CS 1340 Introduction to Computing Concepts

Instructor: Xinyi Ding
Sep 18 2019, Lecture 10

Announcement

- Quiz next Friday (Sep 27)
  - may take ~10mins at the beginning of the class
- Midterm Exam will be on Oct 18 (Friday, after fall break).
- Details about topics covered will be posted on canvas

Agenda

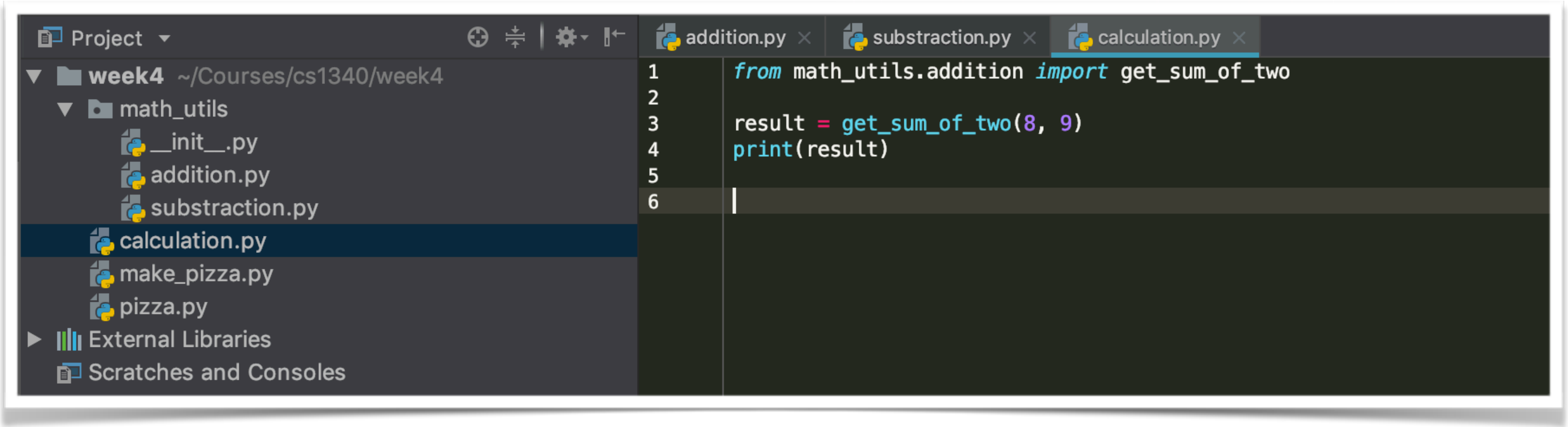
- Agenda:
  - Quick review of concepts from last lecture
  - File Handling
  - Exceptions

Modules

- Consider a module to be the same as a code library
- A file containing a set of functions you want to include in your application
- To create a module just save the code you want in a file with the extension .py
- Use import statement to include the code you want to use

Packages

- Put related modules in a package
  - A directory with a __init__.py file
  - Organize related modules



```python
from math_utils.addition import get_sum_of_two

result = get_sum_of_two(8, 9)
print(result)

```

Python Input/Output

- Up till now, our program were static. The value of variables were defined or hard coded into the source code.
- To Allow flexibility we might want to take the input from the user. We use input() function in python

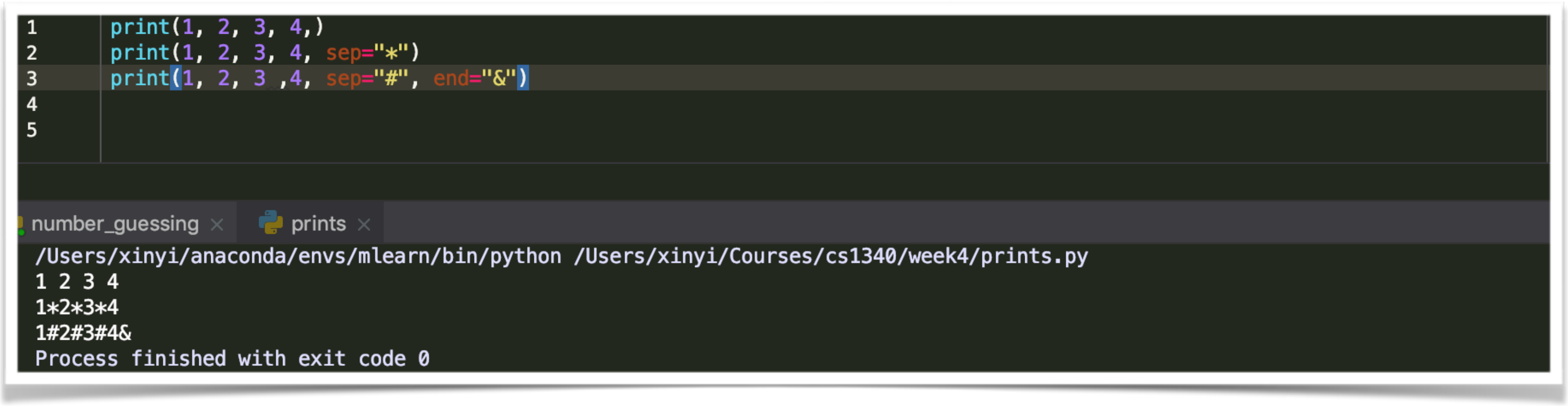
```
input([prompt])
```

Where prompt is the string we wish to display on the screen, it is optional

Python Input/Output

- Input() takes input from the standard device(screen)
- print() outputs data to the standard output device (screen)
- The actual syntax of the print() function is

print(*objects, sep=' ', end='\n', file=sys.std, flush=False)



```
print(1, 2, 3, 4,)
print(1, 2, 3, 4, sep="*")
print(1, 2, 3 ,4, sep="#", end="&")
```

