

**Course** : Diploma in Electronics & Computer Engineering (EGDF20)

**Module**  : Connected System Design Project (EGE205)

**Laboratory No**. : Lab 1b

**Laboratory Title** : Introduction: Getting Started with BeagleBone Black Wireless (BBBW) Board

**Objective** : To get to know more on Linux shell command and write some intermediate

python program.

**Hardware**  : BBBW Board with USB Cable x1

**Contents**

1. Working More on Linux Shell Command
   1. Intermediate Linux Shell Command
2. Writing Python Program in the (BBBW) Board
   1. Intermediate Python Programming

# **Working More on Linux Shell Command**

## Intermediate Linux Shell Command

1. **Log in** to the BBBW Board through SSH using the default username “**debian**” and password “**temppwd**”.
2. **Type** in the command “**pwd**” and **hit** the “Enter” key. It returns the information of the current working directory as shown in the Figure below.



Figure 1.1a: The “pwd” Command

1. **Type** in the command “**ls**” and **hit** the “Enter” key. It returns the information of the folders’ and files’ name located in the current working directory as shown in the Figure below.



Figure 1.1b: The “ls” Command

1. **Type** in the command “**touch new.txt**” and **hit** the “Enter” key. **Type** in the command “**ls**” and **hit** the “Enter” key again. The command “**touch**” creates a file with a name. The name of the file must be typed in followed by the command as shown in the Figure below.



Figure 1.1c: The “touch” Command

1. **Type** in the command “**man touch**” and **hit** the “Enter” key. The command “**man**” shows the manual pages of the command “**touch**”. **Hit** the “**q**” key to exit the manual page as shown in the Figure below

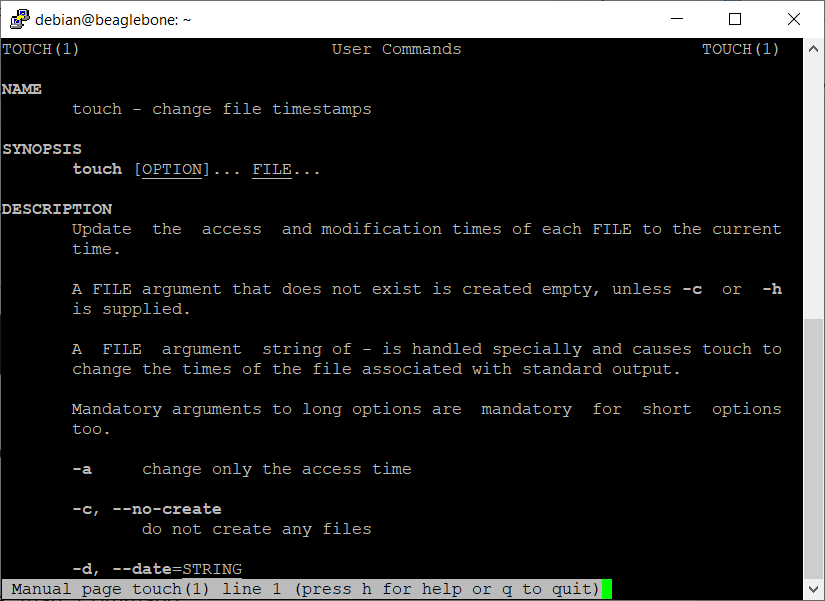


Figure 1.1d: The “man” Command

1. **Type** in the command “**touch --help**” and **hit** the “Enter” key. The command argument “**--help**” shows show which ways the command can be used as shown in the Figure below.

