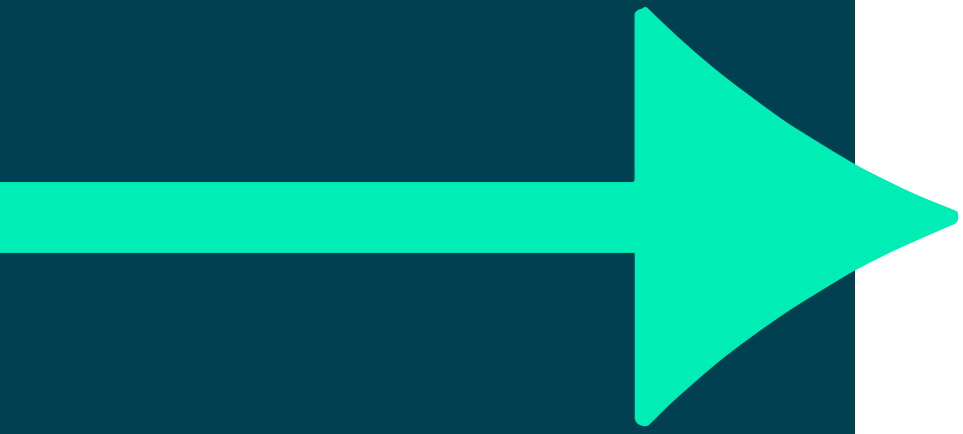
- Read data from file in Python
- Write data to file in Python

LET'S WARM UP

Write a python program which opens a text file ('hello.txt'), reads the whole its content, and prints the whole content of that file.

- → **What do you do when you don't know what to do?**
- → Find suitable code on the internet and adapt it to the task.
- → You will find multiple ways, including the use of some of the Python libraries.
- → There are multiple ways of doing almost everything. We call them "model solutions." Select one or another depending what suits our task best.

FILE INPUT - SOLUTIONS, SOLUTIONS



```
# Solution 1
f = open('hello.txt', 'r')
print(f.read())

# Solution 2
# If the file is in a different location, you have to specify the path
file_location = 'hello.txt'
file_pointer = open(file_location, 'r')
file_content = file_pointer.read()
print(file_content)

# Solution 3
print(open('hello.txt', 'r').read())

# Solution 4 - a bit more advanced
with open('hello.txt', 'r') as MyFile:
        interesting = MyFile.read()
        print(interesting)
```

THE OPEN() FUNCTION

The open() function takes two parameters: file name and mode. Only the file name is mandatory.

There are four different modes for opening a file:

"r" - Read - Default value. Opens a file for reading, error if the file does not exist.

"a" - Append - Opens a file for appending, creates the file if it does not exist.

"w" - Write - Opens a file for writing, creates the file if it does not exist.

"x" - Create - Creates the specified file, returns an error if the file exists.

In addition, you can specify if the file should be handled as binary or text mode.

"t" - Text - Default value. Text mode

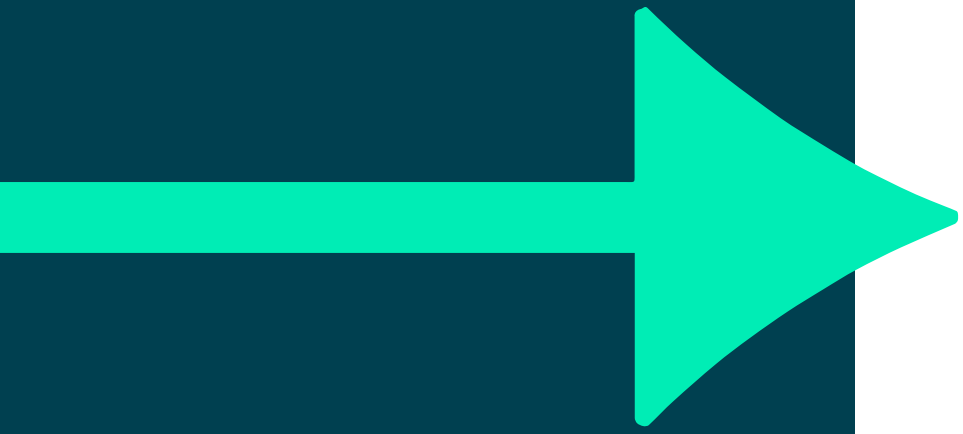
"b" - Binary - Binary mode (e.g., images)

