ADAM MICKIEWICZ UNIVERSITY IN POZNAŃ
Faculty of English

Robert Dyzman, M.Sc.Eng.

PYTHON PROGRAMMING
CLASS 08

Run „Teams”

Start your IDE

AGENDA:

- Create a file *„class_pp_08.py”*
- Pt_02.py solved
- Input/Output operations
- Writing to a file
- Reading from a file
- Exercise 160 („shakespeare.txt” – on Moodle (auxiliary…)
- Randint() => homework
- Push to GitHub, alternatively Copy/Paste to Teams

INPUT-OUTPUT

- Now we start to write programs that can respond to data entered by a user or from a file
- In this class, we treat the following ways of inputting data:

1. **Command line**, data can be entered on the command line when the program is invoked.
2. **Keyboard input**, a user can enter data when prompted by the program.
3. **File input-output**, a program can read data from or write data to files. We will focus on textual data.

COMMAND LINE

import sys

sys.argv → is a list that contains the command-line arguments passed to a Python script when it is executed.

```
import sys
sys.argv
##########################
## 01
##########################
import sys
L1 = sys.argv
print(L1)
# python class_pp_08.py this
# python class_pp_08.py this is my class
```

COMMAND LINE

import sys

sys.argv → is a list of all command-line arguments given when the program is invoked

```
import sys
sys.argv
##########################
## 01
##########################
import sys
L1 = sys.argv
print(L1)
# python pp_08.py this
# python pp_08.py this is my class
```

COMMAND LINE

```
##########
## 02, argument in the code
##########

vowels = 'aeiou'  # define vowels
word = 'Winnepesaukee'  # set word
# create vowel counter
vowelcount = 0
# go letter by letter
for letter in word:
    # is current letter a vowel?
    if letter in vowels:
        vowelcount += 1
print(f'There are {vowelcount} vowels in this word')
```

COMMAND LINE

```
##########
## 03, 1 argument read from cmd input
## python pp_class08.py Winnepesaukee
##########
import sys
vowels = 'aeiou'  # define vowels
word = sys.argv[1]  # set word
# create vowel counter
vowelcount = 0
# go letter by letter
for letter in word:
    # is current letter a vowel?
    if letter in vowels:
        vowelcount += 1
print(f'There are {vowelcount} vowels in this word')
```

COMMAND LINE

```
####################
# 04, handling more arguments from cmd
# python pp_class09.py my headphones are good
####################
import sys
vowels = 'aeiou'  # define vowels
words = sys.argv[1:]  # set list of words
# let's number the words
word_no = 0
# go letter by letter
for word in words:
    # create vowel counter
    vowelcount = 0
    word_no += 1
    for letter in word:
        # is current letter a vowel?
        if letter in vowels:
            vowelcount += 1
    print(f'There are {vowelcount} vowels in this word no {word_no}')
```

KEYBOARD INPUT

```
#####################################
# 05 KEYBOARD INPUT
#####################################

myText = input('Type in sth. .... ')
print(f'You have entered : {myText}')
```

FILE INPUT-OUTPUT

- IMPORTANT SAFEGUARDS

1. Do not experiment with important files,
2. When you do want to start working on your own files – make copies of them,

testfile.txt

OPENING A FILE, WRITING TO A FILE

1. Opening a file

open(file_path) → method returns file object

e.g.

```
outFile = open('testfile.txt', 'w’)
FileNotFoundError
```

Modes:

r → open a file for reading (*default*)

w → open a file for writing. Creates a new file if it doesn’t exist or truncates the file if it exists.

a → open for appending at the end of the file without truncating it. Creates a new file if it doesn’t exist.

t → open in text mode (*default*)

OPENING A FILE, WRITING TO A FILE

```
outFile.write('some text!\n')
outFile.write('...and some more text!\n’)
```

After carrying out all file operations, you have to close the file and free up the memory.

outFile.read()

outFile.write()

```
outFile.close()  # close file object
```

WRITING TO A FILE

```
# 2. Writting once more  to a file


# open / create file stream
outFile = open('testfile.txt', 'w')
# now it will overwrite a file
outFile.write('once more some text!\n')
outFile.write('...and once more some more
text!\n')
outFile.close()  # close file stream
```

APPENDING TO A FILE

With file access mode ‘a’, open() function first checks if file exists or not. If the file doesn’t exist, then it creates an empty file and opens it. Whereas, if the file already exists then it opens it. In both cases, **it returns a file object, and it has writing cursor, which points to the end of the opened file**.

