

Name: Vaibhav Gorakh Lonkar

PRN: 21410027

Batch: EN2

RTOS Experiment No.6

Title: Execution of Python file in Linux.

Part A: Execution of Single Python file (Python code includes arithmetic operation).

Procedure:

1. Create a Python file.

You can create a Python file with any text editor like nano. For instance, create a file arithmetic.py that includes a simple arithmetic operation.

2. In the file, write the Python code, Save and exit.

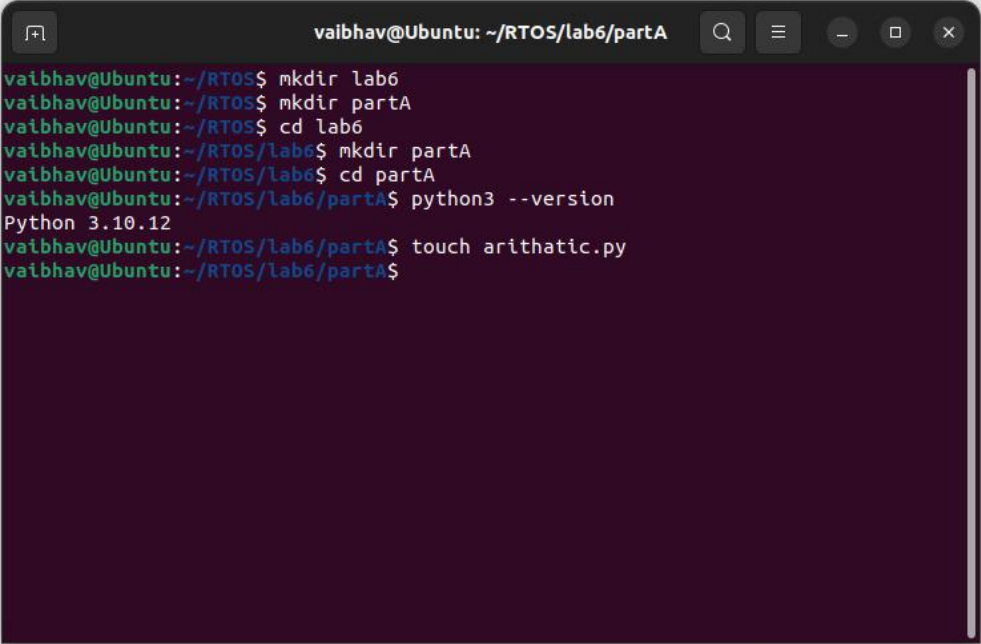
3. Execute the Python file.

4. Result- The output will be displayed.

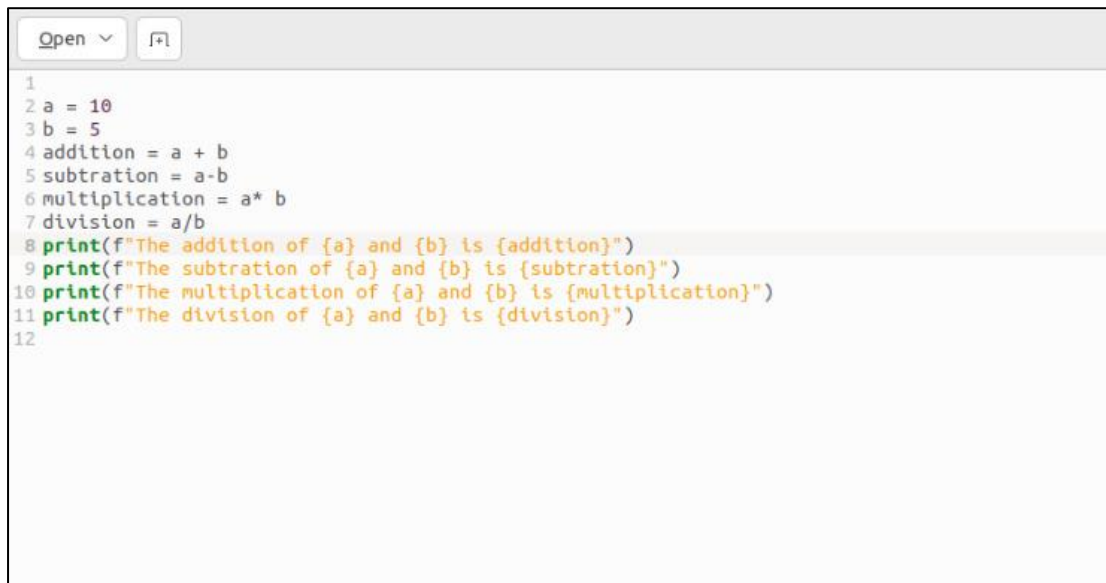
Commands used:

- ☐ **touch arithmetic.py** — To create the Python file.
- ☐ **python3 arithmetic.py** — To execute the Python file.

