**RE-DIRECTIONS**

**“>” and “>>” both of these operators represent output redirection in Linux.**

Operators are characters that offer various functionalities. And these redirection operators redirect the result or the output. Anyone can learn the Linux terminal, but you need to put effort to have a firm grasp on it. So, how’s “>” different from “>>”? What’s the main difference? In this tutorial, you’ll learn redirection in a nutshell and the difference between “>” and “>>” in Linux.

**What is Redirection?**

Redirection is a Linux feature. Inputs and outputs of a command can be redirected to another command or a file using redirection operators.

Linux programmers use:

1*) Input redirection operator* to redirect the input given

2) *Output redirection operator*to redirect the output.

A less-than sign (<) represents input redirection. On the other hand, a greater than sign (>) is used for the output redirection. “<” and “>” are also called angled brackets. But what’s the need for using “>” and “>>” both to redirect the output? Let’s find the reason for using both of these as output redirects.

**Difference between “>” and “>>” in Linux**

“>” and “>>” both are output (STDOUT) direction operators; however, they differ in the following ways.

**The “>” Operator**

“>” overwrites an already existing file or a new file is created providing the mentioned file name isn’t there in the directory. This means that while making changes in a file you need to overwrite certain any existing data, use the “>” operator. This sign is used to redirect the output to something else than stdout.

**Example 1:**

echo “Welcome to Linux” > my\_file\_1.txt

After executing the above command, you’ll find that a text file “my\_file\_1.txt” is created in the directory. It’ll contain the text “Welcome to Linux” in it.

To check whether the file has been created successfully or not, type:

ls

The following command helps read the file type.

 cat my\_file\_1.txt

Now, let’s execute the same command with a different text.

echo “Learn latest tips about Linux” > my\_file\_1.txt

And type the command below to read the file.

cat my\_file\_1.txt

You’ll see that the new text has successfully overwritten the earlier text.

**Example 2:**

ls > allmyfiles.txt

This is another example using the “>” operator. The above command creates the file called “allmyfiles.txt”. And fills it up with the directory listing given by the “ls” command.

**Example 3:**

> newzerobytefile

Here a zero byte file is created having the name “newzerobytefile”.

Alternatively, this command can overwrite a file that is already present with the same name. And it’ll make the file zero bytes in size.

**The “>>” Operator**

“>>” operator appends an already present file or creates a new file if that file name doesn’t exist in the directory.

**Example 1:**

echo “Welcome to Linux” >> my\_file\_2.txt

The above command will create a file by the name “my\_file\_2.txt” in your current directory.

Once the command is executed, type: **ls** This will verify if the file has been created successfully.

Read the file by:

cat my\_file\_2.txt

Let’s alter the text, now, into:

echo “Learn latest tips about Linux” >> my\_file\_2.txt

Since you’re using a file that was created previously, to check all the modifications made by “>>”, run the following command:

cat my\_file\_2.txt

And you’ll see instead of overwriting the previously entered text, the “>>” operator has appended the text.

**Example 2:**

echo "End of directory listing" >> allmyfiles.txt

The above command will add “End of directory listing” at the end of a file called “allmyfiles.txt”

**Summary**

Because of the operators, some commands may seem to be a bit tricky to understand for first-time Linux users. That’s why I’ve used more than one example to explain each of the operators. So, what we learned is, the “>” is the output redirection operator used for overwriting files that already exist in the directory. While, the “>>” is an output operator as well, but it appends the data of an existing file. Often, both of these operators are used together to modify files in Linux.