number_guessing × prints ×

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/prints.py
1 2 3 4
1*2*3*4
1#2#3#4&
Process finished with exit code 0
```

Demo

File Handling

- Input() and print() allow you to get input from and output data to the standard device (screen)
- Often the case, you need to write your data to a file for later use or reading data from files. This makes your program more usable.
- Python has several functions for creating, reading, updating, and deleting files

File Handling

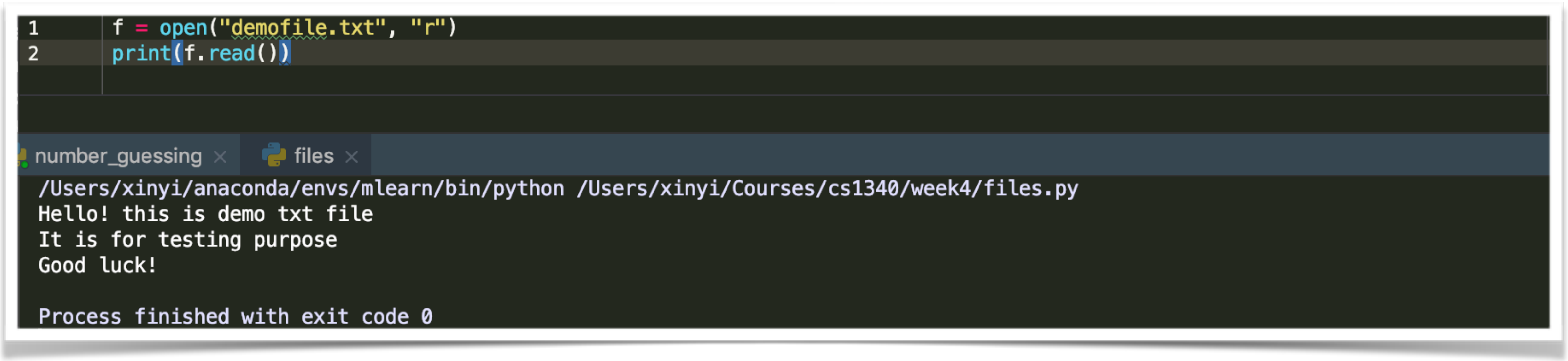
- The key function for working with files is the open() function.
- The open() function takes two parameters: *filename* and *mode*
- There are four different modes for opening a file
  - "r" - Read-Default value. Opens a file for reading, error if the file does not exist
  - "a" - Append- Opens a file for appending, creates the file if it does not exist
  - "w"- Write- Opens a file for writing, create the file if it does not exist
  - "x" - Create -Creates the specified file, returns an error if the file exists

File Handling

- 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)
- However, when working with special file like images or cvs, json files, we usually use third party libraries, rather than using the built in functions.

File Handling

- Open a file
  - To open the file, use the built-in open() function.
  - The open() function returns a file object, which has a read() method for reading the content of the file

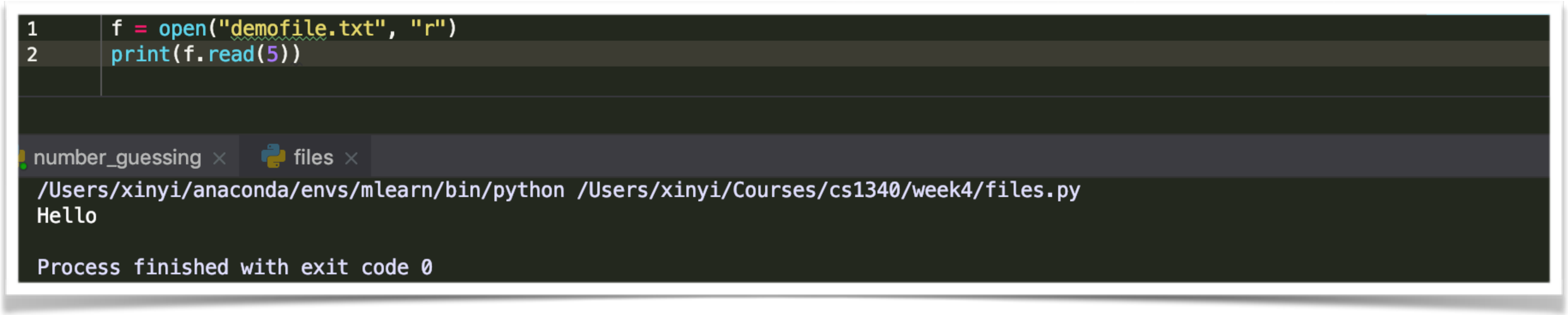


```python
f = open("demofile.txt", "r")
print(f.read())
```

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/files.py
Hello! this is demo txt file
It is for testing purpose
Good luck!

Process finished with exit code 0
```

File Handling

- Read parts of the file
    - use read() method and specify how many characters to read.

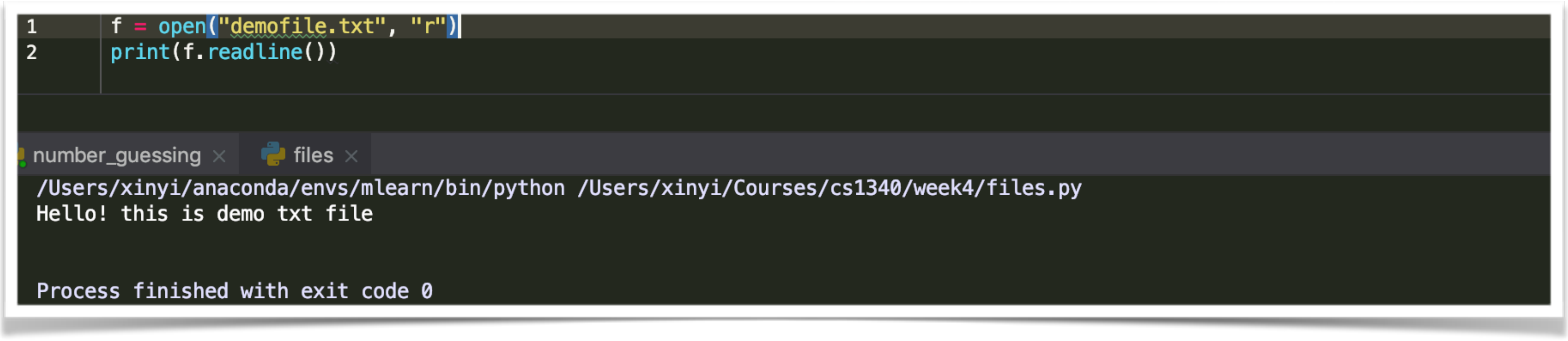


```
f = open("demofile.txt", "r")
print(f.read(5))
```

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/files.py
Hello

Process finished with exit code 0
```

File Handling

- Read lines
    - You can return one line by using the readline() method

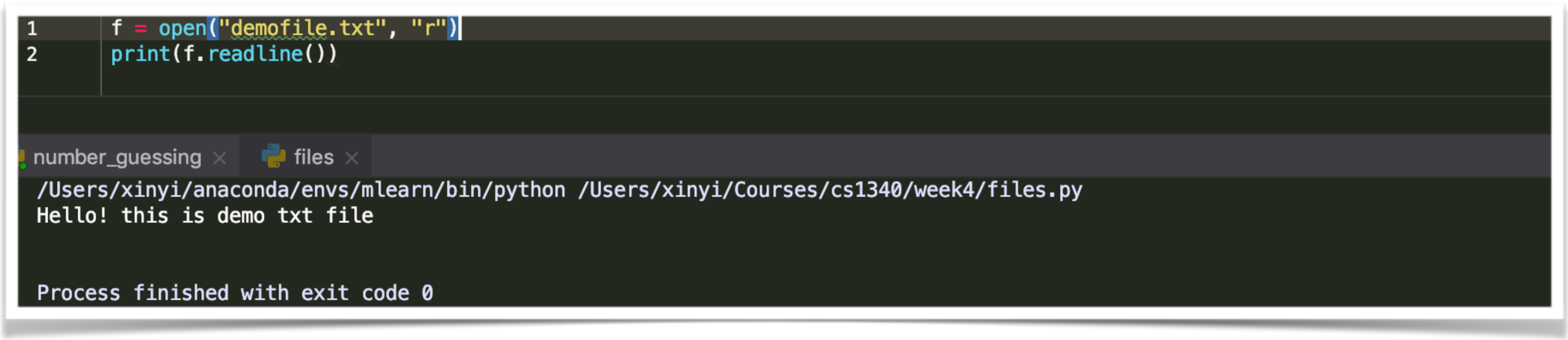