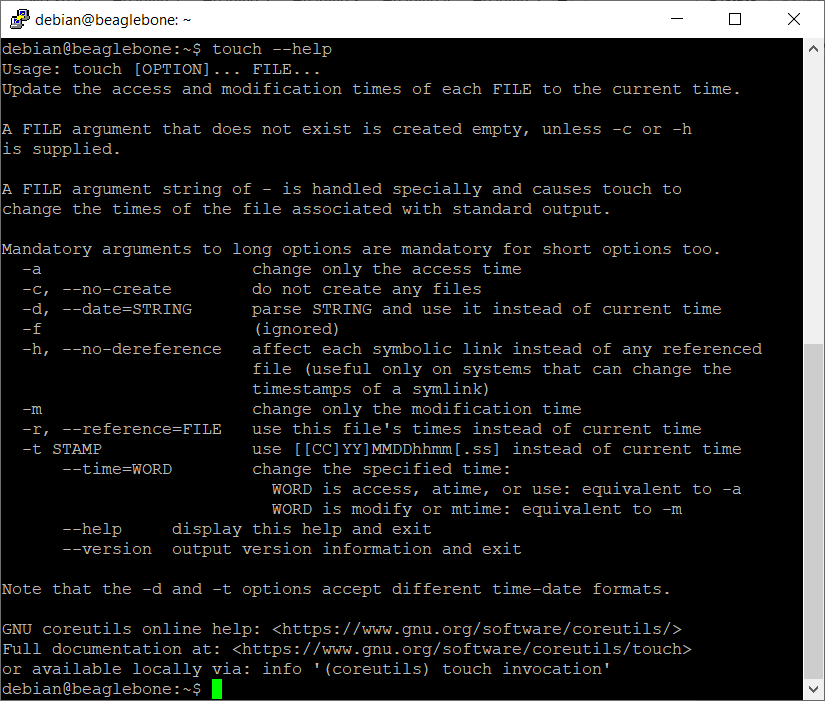


Figure 1.1e: The “--help” Command Argument

1. **Type** in the command “**cp new.txt /home/debian/bin**” and **hit** the “Enter” key. The command “**cp**” copy files through the command line. It takes two arguments where the first is the location of the file to be copied, the second is where to copy to. **Type** in the command “**cd bin**” and **hit** the “Enter” key to access the bin folder. Then, **Type** in the command “**ls**” and **hit** the “Enter” key. It is observed that the new.txt has been copied into bin folder as shown in the Figure below.

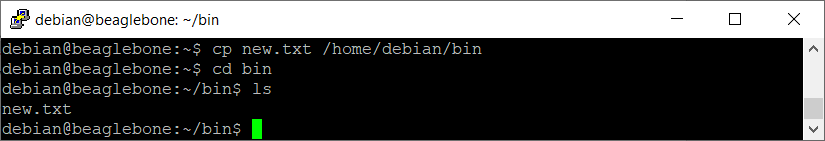


Figure 1.1f: The “cp” Command

1. **Type** in the command “**mv new.txt old.txt**” and **hit** the “Enter” key. The command “**mv**” renames the file “new.txt” to “old.txt”. **Type** in the command “**ls**” and **hit** the “Enter” key. It is observed that the new.txt has been renamed to old.txt as shown in the Figure below.

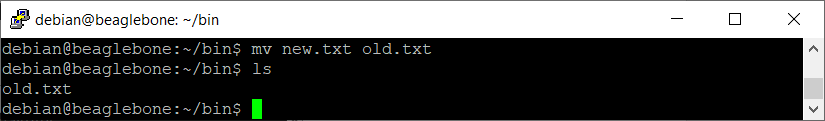


Figure 1.1g: The “mv” Command

1. **Type** in the command “**mv old.txt /home/debian**” and **hit** the “Enter” key. The command “**mv**” moves the file “old.txt” to /home/debian directory. **Type** in the command “**cd ..**” and **hit** the “Enter” key to go to /home/debian directory. **Type** in the command “**ls**” and **hit** the “Enter” key. It is observed that the old.txt has been moved to /home/debian directory as shown in the Figure below.

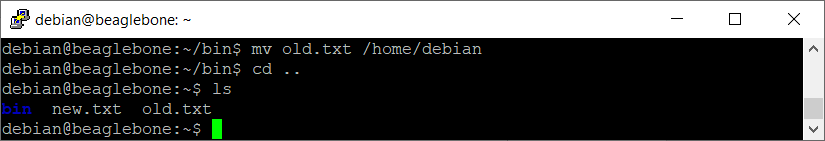


Figure 1.1h: The “mv” Command

1. **Type** in the command “**echo Hello, NYP! >> new.txt**” and **hit** the “Enter” key. The command “**echo**” moves some data, usually text into a file. **Type** in the command “**cat new.txt**” and **hit** the “Enter” key. “**cat**” command is used to display the contents of a file. It is observed that the text “**Hello, NYP!**” has been entered into the new.txt file as shown in the Figure below.

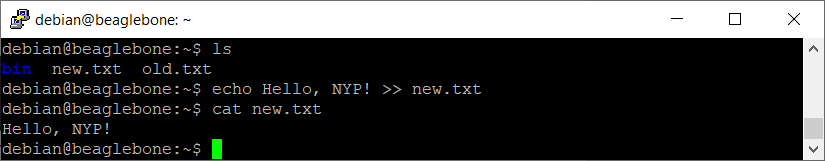


Figure 1.1i: The “echo” & “cat” Command

1. **Type** in the command “**ps -aux**” and **hit** the “Enter” key. The command “**ps**” lists the processes running on your computer as shown in the Figure below.

