Programming for Data Science: File I/O

Instructor: Jakramate Bootkrajang

Based on material by Kittipitch Kuptavanich

Outlines

- Motivations
- Basic steps for manipulating file
  - Opening file
  - Reading from file or writing to file
  - Closing file
- try...finally

Motivations

- Data science usually involves large data that stored in file
  - DNA sequences
  - Text documents
  - Image
  - Data in CSV (comma separated values)
- Rarely do we input data using keyboard
- Being able to open and read file from files is essential to data analysis

Basic step for file manipulation

- Open file
  - We will get a file handler
- Read from or write to file via the file handler
- When finished, close file and destroy the file handler

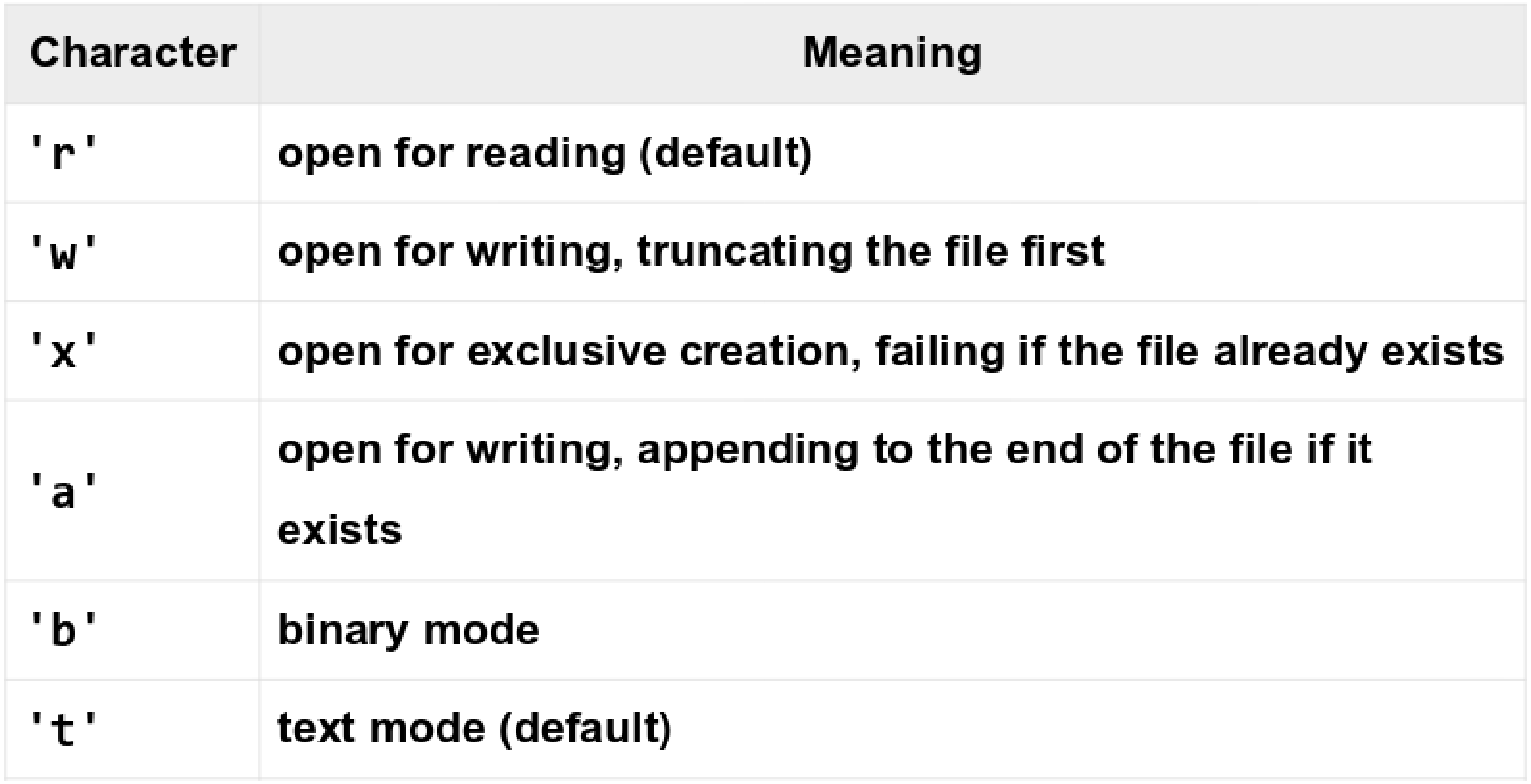
Opening file

- Use Python's built-in function

  ```
  open(filename, mode, encoding="utf-8")
  ```