Result: Screen Shots

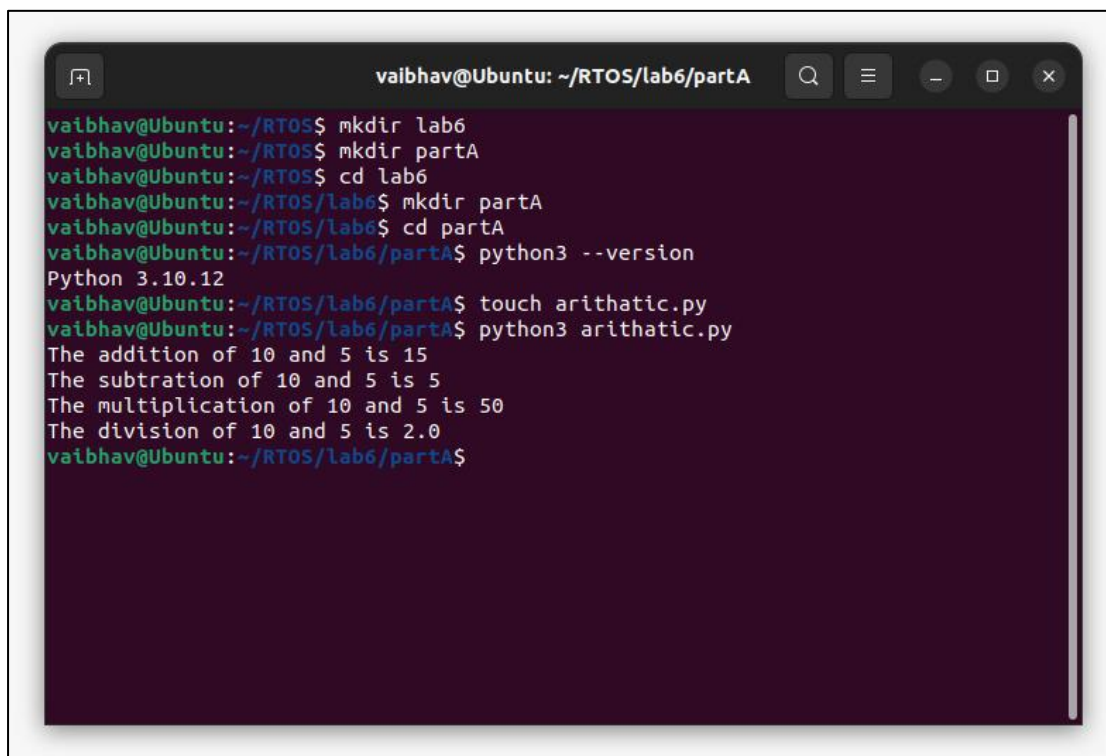


```
vaibhav@Ubuntu: ~/RTOS/lab6/partA
vaibhav@Ubuntu:~/RTOS$ mkdir lab6
vaibhav@Ubuntu:~/RTOS$ mkdir partA
vaibhav@Ubuntu:~/RTOS$ cd lab6
vaibhav@Ubuntu:~/RTOS/lab6$ mkdir partA
vaibhav@Ubuntu:~/RTOS/lab6$ cd partA
vaibhav@Ubuntu:~/RTOS/lab6/partA$ python3 --version
Python 3.10.12
vaibhav@Ubuntu:~/RTOS/lab6/partA$ touch arithatic.py
vaibhav@Ubuntu:~/RTOS/lab6/partA$
```



A screenshot of a code editor window. The window has a title bar with an 'Open' button and a file icon. The code is as follows:

```
1
2 a = 10
3 b = 5
4 addition = a + b
5 subtration = a-b
6 multiplication = a* b
7 division = a/b
8 print(f"The addition of {a} and {b} is {addition}")
9 print(f"The subtration of {a} and {b} is {subtration}")
10 print(f"The multiplication of {a} and {b} is {multiplication}")
11 print(f"The division of {a} and {b} is {division}")
12
```