THE OPEN() FUNCTION

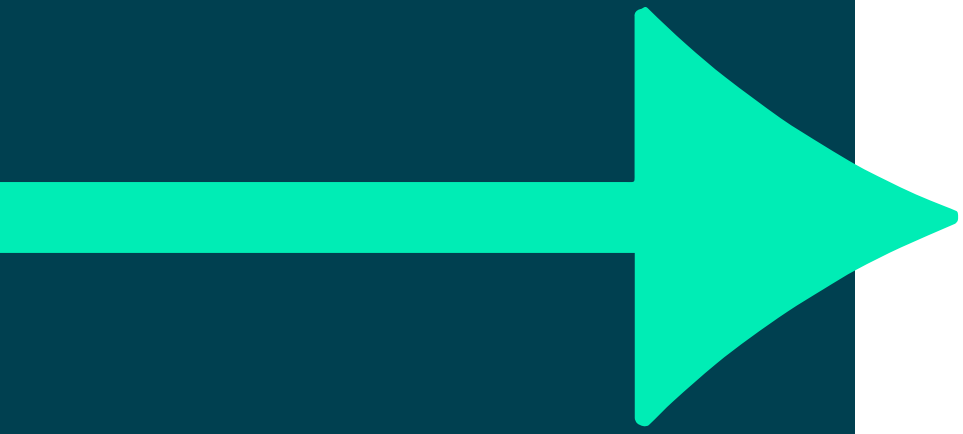
The open() function takes two parameters: file name and mode.

The following are equivalent:

```
f = open('hello.txt')
f = open('hello.txt', 'r')
f = open('hello.txt', ‘rt')
```



FILE INPUT



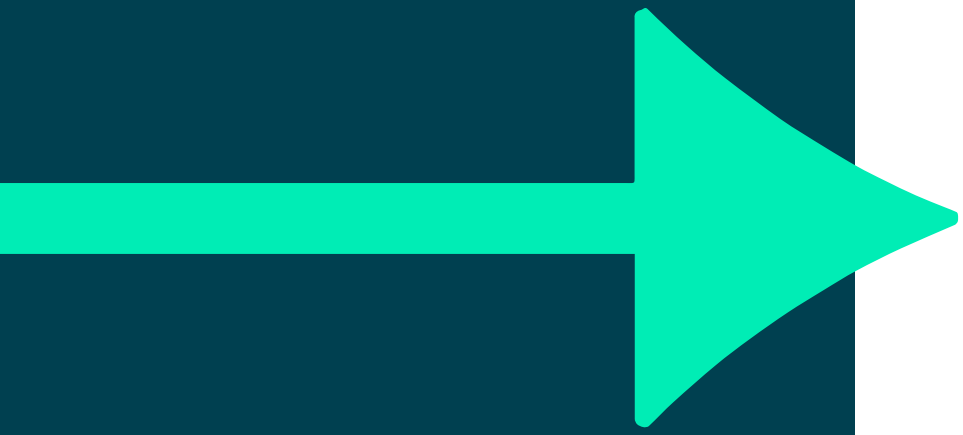
You can not only read the whole text, but you can also specify what part of it.

```
# The first 5 characters of the file
f = open('hello.txt', 'r')
print(f.read(5))

# The first line of the file
f = open('hello.txt', 'r')
print(f.readline())

# The first 2 lines of the file
f = open('hello.txt', 'r')
print(f.readline())
print(f.readline())
```

WORKING WITH A FILE



Let's read file data.txt (supplied):

```
# locate the file
a = 'data.txt'
# open the file for reading
f = open(a, 'r')
# read the whole content of that file into a single string variable
b = f.read()
# print it
print(b)
```

```
3
5
-2
11
0
7
1
```

Even though it looks like multiple lines, technically variable b will contain this:

'3\n5\n-2\n11\n0\n7\n1'

There are NO new lines (\n) after the last element, i.e. 1 is the last character)

WORKING WITH A FILE

'3\n5\n-2\n11\n0\n7\n1'

```
# split the string variable b into array of strings
c = b.split('\n')
print(c)
```

```
['3', '5', '-2', '11', '0', '7', '1']
```

The code so far can be written in more concise form:

```
file_content = open('data.txt', 'r').read().split('\n')
print(file_content)
```

```
['3', '5', '-2', '11', '0', '7', '1']
```

And then the list with the file content can be further processed as needed.