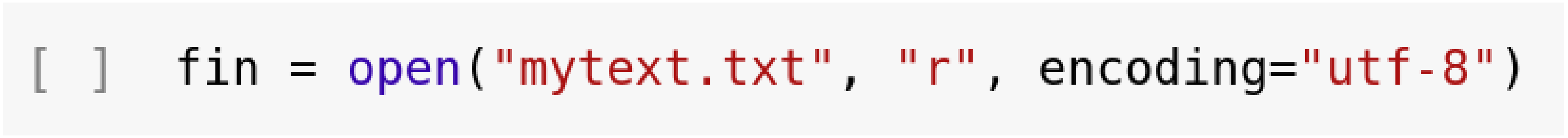
- filename is the name of file to be opened
- mode indicates whether to open for reading, writing or appending
- encoding is keyword argument for specifying character encoding mode.
  - UTF-8 is the new standard for displaying multi-languages characters

File operations modes



Character	Meaning
`'r'`	open for reading (default)
`'w'`	open for writing, truncating the file first
`'x'`	open for exclusive creation, failing if the file already exists
`'a'`	open for writing, appending to the end of the file if it exists
`'b'`	binary mode
`'t'`	text mode (default)

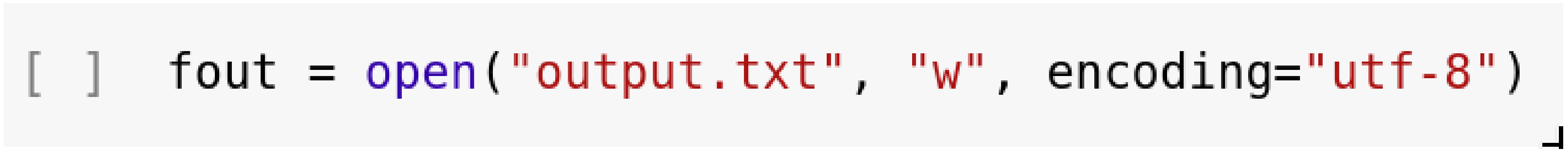
Example (Reading)



```
[ ]  fin = open("mytext.txt", "r", encoding="utf-8")
```

- open() returns a file handler to “mytext.txt” and we assigned it to fin
- Subsequence reading from “mytext.txt” can be done via fin
- fin is commonly used variable for file input
- Open returns **None** if file can’t be opened

Example (Writing)



```
[ ]  fout = open("output.txt", "w", encoding="utf-8")
```

- open() returns a file handler to “output.txt” and we assigned it to fout
- Subsequence writing to “output.txt” can be done via fout
- fout is commonly used variable for file output
- Open returns **None** if file can’t be opened

File Handler

- A channel by which reading and writing operation can be done
- Returned by open() function
- It is iterable
  - Meaning it can be iterated in for loop

```
fin = open("myfile.txt", "r")

for line in fin:    # iterate thru lines in myfile.txt
  print(line)
```

File operations

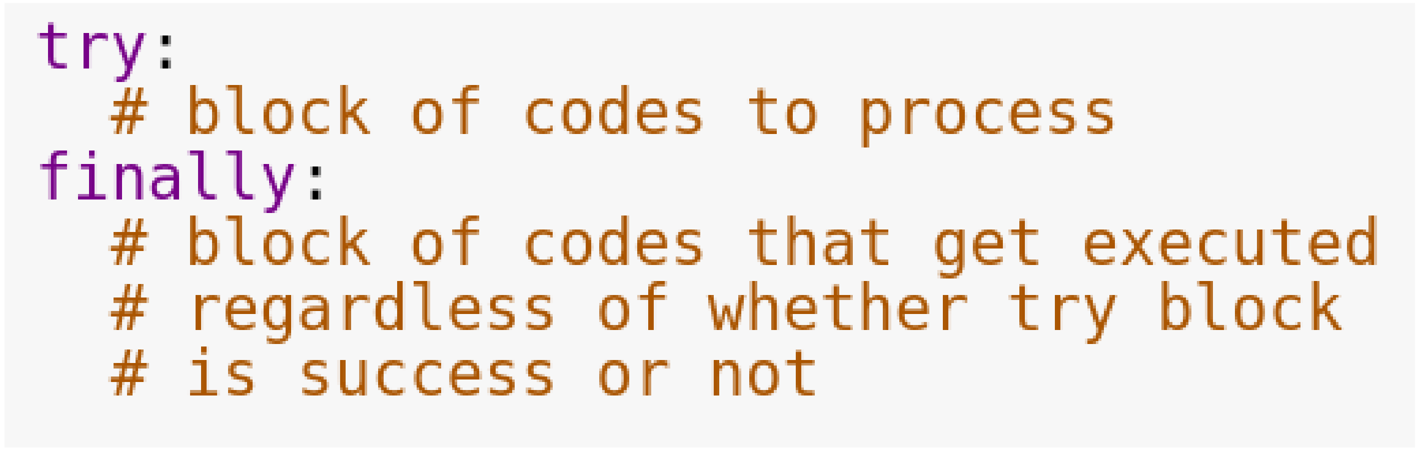
- Now that we have opened the file we can
- Read all content from file using
  - read()
- Read a line from file using
  - readline()
- Read all lines and get list of lines
  - readlines()
- Write some string to file using
  - write()