A screenshot of a terminal window titled 'vaibhav@Ubuntu: ~/RTOS/lab6/partA'. The terminal shows the following commands and output:

```
vaibhav@Ubuntu:~/RTOS$ mkdir lab6
vaibhav@Ubuntu:~/RTOS$ mkdir partA
vaibhav@Ubuntu:~/RTOS$ cd lab6
vaibhav@Ubuntu:~/RTOS/lab6$ mkdir partA
vaibhav@Ubuntu:~/RTOS/lab6$ cd partA
vaibhav@Ubuntu:~/RTOS/lab6/partA$ python3 --version
Python 3.10.12
vaibhav@Ubuntu:~/RTOS/lab6/partA$ touch arithatic.py
vaibhav@Ubuntu:~/RTOS/lab6/partA$ python3 arithatic.py
The addition of 10 and 5 is 15
The subtration of 10 and 5 is 5
The multiplication of 10 and 5 is 50
The division of 10 and 5 is 2.0
vaibhav@Ubuntu:~/RTOS/lab6/partA$
```

Part B: Execution of Multiple Python files : 1. using shell script

2. using && operation

(create 3 to 4 files having different python code like loops, arithmetic operation etc.)

1. Using shell script

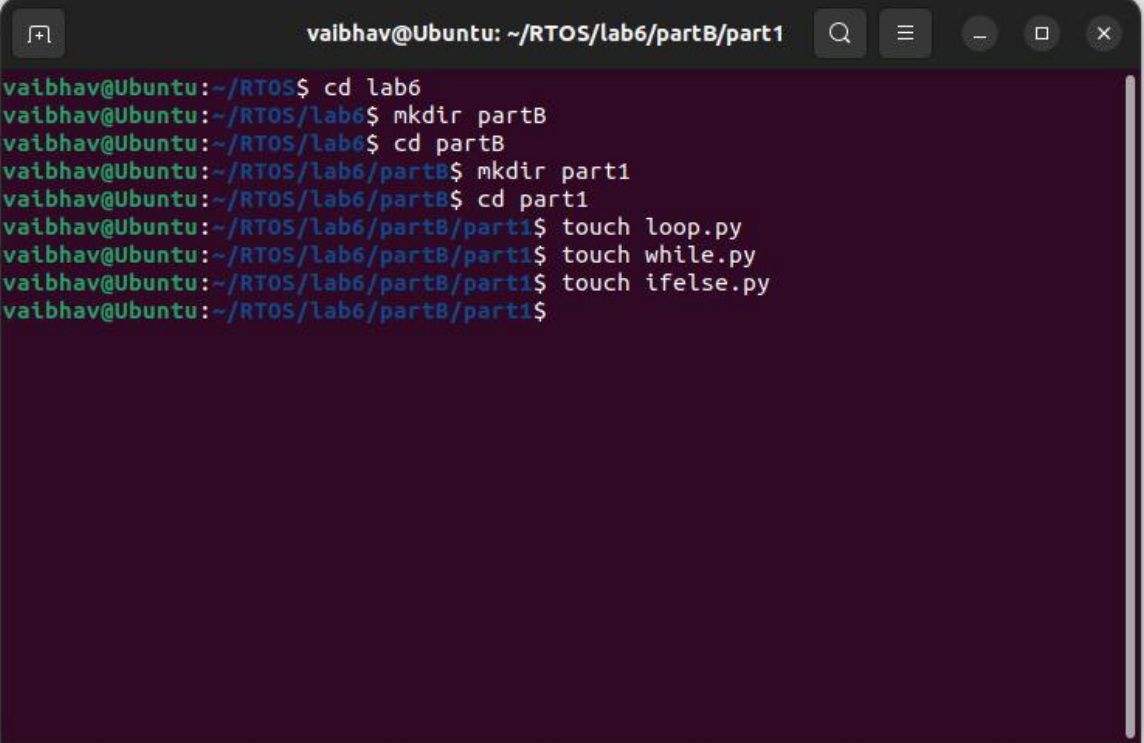
Procedure:

- 1.Create Multiple Python Files-Creat 3 Python files.
- 2.Create a Shell Script: Next, create a shell script that will execute all the Python files one by one.
- 3.Save and exit.
- 4.Give Execute Permission to the Shell Script.
- 5.Run the Shell Script-Execute the script.

Commands used :

- ☐ **chmod +x run_all.sh** — To give execute permission to the script.
- ☐ **./run_all.sh** — To execute the script and run multiple Python files.