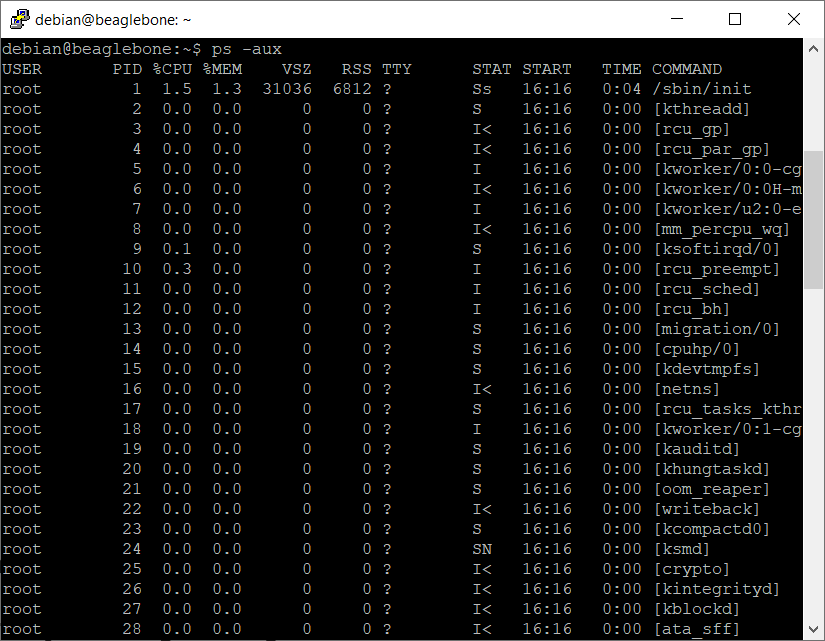


Figure 1.1j: The “ps” Command

1. **Type** in the command “**sudo** **kill xxxx**” and **hit** the “Enter” key (replace **xxxx** with the PID number found in the generated by the “**ps**” command, this example uses PID number of 1786). **Type** in the password “**temppwd**” and **hit** the “Enter” key again. The command “**kill**” is used to end a running task as shown in the Figure below.



Figure 1.1k: The “kill” Command

1. **Type** in the command “**ps -aux**” and **hit** the “Enter” key. It is observed that the process ID of 1786 is no longer listed as shown in the Figure below.

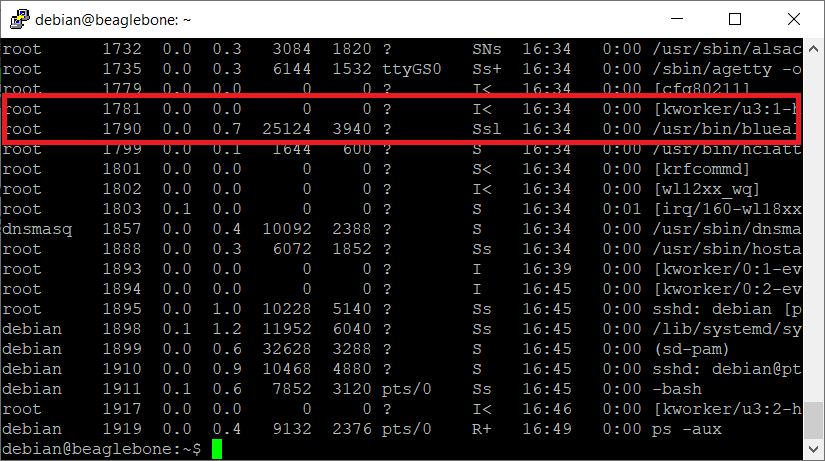


Figure 1.1l: The “ps” Command

1. Congratulations! You have gotten to know even more Linux Shell commands now!

# **Writing the Python Program in the (BBBW) Board**

## Intermediate Python Programming

**Set & Dictionary**

**Sets** are used to store multiple items within a **curly** bracket in a single variable. Set items are **unordered**, **unchangeable**, and **do not allow duplicate** values.

**Dictionaries** are used to data value in key : value pairs within a **curly** bracket in a single variable. Dictionary items are **ordered**, **changeable**, and **does not allow duplicates**.

**Unordered** means that the items in a set do not have a defined order. Set items can appear in a different order every time you use them and cannot be referred to by index or key.

1. **Enter** the following code into the Cloud9 IDE.

|  |
| --- |
| myset = {"Alex", "David", "Peter"}  mydictionary = {"Name1": "Alex", "Name2": "David", "Name3": " Peter"}  mydictionary.update({"Name1": "John"})  print(mydictionary)  myset[1] = "John" |

1. **Click** on the “Run” button to execute the code.
2. **Observe** and **Compare** the program output at the console output window with your teammates and consult your lecturer for advice if it is not the same.
3. **Write** the program output in the white box below for future reference if needed.

|  |
| --- |
| *Right click and select “New comment” to insert your program as a comment.* |

**Module**

A **module** allows you to logically organize your Python code. Grouping related code into a module makes the code easier to understand and use. A module is a Python object with arbitrarily named attributes that you can bind and reference.

Simply, a module is a file consisting of Python code. A module can define functions, classes, and variables. A module can also include runnable code.