```
f = open("demofile.txt", "r")
print(f.readline())
```

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/files.py
Hello! this is demo txt file


Process finished with exit code 0
```

File Handling

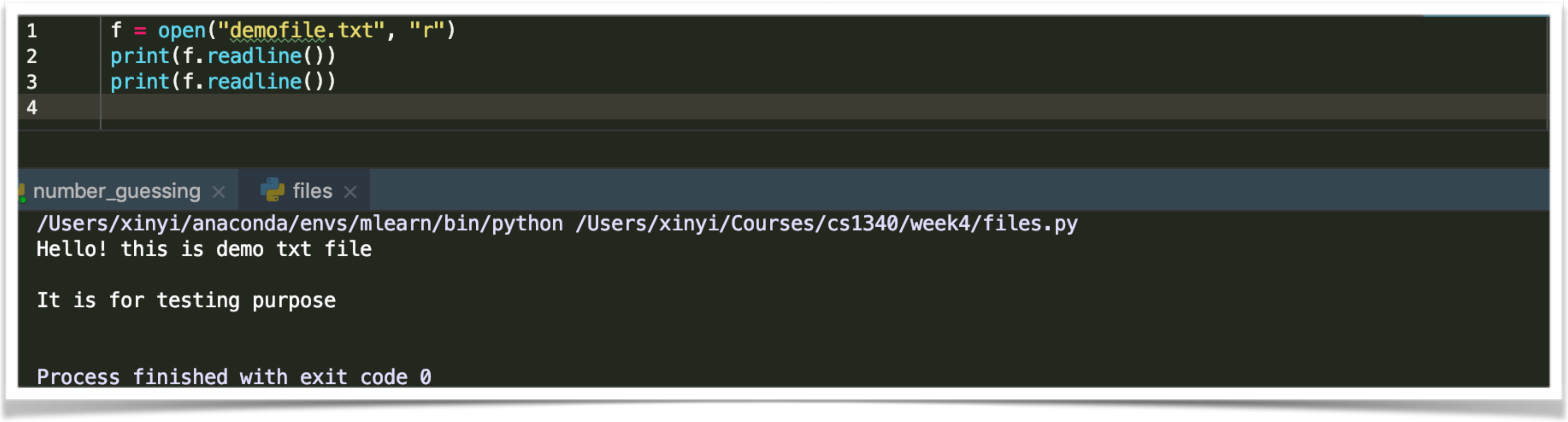
- Read lines
    - You can return one line by using the readline() method



```
f = open("demofile.txt", "r")
print(f.readline())
```

number_guessing × files ×

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/files.py
Hello! this is demo txt file


Process finished with exit code 0
```



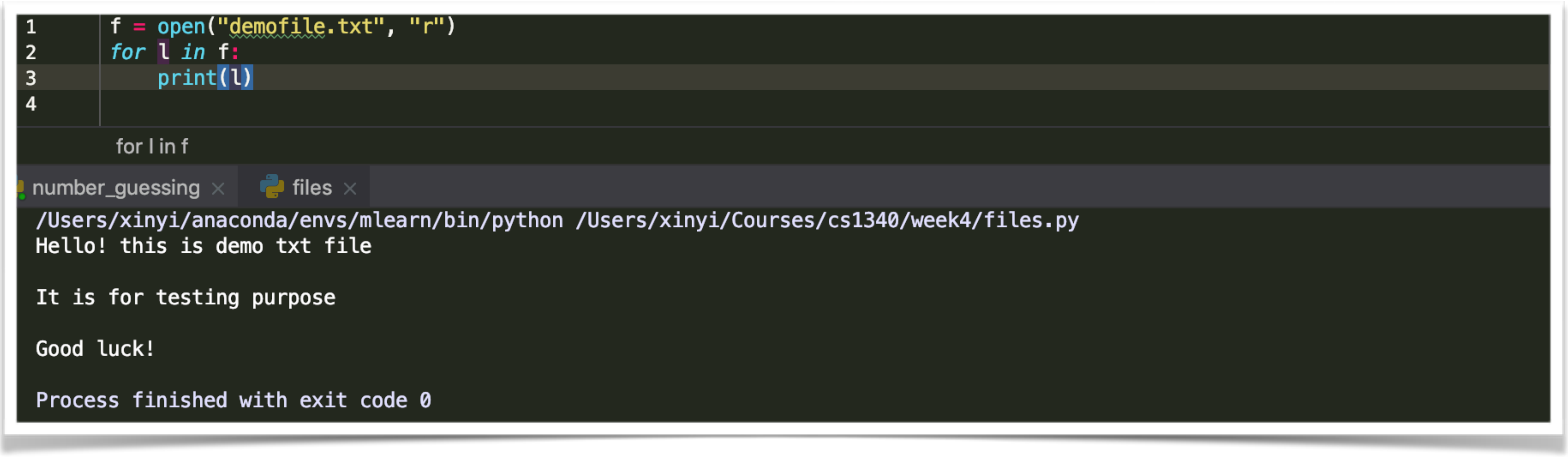
```
f = open("demofile.txt", "r")
print(f.readline())
print(f.readline())

```

number_guessing × files ×

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/files.py
Hello! this is demo txt file

It is for testing purpose


Process finished with exit code 0
```

File Handling

- Read lines
  - Loop through the file line by line

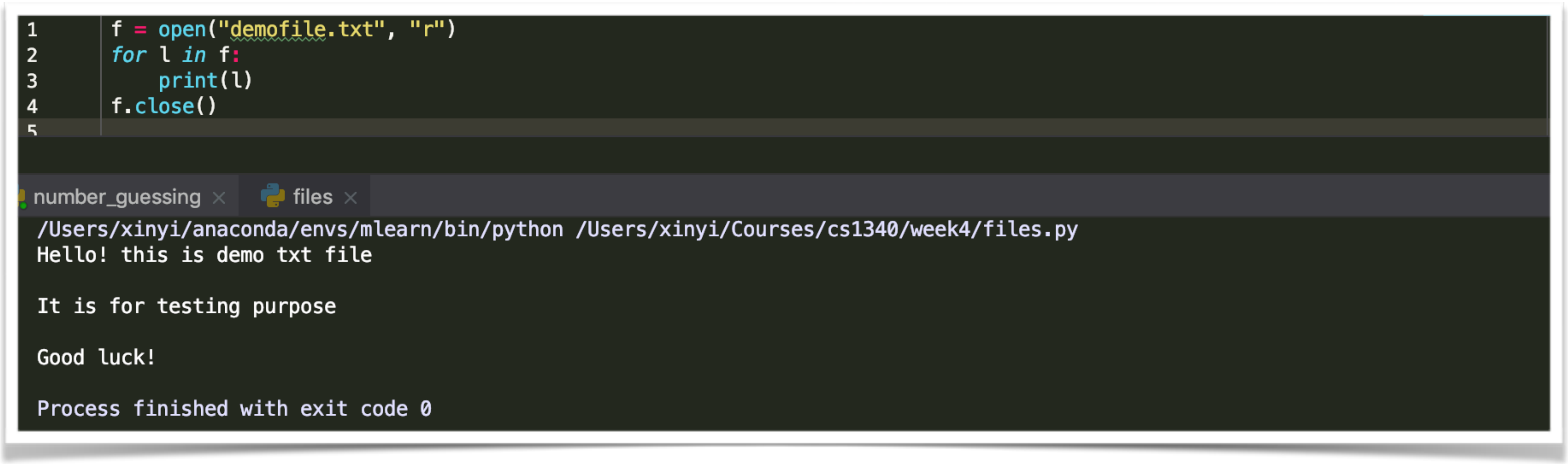


```python
f = open("demofile.txt", "r")
for l in f:
    print(l)

```

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/files.py
Hello! this is demo txt file

It is for testing purpose

Good luck!