Result: Screen shots



```
vaibhav@Ubuntu: ~/RTOS/lab6/partB/part1
vaibhav@Ubuntu:~/RTOS$ cd lab6
vaibhav@Ubuntu:~/RTOS/lab6$ mkdir partB
vaibhav@Ubuntu:~/RTOS/lab6$ cd partB
vaibhav@Ubuntu:~/RTOS/lab6/partB$ mkdir part1
vaibhav@Ubuntu:~/RTOS/lab6/partB$ cd part1
vaibhav@Ubuntu:~/RTOS/lab6/partB/part1$ touch loop.py
vaibhav@Ubuntu:~/RTOS/lab6/partB/part1$ touch while.py
vaibhav@Ubuntu:~/RTOS/lab6/partB/part1$ touch ifelse.py
vaibhav@Ubuntu:~/RTOS/lab6/partB/part1$
```



A screenshot of a code editor window titled "while.py". The editor has a menu bar with "Open" and a file icon. The code is as follows:

```
1 # whileloop.py
2 count = 0
3 while count < 3:
4     print(f"Count is {count}")
5     count += 1
6
```



A screenshot of a code editor window titled "loop.py". The editor has a menu bar with "Open" and a file icon. The code is as follows:

```
1 # loop.py
2 for i in range(5):
3     print(i)
4
```

