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:

When a Python program is stored within a text file, **'.py'** suffix should be used for the filename.

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.

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, when a script is executed from a file, the results of evaluating expressions are not automatically displayed on the screen without the need for a print() function call.

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

Answer:

The command **'python PrintNames.py'** needs to be typed in an operating system terminal window to execute a Python script called `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:

The command **'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:

The **sys** module needs to be imported to access command line arguments.

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

Answer:

The data-type of the `sys.argv` variable is a **list.**

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

Answer:

The **name of the python script** that is being executed is stored within the first element of the sys.argv variable.

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

arguments = sys.argv[1:]
# Check if at least one name was provided
if arguments:
    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:

A symbol table stores information about symbols in a program, like variable names, data types, memory locations, and scope details.

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

```
from math import *
```

Answer:

from math import * is not recommended as it can lead to namespace issues, naming conflicts, etc.

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

Answer:

In interactive-mode **dir()** function can be used to list all names defined within a module.

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

Answer:

The sys.path variable is used for specifying the search path for modules.

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

Answer:

When a program is being executed as a script, the special variable __name__ is assigned the value "__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, the variable __name__ is assigned 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:

Detecting whether a program is running as a script or has been imported as a module is useful for controlling the execution of specific code blocks.

Exercises are complete

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