UBUNTU COMMANDS

OPERATING SYSTEMS LABS

ASSIGNMENT # 01

Submitted By

ASJID TAHIR

(19P-0085)

Submitted to

MR. MUHAMMAD ABDULLAH

(INSTRUCTOR CS)

DEPARTMENT OF COMPUTER SCIENCE

FAST NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCINCES, PESHAWAR

Session 2019-2023

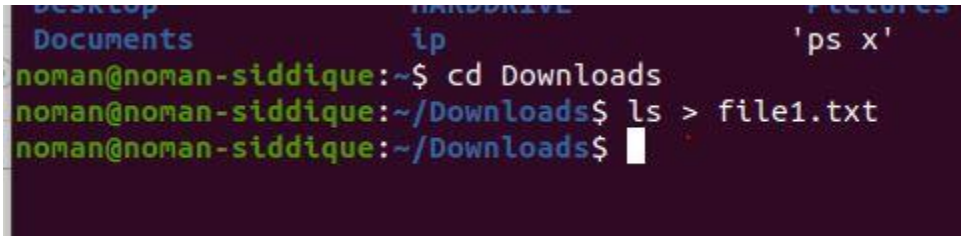
I/O redirection Command:

In this lesson, we will explore a powerful feature used by command line programs called ***input/output redirection*.** As we have seen, many commands such as **`ls`** print their output on the display. This does not have to be the case, however. By using some special notations we can *redirect* the output of many commands to files, devices, and even to the input of other commands.

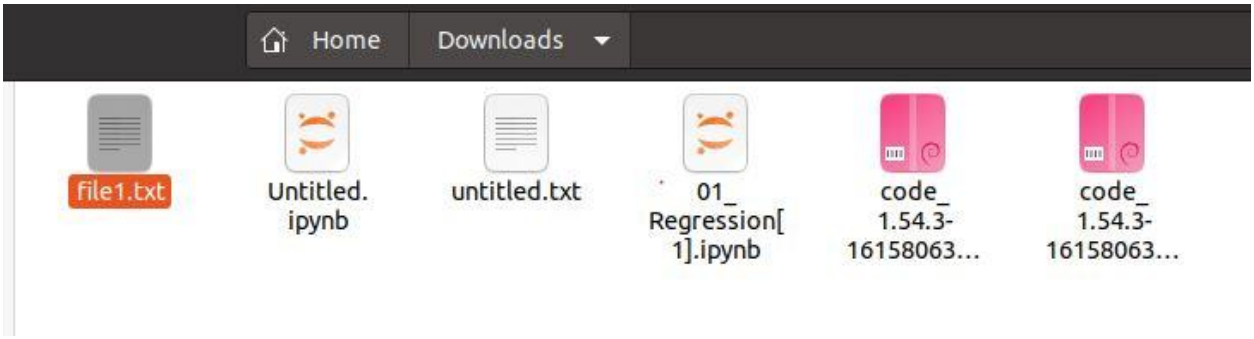
Standard Output:

Most command line programs that display their results do so by sending their results to a facility called ***standard output***. By default, standard output directs its contents to the display. To redirect standard output to a file, the ">" character is used like this:

```
Ls>file1.txt
```



In this example, the ls command is executed and the results are written in a file named file_list1.txt. Since the output of **ls** was redirected to the file, no results appear on the display.