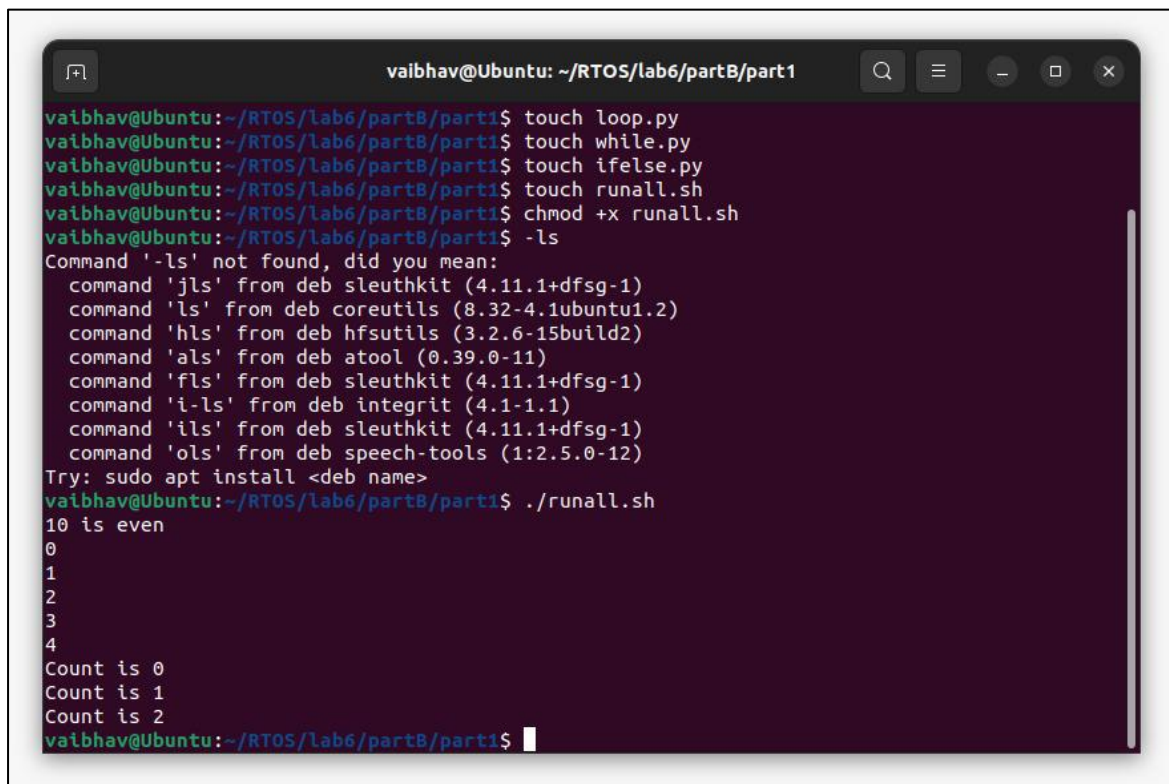


A screenshot of a code editor window titled "ifelse.py". The editor has a menu bar with "Open" and a file icon. The code is as follows:

```
1 # ifelse.py
2 num = 10
3 if num % 2 == 0:
4     print(f"{num} is even")
5 else:
6     print(f"{num} is odd")
7
```



```
1 #!/bin/bash
2 python3 ifelse.py
3 python3 loop.py
4 python3 while.py
5
```



```
vaibhav@Ubuntu: ~/RTOS/lab6/partB/part1
vaibhav@Ubuntu:~/RTOS/lab6/partB/part1$ touch loop.py
vaibhav@Ubuntu:~/RTOS/lab6/partB/part1$ touch while.py
vaibhav@Ubuntu:~/RTOS/lab6/partB/part1$ touch ifelse.py
vaibhav@Ubuntu:~/RTOS/lab6/partB/part1$ touch runall.sh
vaibhav@Ubuntu:~/RTOS/lab6/partB/part1$ chmod +x runall.sh
vaibhav@Ubuntu:~/RTOS/lab6/partB/part1$ -ls
Command '-ls' not found, did you mean:
  command 'jls' from deb sleuthkit (4.11.1+dfsg-1)
  command 'ls' from deb coreutils (8.32-4.1ubuntu1.2)
  command 'hls' from deb hfsutils (3.2.6-15build2)
  command 'als' from deb atool (0.39.0-11)
  command 'fls' from deb sleuthkit (4.11.1+dfsg-1)
  command 'i-ls' from deb integrit (4.1-1.1)
  command 'ils' from deb sleuthkit (4.11.1+dfsg-1)
  command 'ols' from deb speech-tools (1:2.5.0-12)
Try: sudo apt install <deb name>
vaibhav@Ubuntu:~/RTOS/lab6/partB/part1$ ./runall.sh
10 is even
0
1
2
3
4
Count is 0
Count is 1
Count is 2
vaibhav@Ubuntu:~/RTOS/lab6/partB/part1$
```

2. Using && operation

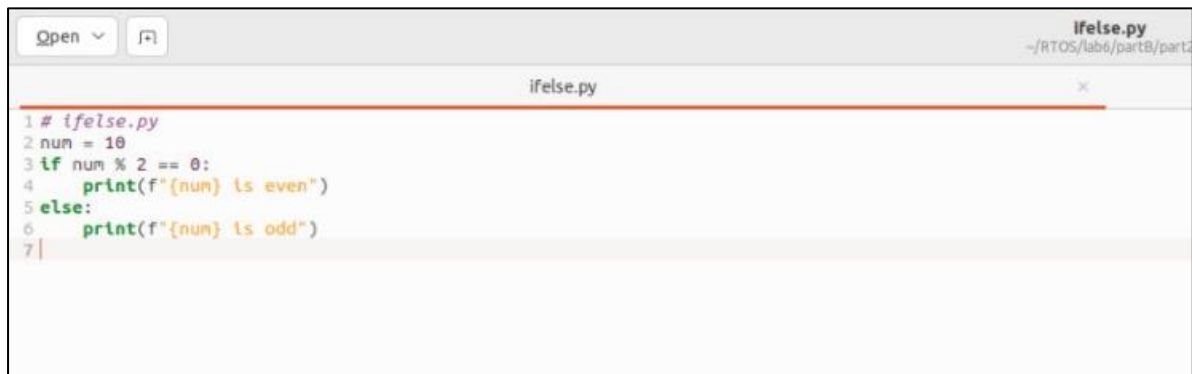
Procedure:

1.Run Multiple Files with &&, we can also run multiple Python files using the && operator, which ensures that the next command will only execute if the previous one is successful.

Commands used :

python3 loop.py && python3 subtraction.py && python3 multiplication.py

Result: Screen shots



A screenshot of a code editor window titled 'ifelse.py'. The editor shows the following Python code:

```
1 # ifelse.py
2 num = 10
3 if num % 2 == 0:
4     print(f"{num} is even")
5 else:
6     print(f"{num} is odd")
7 |
```