Process finished with exit code 0
```

File Handling

- Close files
  - You should always close your files, in some cases, due to buffering, change made to a file may not show until you close the file



```python
f = open("demofile.txt", "r")
for l in f:
    print(l)
f.close()
```

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/files.py
Hello! this is demo txt file

It is for testing purpose

Good luck!

Process finished with exit code 0
```

Demo

File Handling

- With Statement
  - The keyword with closes the file once access to it is no longer needed
  - with syntax:

Indentation(4 spaces)

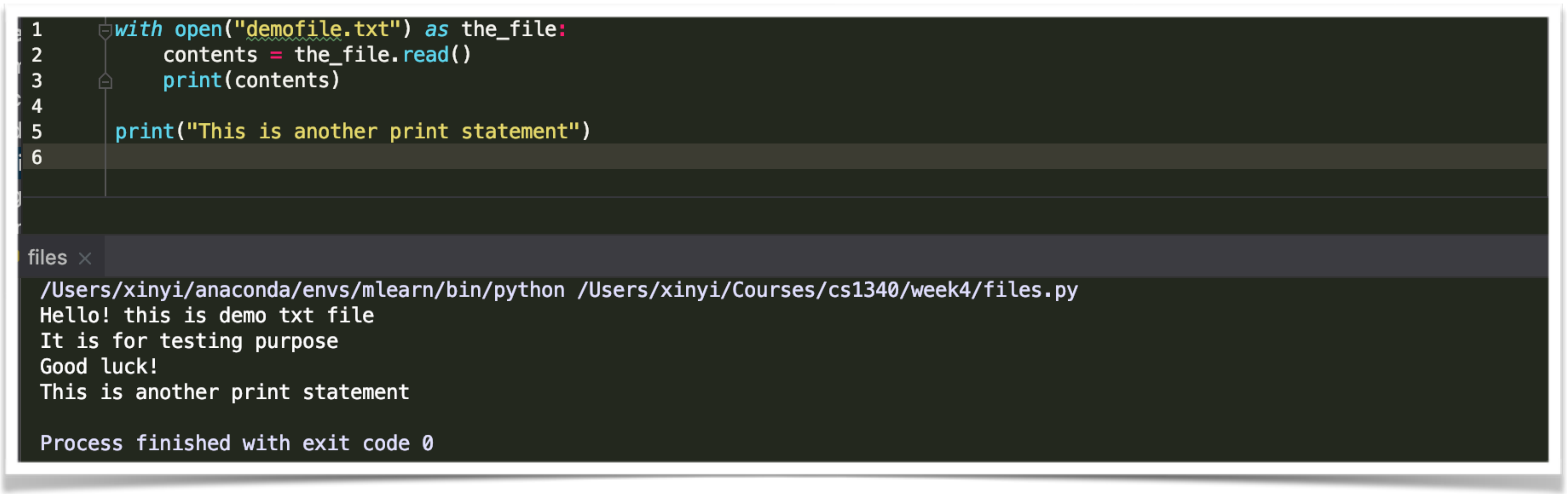
```
with  open(filename) as the_file:
    contents = the_file.read()
    print(contents)
...
```

Colon

First line with less indentation is considered to be outside of the with code block

File Handling

- With Statement



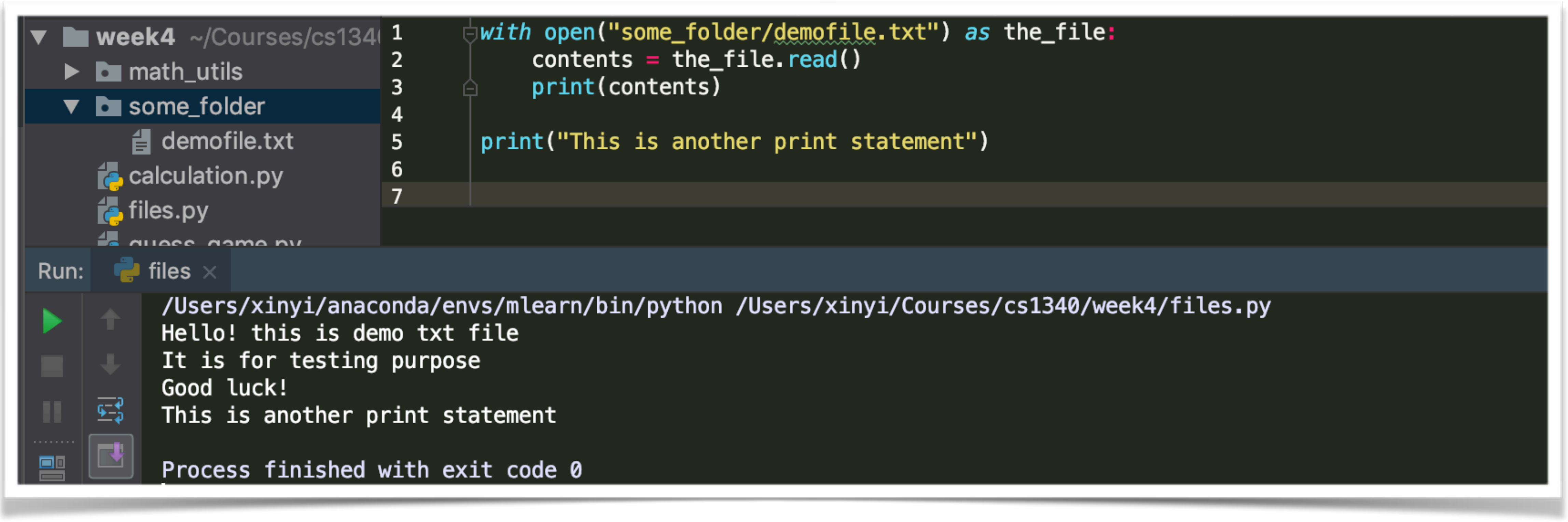
```python
with open("demofile.txt") as the_file:
    contents = the_file.read()
    print(contents)

print("This is another print statement")

```

files

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/files.py
Hello! this is demo txt file
It is for testing purpose
Good luck!
This is another print statement

Process finished with exit code 0
```

File Handling

- File Paths
  - when you pass a simple filename to open() function, Python looks in the directory where the file that's currently being executed



