1811/2807/7001ICT
Programming Principles

School of Information and Communication Technology
Griffith University

Trimester 1, 2024

26 Exploring Modules

The success of a programming language is not measured by its intrinsic qualities. The successful languages are used by more people for more problems.

Python is a *very* widely used and popular programming language. As a consequence there are very many useful libraries and extensions.

The next course, Software Technologies, explores many of these. This is just a taste.

26.1 Unix/Linux commands

In this course we have used the command line in the Windows Command Prompt or the MacOS/Linux Terminal apps.

So far, to run a Python program from the command line, we have had to invoke the Python interpreter explicitly. For example:

```
# script: program1.py
# A program

print("Hi!")
```

```
$ python3 program1.py
Hi!
$
```

Unix (and therefore Linux) has a system feature for scripting/interpreted languages, where the script can say what interpreter should be used, the "magic line" or the "Shebang line".

If the first line starts with #!, the rest of the line is the path to the interpreter that should be used to run this program.

Then it is not necessary to add the .py extension to the script file name, or invoke the interpreter explicitly.

```
#!/usr/local/bin/python3

# script: program2
# A program that runs as a native command.

print("Hi!")
```

```
$ ./program2
Hi!
$
```

The `./` is to say run the command that is a file in the current directory.

It will not be needed if the script is saved in a directory that is normally searched for commands.

26.2 Command line arguments, module sys

Command line arguments are extra words typed on the command line that invokes a program.

For example, the Python interpreter accepts an argument, the file to interpret.

```
$ python3 program1.py
Hi!
$
```

If you want to access command line arguments typed by the user of your program, the module `sys`, gives you access.

```
#!/usr/local/bin/python3

# script: program3
# A program that runs as a native command,
# using a command line argument.

import sys

print("Hi {}!".format(sys.argv[1]))
```

```
$ ./program3 Jun
Hi Jun!
$
```

26.3 Running other commands, module os

Rule Zero of programming: “If there is already a program that does the job, you don’t need to write a program.”

Reuse of existing software is the cheapest way to get a job done.

Sometimes we can use a function from a module.

Sometimes we can use a whole program to solve part of a problem.

A Python program can run another program on Windows or Unix/Linux using the function os.system.

```
#!/usr/local/bin/python3

# script: HiJohn
# This program says hi to John, using my other program
# that knows how to say Hi to people.

import os

os.system("./program3 John")
```

```
$ ./HiJohn
Hi John!
$
```

So Python may be used as an alternative to other scripting languages, usually the “shell” languages like `bash`.

26.4 C interface – data science, etc

Python is a popular language for “data science” or Artificial Intelligence applications.

The core of these applications is a strategy using artificial neural networks.

Neural networks emulate how biological neuron cells, as in your brain, work.

A neural network simulated in a computer is a numerical model that is “trained” with inputs (say an image) and outputs (what the image is).

The training is quite computationally intensive (*i.e.* slow).

If this code was written in Python (an interpreted language), it would be unacceptably slow.

So the computationally intensive parts are implemented in a lower-level, compiled language, such as C.

Python is an easy language to learn, and very good for text processing, so very good for writing the input and output parts of a program, even if the hard part, say machine learning, is built in C or another language.

Python provides modules to interface to code written in C.

Most modern languages do this.

C is the language that most operating systems are written in.

So Python can interface with code written in many languages, as the other languages are also compatible with C, the *lingua franca*.

26.5 Django – web development

Django is a framework, implemented in Python for developing web sites, particularly those with content managed by a database.

So Python/Django is a popular altenative to PHP.

Section summary

This section covered:

- some examples from the Python standard library that extend its utility.