Closing file

- Opened file is vulnarable to modification
  - Intensionally or unintentionally
- It is advisable to always close file when file operations is finished.
- Closing file is done by a close() method
  - fin.close() # fin is our input file handler
  - fout.close()

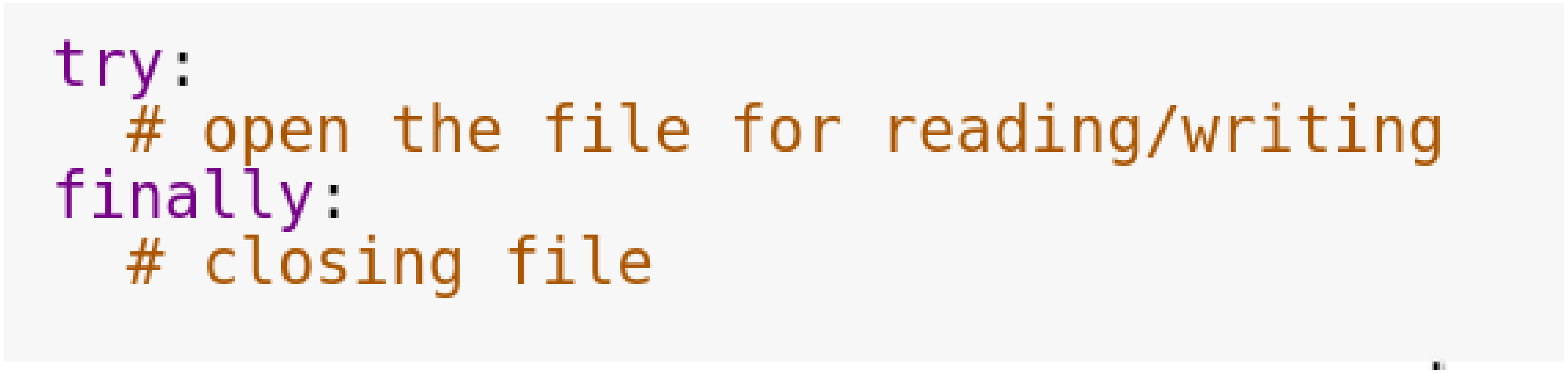
Try … finally

- Useful for working with file (ensure that file is always closed after use)