```python
with open("some_folder/demofile.txt") as the_file:
    contents = the_file.read()
    print(contents)

print("This is another print statement")
```

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/files.py
Hello! this is demo txt file
It is for testing purpose
Good luck!
This is another print statement

Process finished with exit code 0
```

File Handling

- Using absolute file path
  - On Linux and MacOS, absolute paths look like this:

```
file_path = 'home/eric/other_files/text_files/filename.txt'
with open(file_path) as file_object:
```

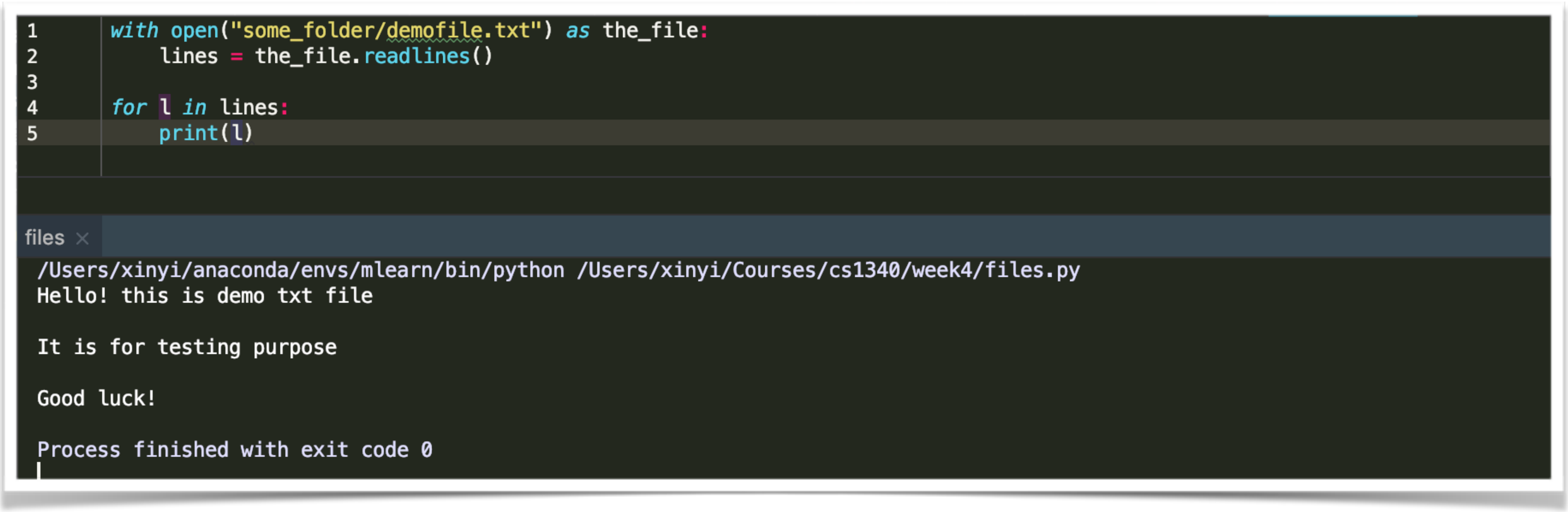
  - On Windows they look like this:

```
file_path = 'C:\users\eric\other_files\text_files\filename.txt'
with open(file_path) as file_object:
```

Note: Hard code an absolute file path may not be a good idea. May cause error if run at a different operating system. Using pathlib for more info https://docs.python.org/3/library/pathlib.html

File Handling

- Making a list of lines from a file
  - The file object returned by open() is only available inside the with block that contains it
  - Store lines to list for later process

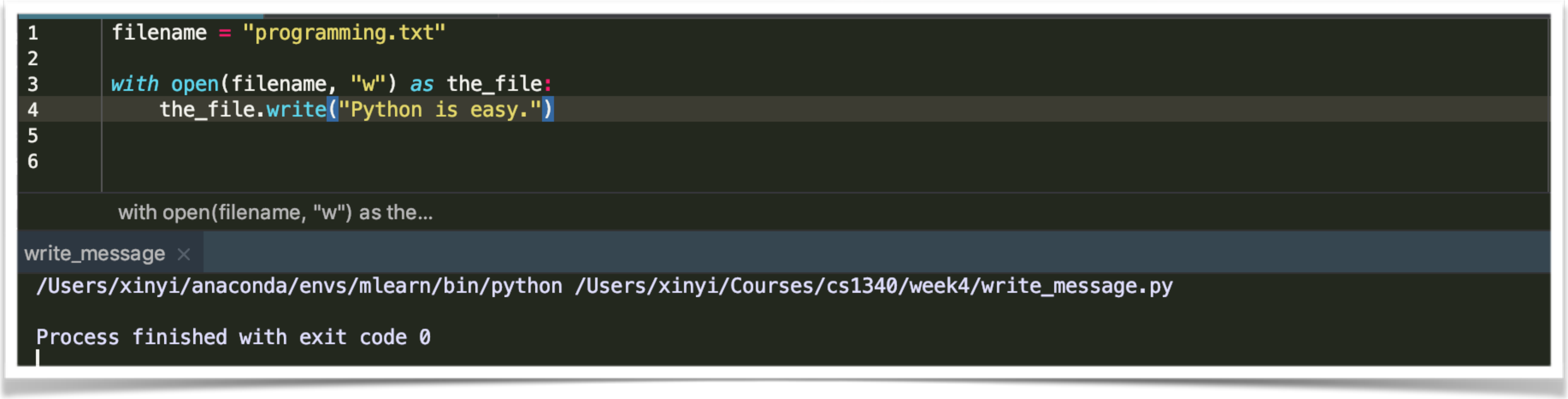


```python
with open("some_folder/demofile.txt") as the_file:
    lines = the_file.readlines()

for l in lines:
    print(l)
```

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/files.py
Hello! this is demo txt file

It is for testing purpose

Good luck!

Process finished with exit code 0
```

File Handling

- Writing to an empty file
  - specify the "w" write mode
  - will create a new file if not exist



```python
filename = "programming.txt"

