Python
Setup Guide

During the course we will be writing and debugging a series of Python programs. To do that we need an editor and a debugger. On Unix there are many editors to choose from, vi, vim, emacs, pico etc and from the early days of Unix there have been a number of debuggers available. In the mid 80s gdb was modelled after dbx (an earlier debugger) and is now the principle debugger used in Unix systems. Python uses a slightly modified version of gdb called pdb.

Integrated Development Environments or IDEs combine editors and debuggers into a single package. IDEs often provide additional facilities, but the editor and debugger remain the principle components. Since 2000, the Eclipse IDE has been very popular on Unix systems. Eclipse employs a series of plug-ins to provide support for various languages (C, C++, Perl, Pyton, Rust etc). The Python plug-in is called PyDev and so Eclipse/PyDev was the IDE of choice for a number of years.

However, recently JetBrains released the excellent IDE, PyCharm. Many developers prefererred PyCharm to Eclipse and things changed again when Microsoft released Visual Studio Code or vscode as its normally called (to avoid confusion with the Windows only IDE also called Visual Studio). All of these IDEs work cross platform on Windows, MacOS and Linux.

As of now (May 2021), vscode has become the most popular IDE on Unix. The editor in the IDE is basically the same as its Windows counterpart, but on Linux, vscode uses gdb/pdb as a debugger. The IDE integrates seamlessly with Anaconda Python (the Python version used at Diamond).

My prefered opition (and the one I'll be using during the course) is Microsoft's VSCODE.

1. Downloading Examples

All the examples for the course are stored on github. You can clone the repository with:

```
git clone https://github.com/seddon-software/python-course.git
```

2. Module Loads

Module loads are the prefered way of setting up applications at Diamond. Note that all module loads only apply to the current terminal window. If you create a new window you will need to type these commands in again (unless you add the commands to your .bashrc file).

3. Setting up Python and VSCode

Assuming you're working on a Diamond machine, you'll need to configure the latest version of Python (currently 3.9):

```
module load python/3.9
```

Make sure you don't have any other versions of Python loaded - unload as necessary. For example:

```
module unload python/ana
```

To load VSCode:

```
module load vscode/1.56.0
```

You can replace the versions of Python and VScode with later versions if they are available. To see what's available:

```
module avail python
module avail vscode
```

After you've downloaded the examples from github, you can run vscode by:

```
cd python-course
code .
```

Note the dot in the last command (run ***code*** in the current directory).

You should then go to the Extensions icon and add the Python extension module (it may already be installed).

4. Using Jupyter Notebook

An alternative approach is to use Jupyter Notebook; the notebook is automatically installed with Python (Anaconda). The Notebook is a browser version of command line Python and is straightforward to use. Start the Notebook from the command line with:

```
cd python-course
jupyter notebook
```

You can then enter and run Python code in the cells created. There are several tutorials on-line to explain how to use the notebook.

5. Using PyCharm on a Diamond machine

If you prefer to use PyCharm you will need to use the community edition (the default at Diamond) unless you have a commercial licence:

```
module load pycharm
```

6. Using Eclipse/PyDev on a Diamond machine

Eclipse has been around for a long time and can be used for many programming languages. To work with Eclipse/Python you need the PyDev plugin. You use module loads to setup Eclipse and this will also install the PyDev plugin. The following module loads are required:

```
module load python/3.9
module load eclipse
```

Make sure you don’t have two versions of eclipse installed. To see what is loaded type:

```
module list
```

If you see “eclipse/473a(default)”, ir something similar, you will need to unload it:

```
module unload eclipse/473a
```

You can now run Eclipse with:

```
eclipse&
```

Once Eclipse starts you will need to specify a folder for your workspace and then switch to the PyDev prospective (using the Window menu). You can now create a PyDev project (from the File menu). When you are asked to configure the Python interpreter choose "Quick Auto Configure). You should now be able to create projects and run code. Start by importing the example from github.