Chapter 1

System Tools

Lesson 1

sys – tools

Make a directory called Advanced-Python-Programming.

Load Visual Studio and add a bash power shell. This will simulate a linux terminal.

Create a sub-directory called System-Tools using the following command

mkdir System-Tools

mkdir is short for "make directory".

Practice:

Create another sub directory called temp, then remove the directory using the rmdir <name> command.

Navigate to your new directory using the command cd System-Tools. cd stands for "change directory"

Practice:

Navigate back up a level using the cd .. command. Then, navigate back to System-Tools.

The wild card symbol (sometimes called a "glob") is \*. You can use it to save tying.

Try the following:

cd ..

cd Sys\*

Load sys\_tools.py.

Using Visual Studio, add a bash power shell.

Run the script using python3 sys\_tools.py

Lesson 2

os – tools

Load and run osenviron.py

This program prints the current working directory.

Using the bash powershell run the program ospath.py from within the System-Tools subdirectory.

What is the current working directory?

Navigate using bash commands up one directory and run the script using

python3 ./System-Tools/ospath.py

What is the current working directory?

Did you notice that when using our simulation of Linux in the Visual Studio Code's terminal that we use the / (forward slash) symbol as a directory separator, but when the cwd is printed the \ (back slash) symbol is used (if you are using a windows machine)?

Different platforms using different shell commands and different seperators. Without Python, we would have to write different files for running scripts in windows (called Batch files) and for Linux (called bash files). With Python we can make our scripts platform independent. All a computer needs is Python installed.