with open(filename, "w") as the_file:
    the_file.write("Python is easy.")


```

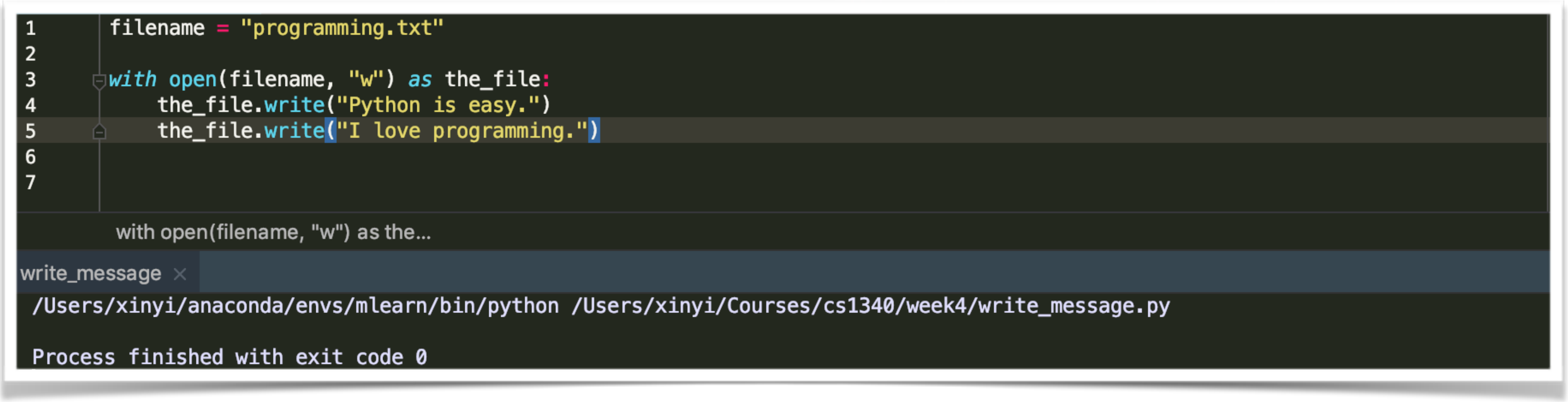
with open(filename, "w") as the...

write_message

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/write_message.py

Process finished with exit code 0
```

File Handling

- Writing multiple lines
  - specify the "w" write mode
  - will create a new file if not exist
  - will overwrite the original content if file exist



```python
filename = "programming.txt"

with open(filename, "w") as the_file:
    the_file.write("Python is easy.")
    the_file.write("I love programming.")
```

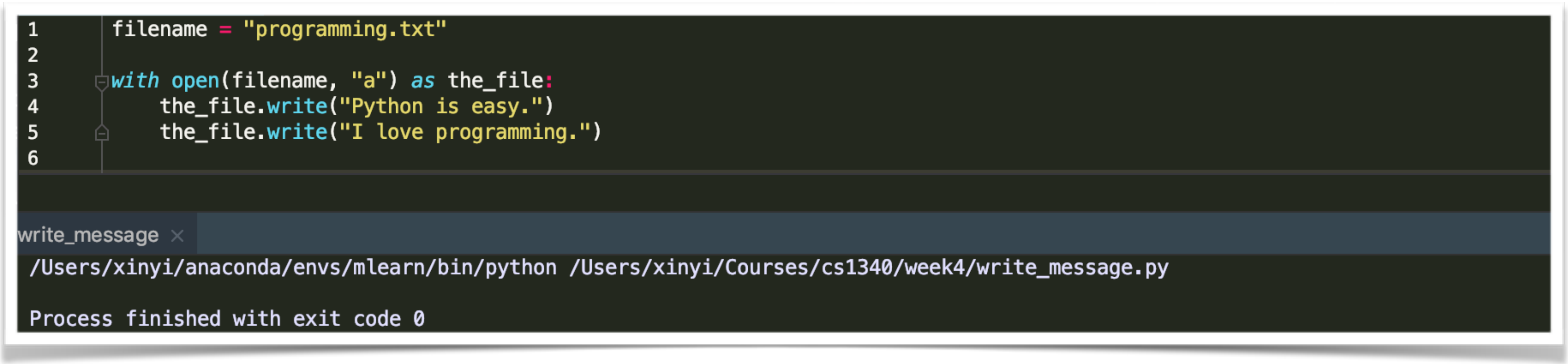
with open(filename, "w") as the...

write_message

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/write_message.py

Process finished with exit code 0
```

File Handling

- Appending to a file
  - use the "a" appending mode
  - will create a new file if not exist



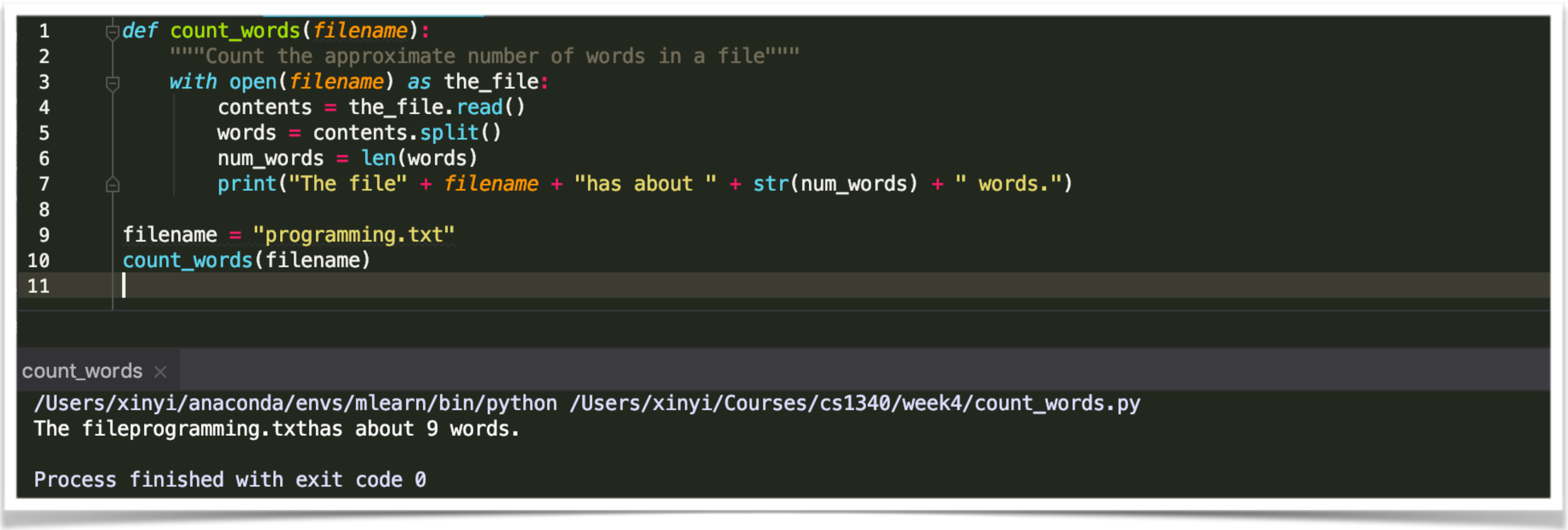
```python
filename = "programming.txt"

with open(filename, "a") as the_file:
    the_file.write("Python is easy.")
    the_file.write("I love programming.")
```

write_message

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/write_message.py

Process finished with exit code 0
```

File Handling

- Work with multiple files



```
def count_words(filename):
    """Count the approximate number of words in a file"""
    with open(filename) as the_file:
        contents = the_file.read()
        words = contents.split()
        num_words = len(words)
        print("The file" + filename + "has about " + str(num_words) + " words.")

filename = "programming.txt"
count_words(filename)

