Scripts and Modules

Exercises

Week 5

Prior to attempting these exercises ensure you have read the lecture notes and/or viewed the video, and followed the practical. You may wish to use the Python interpreter in interactive mode to help work out the solutions to some of the questions.

Download and store this document within your own filespace, so the contents can be edited. You will be able to refer to it during the test in Week 6.

Enter your answers directly into the highlighted boxes.

For more information about the module delivery, assessment and feedback please refer to the module within the MyBeckett portal.

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

Answer:

The filename of a Python program contained in a text file needs to end with ".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 Integrated Development Environment (IDE) to write Python code in text files. A basic text editor is adequate.

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:

No, a print() function call is required in order for the results of evaluating expressions to be seen on the screen when a script is run from a file. Python scripts require the explicit use of the print() function in order to display output.

What command would need to be typed in an operating system terminal window in order to execute a Python script called `PrintNames.py`?

Answer:

To run a Python script named PrintNames.py, type the 'python' command in an operating system terminal window.

What command would need to be typed in a terminal in order to pass the values "John", "Eric", and "Graham" as *command line arguments* to the `PrintNames.py` script?

Answer:

'python PrintNames.py John Eric Graham' needs to be typed in a terminal to pass the values "John", "Eric", and "Graham" as *command line arguments* to the `PrintNames.py` script

---

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

Answer:

A Python script must import the sys module in order to access command line arguments.

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

Answer:

The `sys.argv` variable is a list in Python.

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

Answer:

The first element of the `sys.argv` variable stores the name of the Python script being executed. It is typically the script's filename.

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

# Using list slicing to exclude the script name from command line
arguments arguments = sys.argv[1:]

# Printing each command line argument
for arg in arguments:
 print(arg)
```

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

# Using list slicing to exclude the script name from command
line arguments arguments = sys.argv[1:]

# Checking if at least one name was provided
```

```
if arguments:
 # Printing each command line argument
 for arg in arguments:
 print(arg)
else:
 print("No names provided")
```

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:

Variable names, data kinds, memory locations, and scope specifics are only a few of the facts that are kept about symbols in programs in a symbol table.

_______________ Why is the following type of import statement generally not recommended? `from math import *`

Answer:

from math import * is not advised since it can cause naming conflicts, namespace problems, and problems with the readability, maintainability, and clarity of the code.

When working in *interactive-mode* what convenient function can be used to list all names defined within a module?

Answer:

The dir() function can be used to list all names declared in a module in interactive mode.

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

Answer:

The Python search path for modules is stored in the sys.path variable. The interpreter uses it to find modules when making import statements.

When a program is being executed as a *script* what value is assigned to the special variable `__name__`?

Answer:

The special variable __name__ gets assigned the value "__main__" when a program is run as a script.

What value is assigned to the `__name__` variable when a program has been imported as a *module*?

Answer:

The name of the module (not "main") is assigned to the variable __name__ when a program is imported as a 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's helpful to know if a program is imported as a module or is operating as a script in order to manage the execution of particular code blocks..

_________ Exercises are complete

Save this logbook with your answers. Then ask your tutor to check your responses to each question.