ULI101: INTRODUCTION TO UNIX / LINUX AND THE INTERNET

WEEK 5: LESSON 2

PIPELINE COMMANDS
MULTIPLE / MULTILINE COMMANDS

LESSON 2 TOPICS

Redirection – Part 2

- Purpose of **Pipeline Commands**
- Linux Pipeline Command Syntax: **|**
- **tee** Command

Multiple / Multi-Line Commands

- Multiple Linux Commands using Semicolon ";" and Grouping: **( )**
- Issuing Large Linux Commands over Multiple Lines

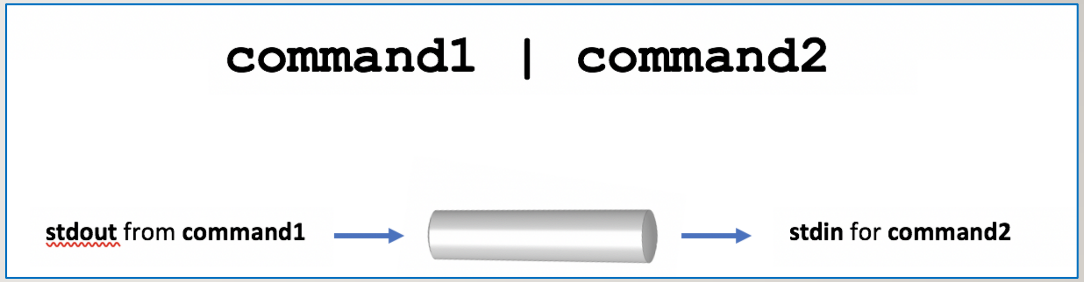
Perform Week 5 Tutorial

- Investigations 2 and 3
- Review Questions (Questions 5 – 12)

PIPELINE COMMANDS

Pipeline Commands use a meta symbol “|” (called a pipe) to allow a command’s **standard output** to be redirected into the **standard input** of other commands WITHOUT having to use **temporary** files.

Therefore, a few simple commands can be **combined** to form a more <u>powerful</u> pipeline command.



command1 | command2

Examples:

```
ls -al | more
ls | sort -r
ls | sort | more
ls -l | cut -d" " -f2 | tr 'a-z' ‘A-Z"
ls | grep Linux | head -5
```

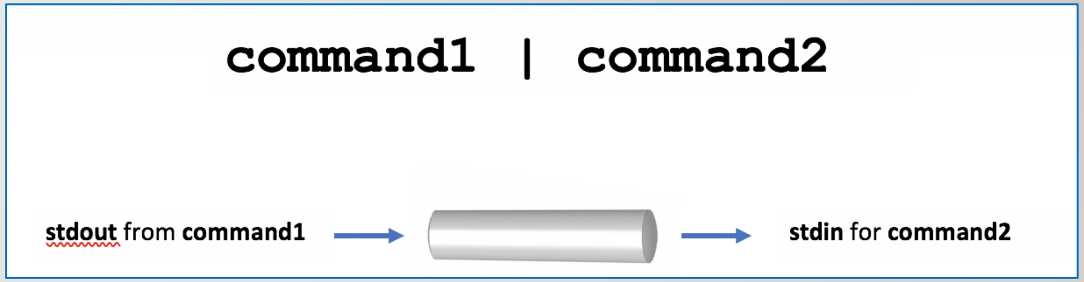
PIPELINE COMMANDS

Filters

Commands to the **right** of the pipe symbol are referred to as **filters**.

They are called *filters* since those commands are used to **modify** the stdout of the previous command.

Many commands can be "piped" together, but these commands (filters) must be chained in **a specific order**, depending on what you wish to accomplish.



```
command1 | command2
```

stdout from **command1** → → stdin for **command2**

PIPELINE COMMANDS

Instructor Demonstration

Your instructor will now demonstrate how to issue **Pipeline Commands**.

PIPELINE COMMANDS

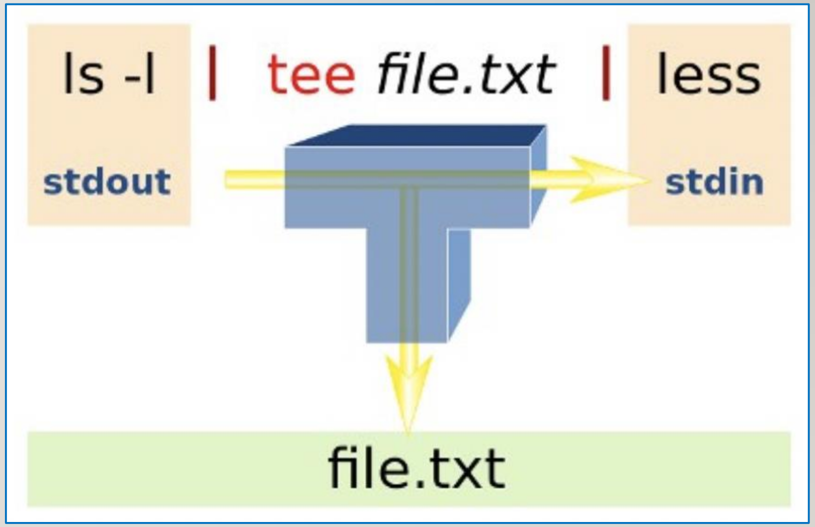
The **tee** utility can be used to split the flow of standard output between a **text file** and the **terminal screen**.

The **tee** option **-a** can be used to add content to the **bottom** of an existing file as opposed to *overwriting* the file's previous contents.

The reason for the name "**tee**" is that the splitting of the flow of information resembles a capital T.

Examples:

```
ls | tee unsorted.txt | sort
ls | grep Linux | tee matched.txt | more
ls | head -5 | tee -a
```



PIPELINE COMMANDS

Instructor Demonstration

Your instructor will now demonstrate how to use the **tee** command within a **Pipeline Command**.

MULTIPLE / MULTI-LINE COMMANDS

There are ways that **multiple commands** can be run within a **single** command line.

You can separate commands by using the **semi-colon character ";"**.

```
command1 ; command2 ; command3
```

Example:

```
sleep 5; ls
```

Multiple commands can also be **grouped** by using **parentheses** to force commands to be run **together** (for example, to redirect **stdout** to a file)

Example:

```
(echo "Who is logged in:"; who) > whoson
```

(***Note:*** *all command output is sent to a file*)

MULTIPLE / MULTI-LINE COMMANDS

Commands may also be **spread-out over multiple lines**, making it easier (for humans) to interpret a long command.

You can add a **backslash** quoting symbol "\" at the end of a line. The \ symbol “*quotes-out*” the meaning of the **ENTER** key as text (i.e. *new-line*) as instead of running the command.

Example:

```
echo "This will be split over multiple \
lines. Note that the shell will realize \
that a pipe requires another command, so \
it will automatically go to the next line” \
| tr '[a-z]' '[A-Z]’
```

```
command1  |   \
command2  |   \
command3
```

MULTIPLE / MULTI-LINE COMMANDS

Instructor Demonstration

Your instructor will now demonstrate how to issue Multiple Commands / Multi-Line Linux Commands:

- Multiple Linux Commands using semicolon “;”
- Multiple Linux Commands using Grouping ( )
- Mult-Line Linux Commands using Backslash \

HOMEWORK

Getting Practice

Perform the online **Tutorial 5**

(Due: Friday Week 6 @ midnight for a 2% grade):

- INVESTIGATION 2: REDIRECTION USING PIPES
- INVESTIGATION 3: ISSUING MULTIPLE UNIX/LINUX COMMANDS
- LINUX PRACTICE QUESTIONS (Questions 6 – 12)