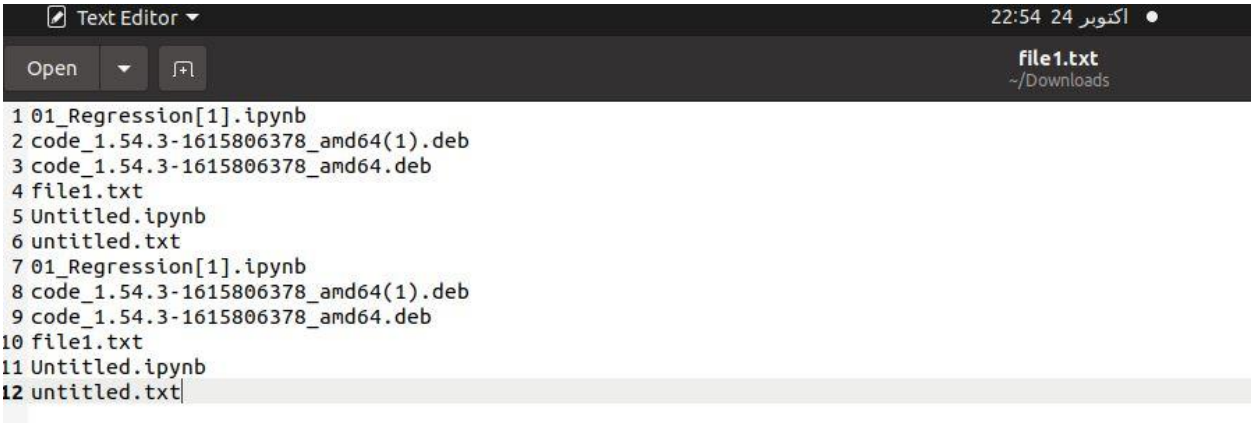


Each time the command above is repeated, file_list.txt is overwritten from the beginning with the output of the command **ls**. To have the new results *appended* to the file instead, we use ">>" like this:

Ls>>file1.txt

When the results are appended, the new results are added to the end of the file, thus making the file longer each time the command is repeated.

The new result is added to the end of the file.



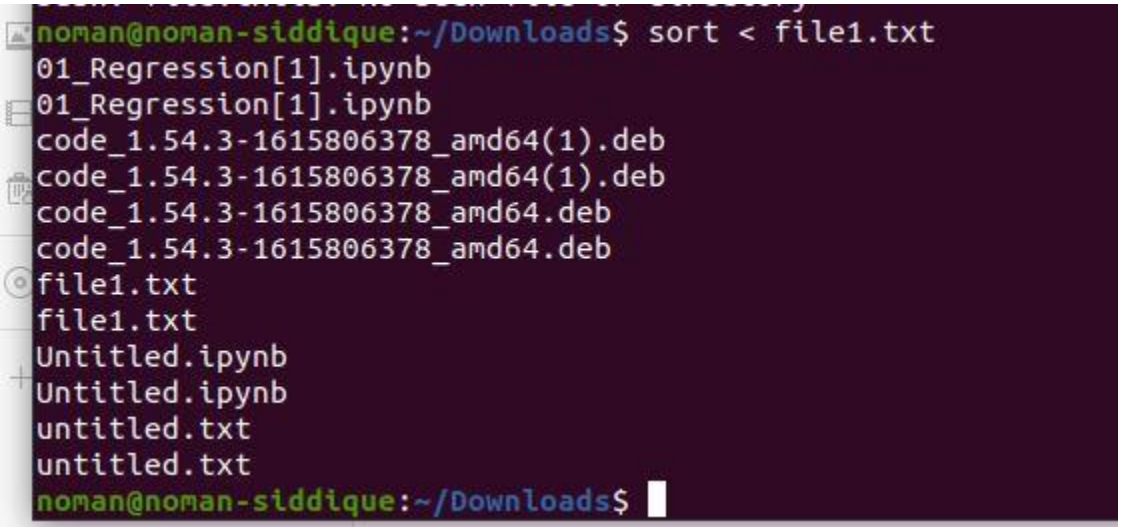
Standard Input:

Many commands can accept input from a facility called *standard input*. By default, standard input gets its contents from the

keyboard, but like standard output, it can be redirected. To redirect standard input from a file instead of the keyboard, the "<" character is used like this:

It show the text of file in command.

```
sort < file1.txt
```



In the example above, we used the **sort** command to process the contents of `file_list1.txt`. The results are output on the display since the standard output was not redirected. We could redirect standard output to another file like this:

It created the file name with sorted file and sorted the text of file1 and store it in file1.txt.

```
sort < file1.txt > sorted_file1.txt
```