NOTE: The list elements are strings, not numbers.

INPUT FROM FILE WITH A HEADER

To remove a header first line:

 # open the file

f = open('data2.txt', 'r')

 # skip the first line (by reading and discarding)

f.readline()

 # read the rest

file_content = f.read().split('\n')

```
# declare an empty array (output will be accumulated here)
data = []

# iterate over the array
for x in file_content:
    # print(x)
    x = x.strip()
    # x = int(x) # this will fail because of empty lines
    if (x != ''):
        x = int(x)
        data.append(x)
# print the output
print(data)
```

```
[3, 5, -2, 11, 0, 7, 1, 0]
```

FILE OUTPUT

To write to a file, open it with one of the following modes:

"a" - Append - Opens a file for appending, creates the file if it does not exist.

"w" - Write - Opens a file for writing, creates the file if it does not exist.

Use the write() function to write output to the file.

Don’t forget to close the file.

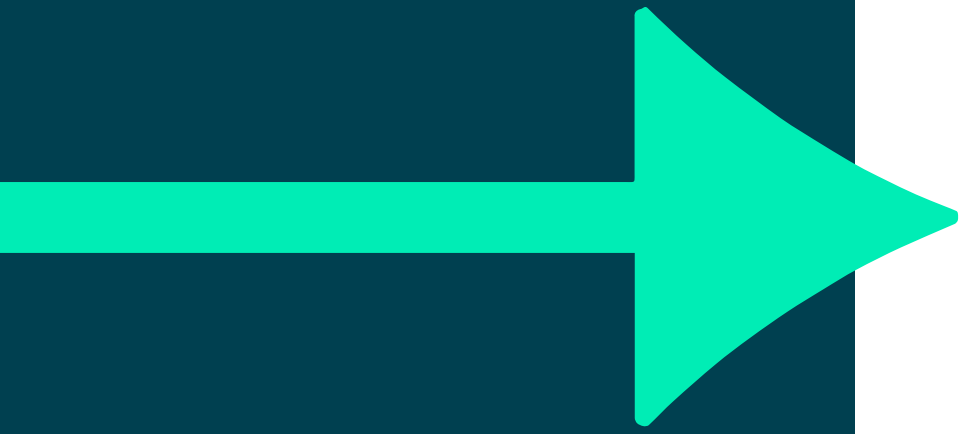
```
f = open('output.txt', 'w’)
f.write(‘Hello’)
f.close()
```

CLOSING A FILE

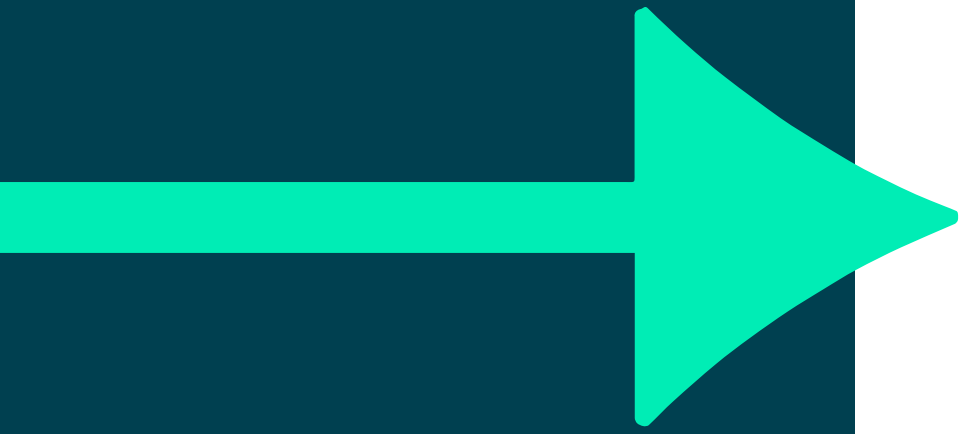
 # It is a very good practice to always close the file when you have finished with it:

f.close()

NOTE: In some cases, due to buffering, changes made to a file may not show until the file is closed.