```

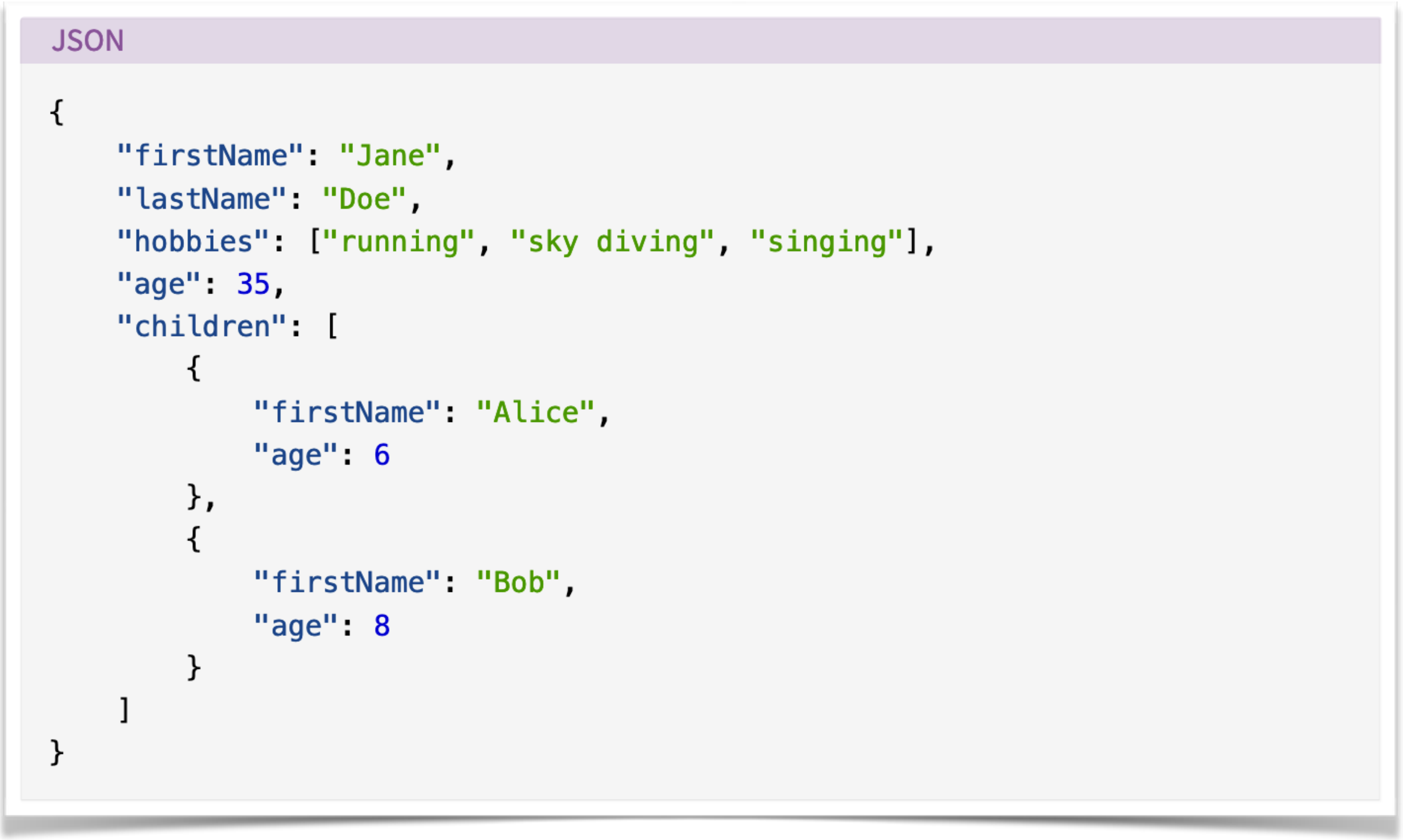
count_words ×

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/count_words.py
The fileprogramming.txthas about 9 words.

Process finished with exit code 0
```

Demo

Work with json in Python

- de facto standard for information exchange.
    - gathering information through an API
    - store your data in a document
- Other format include XML, YAML

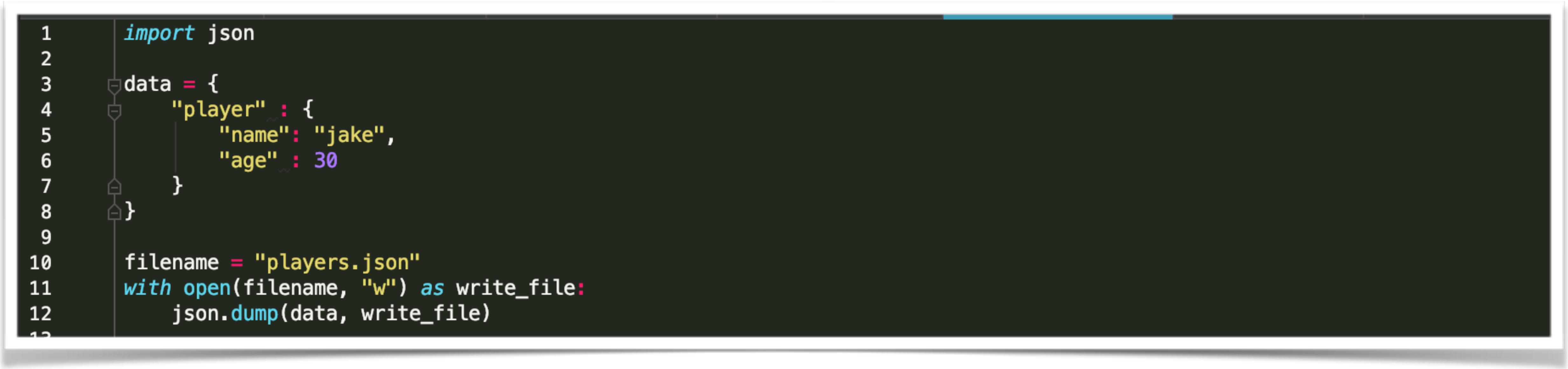


JSON

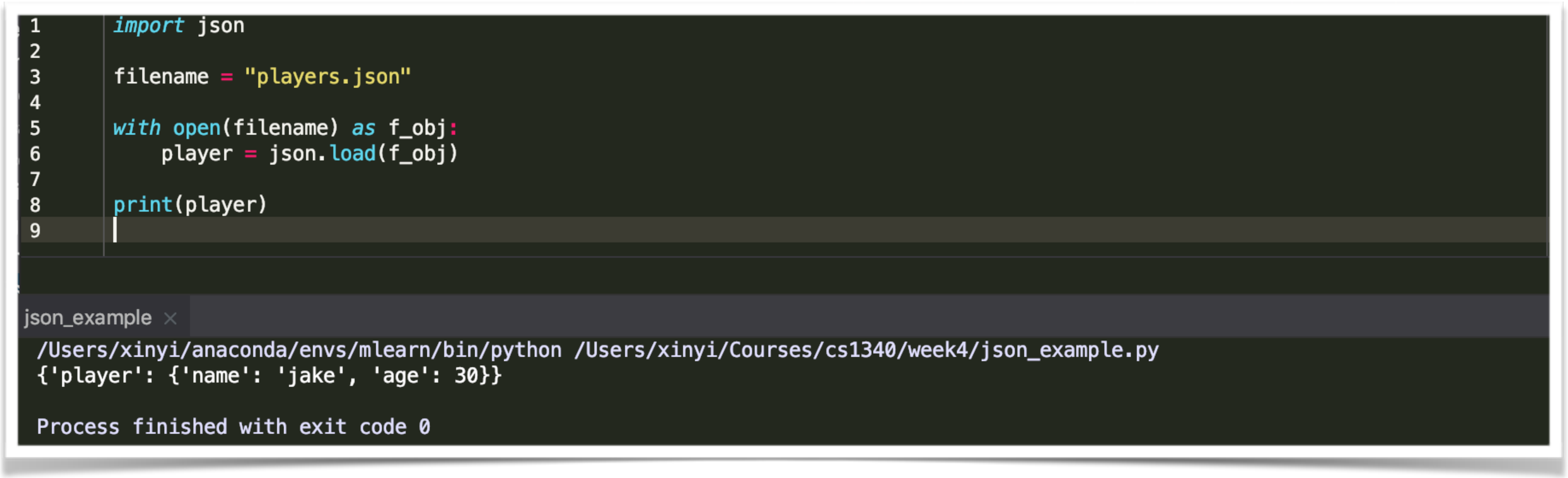
```
{
    "firstName": "Jane",
    "lastName": "Doe",
    "hobbies": ["running", "sky diving", "singing"],
    "age": 35,
    "children": [
        {
            "firstName": "Alice",
            "age": 6
        },
        {
            "firstName": "Bob",
            "age": 8
        }
    ]
}
```

Store data using json

- Saving data to disk in a more organized way using json (instead of plain text)



