Scripts and Modules

Exercises

Week 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

When a Python program is stored within a text file (i.e. a script), what suffix should be used

for the filename?

Answer: Python script filename is .py.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Is it necessary to use a special Integrated Development Environment (IDE) to write Python

code in text files?

Answer: No, it is not necessary to use a special IDE to write Python code in text files. Any text editor

can be used for writing Python code, and it can be saved with a .py extension.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

When a script is executed from a file, are the results of evaluating expressions automatically

displayed on the screen without the need of a print() function call?

Answer: Yes, in Python, when a script is executed from a file, the results of evaluating expressions are automatically

displayed on the screen.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

What command would need to be typed in an operating system terminal window in order to

execute a Python script called PrintNames.py?

Answer: Python PrintNames.py

What command would need to be typed in a terminal in order to pass the values "John",

"Eric", "Graham" as command line arguments to the PrintNames.py script?

Answer: Python PrintNames.py "John" "Eric" "Graham"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

When a Python script wishes to access command line arguments, what module needs to be

imported?

Answer: We can import sys

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

What is the data-type of the sys.argv variable?

Answer: The data-type of the sys.argv variable is a list of strings

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

What is stored within the first element of the sys.argv variable?

Answer: The first element of the sys.argv variable, namely sys.argv[0], always contains the script's

name.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Use a text editor to write the script called PrintNames.py. This should display any

command line arguments that were passed during execution.

Once complete, place your solution in the answer box below.

Answer:

import sys

if len(sys.argv) > 1:

print("Arguments passed:")

for i in range(1, len(sys.argv)):

print(f"Argument {i}: {sys.argv[i]}")

else:

print("No arguements passed.")

Improve the solution so it uses an if statement to check that at least one name was

passed, or otherwise print a message saying “no names provided”. Place your improved

solution in the answer box below.

Answer:

import sys

if len(sys.argv) > 1:

print("Arguments passed:")

for i in range(1, len(sys.argv)):

print(f"Argument {i}: {sys.argv[i]}")

print("\nNames provided:")

for i in range(1, len(sys.argv)):

print(f"{i}. {sys.argv[i]}")

else:

print("No arguments passed.")

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

When using an import statement it is possible to provide an alias that can be used as an

alternative name to access module content.

Write an import statement that imports the whole of the sys module, and renames it to

my\_system.

Answer: import sys as my\_system

Write a from..import statement that imports only the math.floor function, and renames it

to lower

Answer: From math import floor as lower

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

What is stored in a symbol-table?

Answer: Dictionary in Python, key-value pairs are stored.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Why is the following type of import statement generally not recommended?

from math import \*

Answer: Such imports can lead to conflicts and reduce readability.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

When working in interactive-mode what convenient function can be used to list all names

defined within a module?

Answer: dir() It lists all names defined within a module.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

What is the value stored within the sys.path variable used for?

Answer: The sys.path variable in Python is a list of strings that specifies the search path for

modules.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

When a program is being executed as a script what value is assigned to the special variable

\_\_name\_\_?

Answer: The value assigned to the special variable \_\_name\_\_ when a program is being executed as

a script is set to "\_\_main\_\_".

What value is assigned to the \_\_name\_\_ variable when a program has been imported as a

module?

Answer: When a program is imported as a module in Python, the value assigned to the special variable \_\_name\_\_

is set to the name of the module.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Why is it useful for a program to be able to detect whether it is running as a script, or

whether it has been imported as a module?

Answer: It helps in writing reusable code by allowing different behaviours when a script is run directly

or imported as a module.