1. **Enter** the following code into the Cloud9 IDE and save the file as **mymodule.py.**

|  |
| --- |
| StudentInfo = {  "name": "John",  "age": 36,  "country": "Singapore"  }  def Greeting(Name):  print("Hello, " + Name) |

1. **Enter** the following code into the Cloud9 IDE and save the file as **mainfile.py**

|  |
| --- |
| import mymodule  StudentName = mymodule.StudentInfo["name"]  mymodule.Greeting(StudentName) |

1. **Click** on the “Run” button to execute the code.
2. **Observe** and **Compare** the program output at the console output window with your teammates and consult your lecturer for advice if it is not the same.
3. **Write** the program output in the white box below for future reference if needed.

|  |
| --- |
| *Right click and select “New comment” to insert your program as a comment.* |

**File IO**

Python has several functions for **creating**, **updating**, **reading**, and **deleting** files.

The key function for working with files in Python is the **open()** function. The open() function takes two parameters: filename, and mode. There are four different methods (modes) for opening a file: read(r), append(a), write(w) and create(x), In addition you can specify if the file should be handled as binary(b) or text mode(t).

To create a new file, use the **open()** method, with either create(x) or write(w) parameter.

To append content to a file, use the **open()** method, with append(a) parameter.

To read the content of a file, use the **read()** method of the file object returns from the open() function.

To close a file when you are done with it, use the **close()** method of the file object returns from the open() function.

To delete a file, use the import the OS module, and run its **os.remove()** function:

1. **Enter** the following code into the Cloud9 IDE.

|  |
| --- |
| import os  File = open("MyNotes.txt", "w")  File.write("If you can see me means I have written to MyNotes successfully!")  File.close()  File = open("MyNotes.txt", "r")  print(File.read())  File.close()  File = open("MyNotes.txt", "a")  File.write("I am adding more lines!")  File.close()  File = open("MyNotes.txt", "r")  print(File.read())  File.close()  os.remove("MyNotes.txt") |

1. **Click** on the “Run” button to execute the code.
2. **Observe** and **Compare** the program output at the console output window with your teammates and consult your lecturer for advice if it is not the same.
3. **Write** the program output in the white box below for future reference if needed.

|  |
| --- |
| *Right click and select “New comment” to insert your program as a comment.* |

**Exception Handling**

An **exception** is an event, which occurs during the execution of a program that disrupts the normal flow of the program's instructions. In general, when a Python script encounters a situation that it cannot cope with, it raises an exception. An exception is a Python object that represents an error.

When a Python script raises an exception, it must either handle the exception immediately otherwise it terminates and quits.

If you have some suspicious code that may raise an exception, you can defend your program by placing the suspicious code in a **try**: block. After the try: block, include an **except**: statement, followed by a block of code which handles the problem as elegantly as possible.

1. **Enter** the following code into the Cloud9 IDE.

|  |
| --- |
| try:  File = open("demofile.txt", "r")  File.write("How are you?")  except:  print("Something went wrong when writing to the file") |

1. **Click** on the “Run” button to execute the code.
2. **Observe** and **Compare** the program output at the console output window with your teammates and consult your lecturer for advice if it is not the same.
3. **Write** the program output in the white box below for future reference if needed.

|  |
| --- |
| *Right click and select “New comment” to insert your program as a comment.* |

**Class and Object**

**Class** is a user-defined prototype for an object that defines a set of attributes that characterize any object of the class. The attributes are data members (class variables and instance variables) and methods, accessed via dot notation.

**Object** is a unique instance of a data structure that's defined by its class. An object comprises both data members (class variables and instance variables) and methods.

To create a class, use the keyword **class**.All classes have a function called **\_\_init\_\_()**, which is always executed when the class is being initiated. The \_\_init\_\_() function is used to assign values to object properties, or other operations that are necessary to do when the object is being created.

**Objects** can also contain **methods**. Methods in objects are functions that belong to the object.

The **self** parameter is a reference to the current instance of the class and is used to access variables that belongs to the class.

1. **Enter** the following code into the Cloud9 IDE.

|  |
| --- |
| class Student:  def \_\_init\_\_(self, Name, Age):  self.Name = Name  self.Age = Age  def SelfIntro(self):  print("Hello my name is " + self.Name)  Student1 = Student("John", 18)  print(Student1.Name)  print(Student1.Age)  Student1.SelfIntro() |

1. **Click** on the “Run” button to execute the code.
2. **Observe** and **Compare** the program output at the console output window with your teammates and consult your lecturer for advice if it is not the same.
3. **Write** the program output in the white box below for future reference if needed.

|  |
| --- |
| *Right click and select “New comment” to insert your program as a comment.* |

*Congratulations! You have successfully completed the Lab1b. Good job! Take a good break and stay tune for next lab. More exciting lab exercises coming to you!*