```python
import json

data = {
    "player" : {
        "name": "jake",
        "age" : 30
    }
}

filename = "players.json"
with open(filename, "w") as write_file:
    json.dump(data, write_file)
```



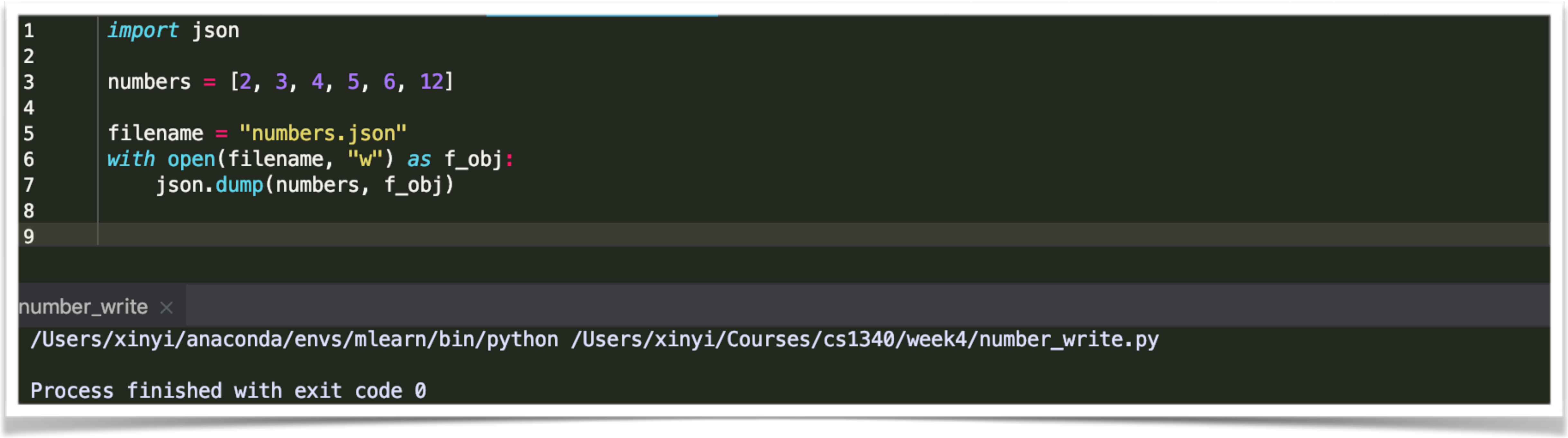
```python
import json

filename = "players.json"

with open(filename) as f_obj:
    player = json.load(f_obj)

print(player)
```

json_example

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/json_example.py
{'player': {'name': 'jake', 'age': 30}}

Process finished with exit code 0
```

Store data using json

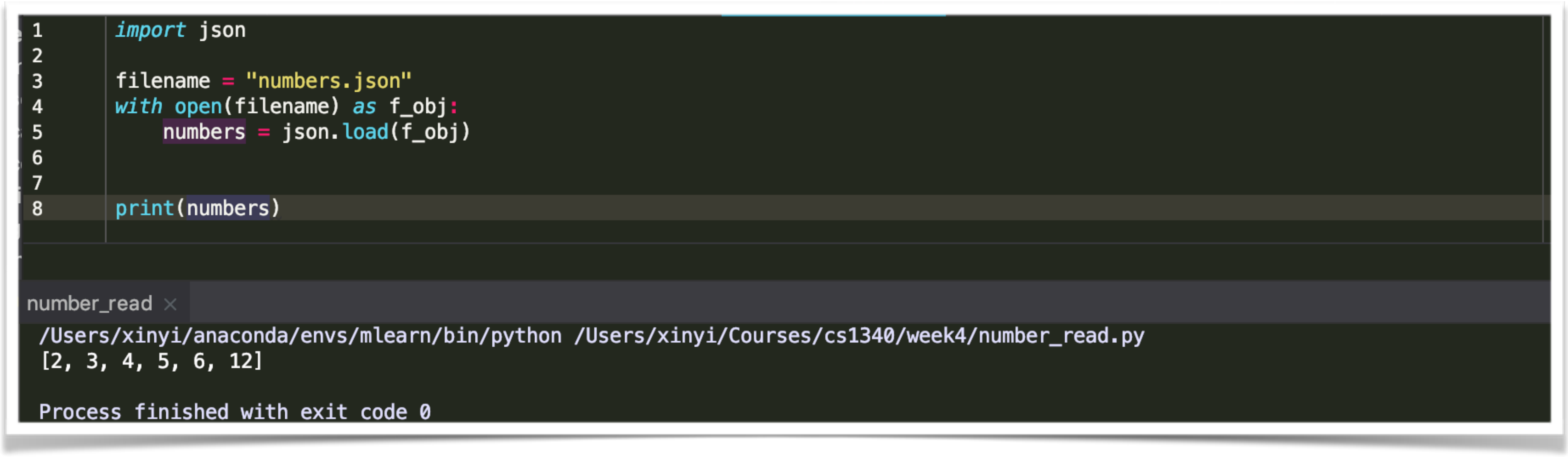
- Using json.dump() and json.load()



```python
import json

numbers = [2, 3, 4, 5, 6, 12]

filename = "numbers.json"
with open(filename, "w") as f_obj:
    json.dump(numbers, f_obj)

```

number_write ×

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/number_write.py

Process finished with exit code 0
```



```python
import json

filename = "numbers.json"
with open(filename) as f_obj:
    numbers = json.load(f_obj)


print(numbers)
```

number_read ×

```
/Users/xinyi/anaconda/envs/mlearn/bin/python /Users/xinyi/Courses/cs1340/week4/number_read.py
[2, 3, 4, 5, 6, 12]

Process finished with exit code 0
```