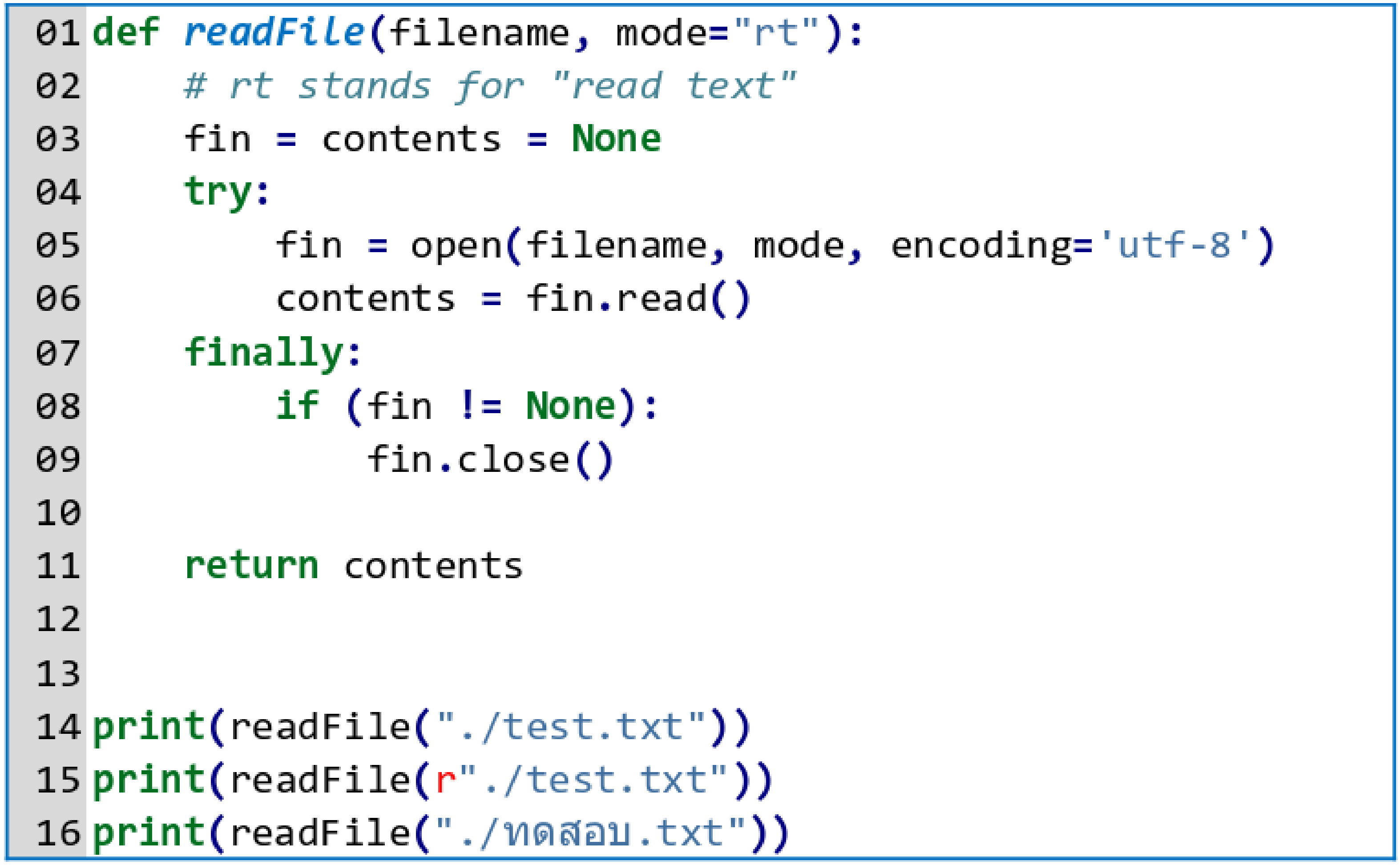
```
try:
  # block of codes to process
finally:
  # block of codes that get executed
  # regardless of whether try block
  # is success or not
```

Try...finally for file



```
try:
  # open the file for reading/writing
finally:
  # closing file
```

Another example for file reading



```python
def readFile(filename, mode="rt"):
    # rt stands for "read text"
    fin = contents = None
    try:
        fin = open(filename, mode, encoding='utf-8')
        contents = fin.read()
    finally:
        if (fin != None):
            fin.close()

    return contents


print(readFile("./test.txt"))
print(readFile(r"./test.txt"))
print(readFile("./ทดสอบ.txt"))
```