Now, if you write anything to the file using this file object, then it will be appended to the end.

APPENDING TO A FILE

```
# 3. Writting to a file (appending)


# open / create file stream
outFile = open('testfile.txt', 'a')
# now it will append text to a file
outFile.write('once more some text!\n')
outFile.write('...and once more some more
text!\n')
outFile.close()  # close file stream
```

READING FROM A FILE

```
# 4. OPENING A FILE
# open file stream
inFile = open('testfile.txt', 'r')
stuff = inFile.read()  # read from it
inFile.close()  # close stream
print(stuff)  # print contents
```

READING FROM A FILE

```
# 5. OPENING A FILE (LINE BY LINE ANALYSIS)
# open file
inFile = open('testfile.txt', 'r')
stuff = inFile.read()  # read ALL file contents
inFile.close()  # close file
lines = stuff.split('\n')  # split into lines
# print lines and lengths
for line in lines:
    print(f'{line}, : {len(line)} characters')
```

READING FROM A FILE – readlines()

Definition and Usage

The `readlines()` method returns a list containing each line in the file as a list item.

Use the hint parameter to limit the number of lines returned. If the total number of bytes returned exceeds the specified number, no more lines are returned.

Syntax

```
file.readlines(hint)
```

Parameter Values

Parameter	Description
hint	Optional. If the number of bytes returned exceed the hint number, no more lines will be returned. Default value is -1, which means all lines will be returned.

https://www.w3schools.com/python/ref_file_readlines.asp

READING FROM A FILE – .readlines()

```
# 5.1 OPENING A FILE (LINE BY LINE ANALYSIS)
# open file
inFile = open('testfile.txt', 'r')
lines = inFile.readlines()  # read ALL file contents
inFile.close()  # close file
# print lines and lengths
for line in lines:
    print(f'{line}, : {len(line)} characters')
```

READING FROM A FILE – better method

```
# 6. OPENING A FILE (LINE BY LINE ANALYSIS) BETTER METHOD
# open file
with open('testfile.txt', 'r') as inFile:
    lines = inFile.readlines()  # read file contents
    # print lines and lengths
    for line in lines:
        print(f'{line}, : {len(line)} characters')
```

WRITING TO A FILE – better method

```
## 7. Writting to a file BETTER METHOD

# # open / create file stream
with open('testfile.txt', 'w') as outFile:
    # write to it
    outFile.write('some text!\n')
    outFile.write('...and some more text!\n')
```

randint() method (homework)

Syntax:

`random.randint(start, stop)`

Where:

start: (required) an integer specifying at which position to start

stop: (required) an integer specifying at which position to stop

```
import random

print(random.randint(3, 9))
```

https://www.w3schools.com/python/ref_random_randint.asp

Exercise 160

```
'''
Exercise 160, pp_160.py

Task is the same as in exercise 40 - finding the longest word in the given text, but
additionally you have to implement a function get_text().
The new function should return the text from a file,
Write a program with a function called map_longest() that takes a text as a parameter and
returns the longest word contained in that text and its length - tuple.

Result of a program should be a message
e.g. after punctuation removal
The longest word in the file 'shakespeare.txt' is 'internethartvmdcsouiucedu' with the length
of 25 characters

e.g. without punctuation removal
The longest word in the file 'shakespeare.txt' is '>internet:hart@.vmd.cso.uiuc.edu' with the
length of 32 characters

use map function together with lambda

Exception handling should be implemented.
Implement the possibility of entering file_path from command line
'''
```

Exercise 160

```
def get_text(file_path: str) -> str:
    '''
    Function opens the text file and returns its content

    Parameters:
        file_path: path to your text file
    Returns:
        text
    '''



def remove_punctuation(word: str) -> str:
    '''
    Function removes punctuation from the given string

    Parameters:
        word: any string with or without the punctuation
    Returns:
        string without punctuation
    '''
```

Exercise 160

```
def map_longest(text: str) -> tuple:
    '''
    Function returns the longest word in the text and it's length

    Parameters:
        text: any text

    Returns:
        tuple: a tuple containing the logest word and it's length

    ',’


# Rest of your code, including exception handling
```