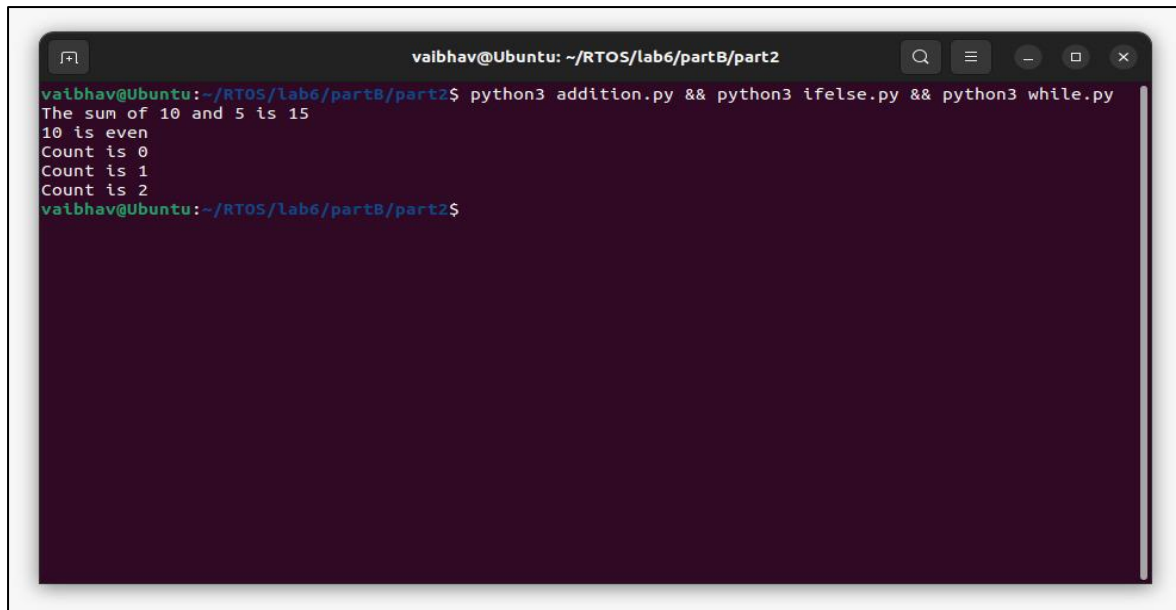
A screenshot of a code editor window titled 'while.py'. The editor shows the following Python code:

```
1 # whileloop.py
2 count = 0
3 while count < 3:
4     print(f"Count is {count}")
5     count += 1
6 |
```



A screenshot of a code editor window titled 'addition.py'. The editor shows the following Python code:

```
1 # arithmetic.py
2 a = 10
3 b = 5
4 result = a + b
5 print(f"The sum of {a} and {b} is {result}")
6 |
```

A terminal window titled 'vaibhav@Ubuntu: ~/RTOS/lab6/partB/part2'. The prompt is 'vaibhav@Ubuntu: ~/RTOS/lab6/partB/part2\$'. The user has entered the command 'python3 addition.py && python3 ifelse.py && python3 while.py'. The output shows the results of these three scripts: 'The sum of 10 and 5 is 15', '10 is even', 'Count is 0', 'Count is 1', and 'Count is 2'. The prompt is now 'vaibhav@Ubuntu: ~/RTOS/lab6/partB/part2\$' again.

```
vaibhav@Ubuntu: ~/RTOS/lab6/partB/part2$ python3 addition.py && python3 ifelse.py && python3 while.py
The sum of 10 and 5 is 15
10 is even
Count is 0
Count is 1
Count is 2
vaibhav@Ubuntu: ~/RTOS/lab6/partB/part2$
```

Q.write short note on interpreter and compiler in the context of linux.



In the Linux environment, both interpreters and compilers are essential tools for executing programs written in various programming languages.

Interpreter:

An interpreter translates and executes code line-by-line. In the context of Python, Linux uses the Python interpreter (python3), which reads and executes Python code one line at a time. Since Python is an interpreted language, the interpreter is invoked to run Python scripts directly, without needing to convert the code into a machine language beforehand. This allows for quicker execution during development, as there is no need for a separate compilation step.

Compiler:

A compiler, on the other hand, translates the entire program code into machine language or bytecode before execution. In languages like C or C++, a compiler like GCC (GNU Compiler Collection) is used. The compiler translates the source code into an executable binary that can be run on the system. Compilation is done once, and the resulting binary can be executed multiple times without recompiling.

Conclusion:

In this exercise, we explored how to execute Python files in a Linux environment. We demonstrated the process of running a single Python file that performs arithmetic operations and multiple Python files containing control structures like if-else, for, and while loops. We also covered two methods for executing multiple Python files: using a shell script and the && operator. Both approaches are efficient for automating tasks and executing multiple scripts sequentially. By utilizing Linux commands and shell scripts, we can streamline the execution of Python programs, making it easy to manage and automate processes in a Linux environment.