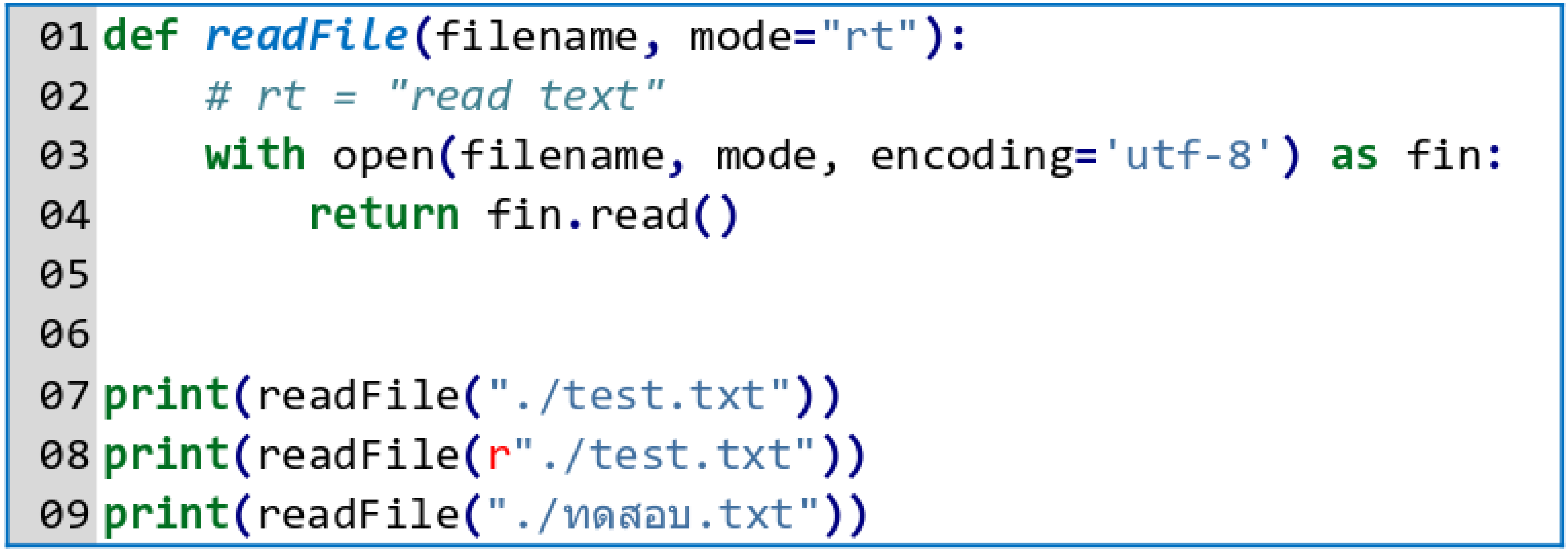
Yet another example

```
try:
  fin = open("myfile.txt", "r")
  for line in fin:    # iterate thru lines in myfile.txt
    print(line)
finally:
  fin.close()
```

Syntactic sugar for try...finally

- Syntactic sugar
  - Slang in programming language
- Instead of writing try...finally block
- We can write
  - with open(“myfile.txt”) as fin

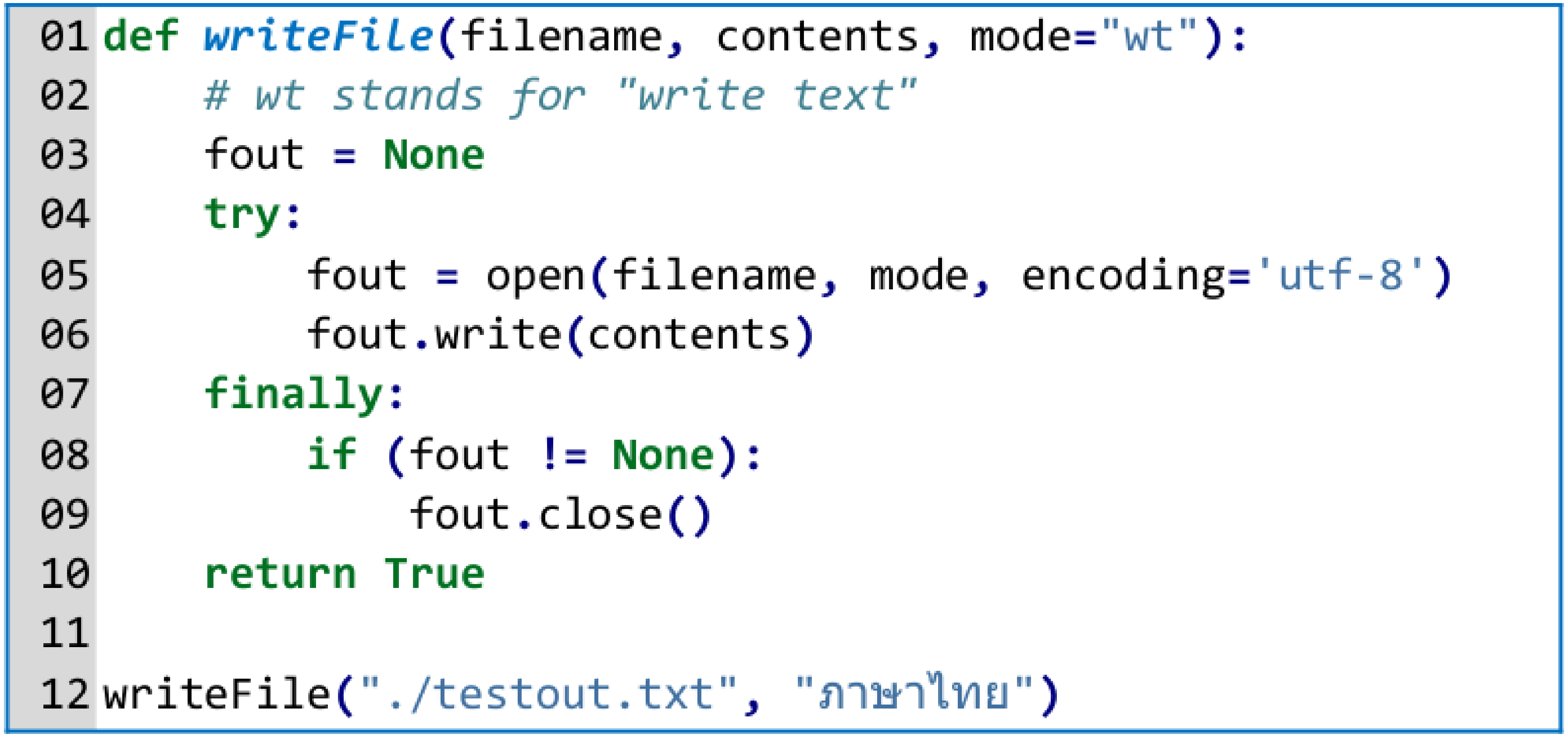
Example



```python
def readFile(filename, mode="rt"):
    # rt = "read text"
    with open(filename, mode, encoding='utf-8') as fin:
        return fin.read()


print(readFile("./test.txt"))
print(readFile(r"./test.txt"))
print(readFile("./ทดสอบ.txt"))
```