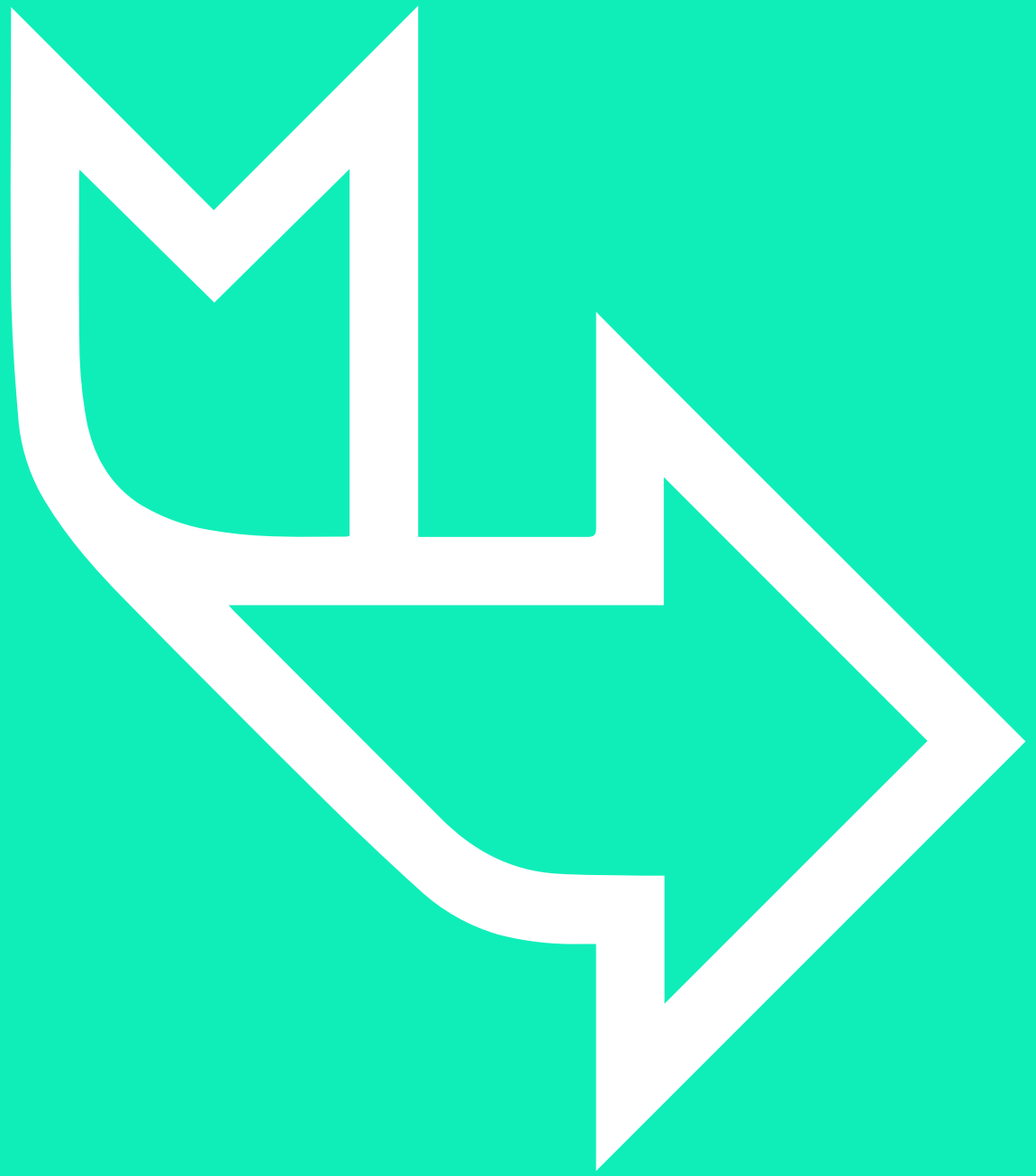


SUMMARY



In this chapter, you've learned how to:

- Read data from file in Python
- Write data to file in Python



Further Reading

https://www.python.org/