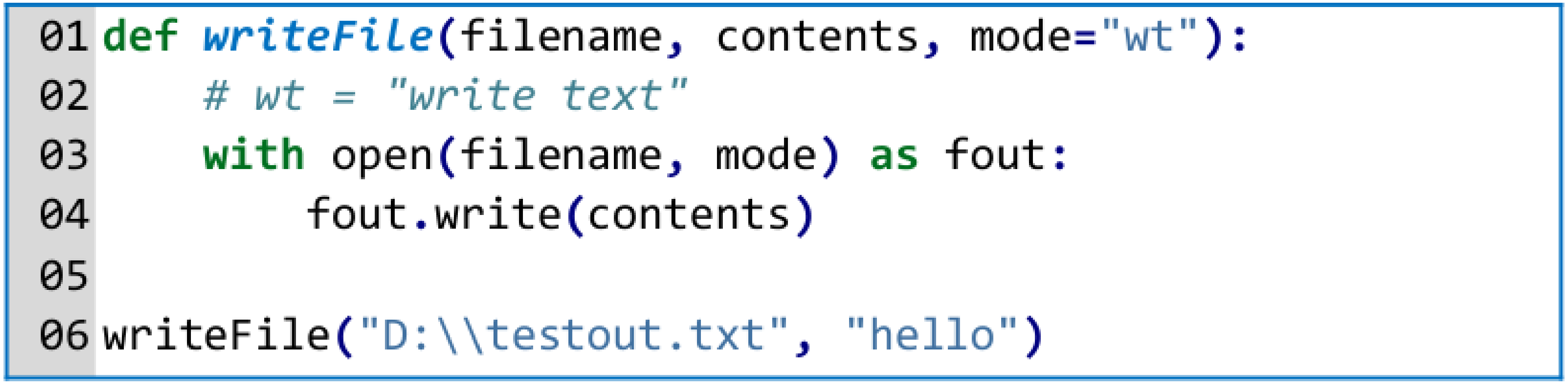
Writing to file



```python
def writeFile(filename, contents, mode="wt"):
    # wt stands for "write text"
    fout = None
    try:
        fout = open(filename, mode, encoding='utf-8')
        fout.write(contents)
    finally:
        if (fout != None):
            fout.close()
    return True

writeFile("./testout.txt", "ภาษาไทย")
```

Writing to file [2]



```python
def writeFile(filename, contents, mode="wt"):
    # wt = "write text"
    with open(filename, mode) as fout:
        fout.write(contents)

writeFile("D:\\testout.txt", "hello")
```