School of Computer Science

COMP30640

Lab 2
Text processing commands & Redirections/Pipes

Teaching Assistant:	Thomas Laurent
Coordinator:	Anthony Ventresque
Date:	Friday 21st September, 2018
Total Number of Pages:	5

Echo and Special Characters.

The goal of this section is to play with special characters and to show how they impact the execution of commands, and can be disabled conveniently. If you need to interrupt/stop the execution of a command (if you're stuck), you can use the combination of keys *control + c*.

- print on the screen a simple word, for instance "hello" (use echo).
- echo can print several strings - print "hello everybody"
- you can also use echo to print non alphabetical characters on screen (as long as they do not belong to the list of special characters). Use echo to print "Hello to the 2 of you!!"
- Disable the special character # in the following string (using the character \) and print it on screen using echo: "# is a very useful character in bash"
- Likewise for the following: "# is more useful than \in bash"
- Print the following using echo: "# is more useful than \in bash but less than *"
- When you need to disable a lot of special characters, you can use simple and double quotes to disable all (simple quotes) or some (double quotes - do not disable $, ‘ and \) of them. Print the following two sentences using simple then double quotes: "# is less useful than * in bash" and "# is more useful than \in bash"

Selecting Columns. *Charles I of England?*

Linux command cut is used for text processing. You can use this command to extract portions of text from a file by selecting columns.

Option -cN extracts only column (character) N of each line from a file:

```
$> cut -c2 /etc/passwd
```

will extract only the second character of each line (the 2nd column of each line).

Range of characters can also be extracted from a file by specifying start and end position delimited with -.

```
$> cut -c2-5 /etc/passwd
```

Either start position or end position can be passed to the cut command with the -c option.

The following command specifies only the start position before the '-'. This example extracts from the 10th character to the end of each line.

```
$> cut -c10- /etc/passwd
```

The following commad specifies only the end position after the '-'. This example extracts 10 characters from the beginning of each line.

```
$> cut -c-10 /etc/passwd
```

Instead of selecting x number of characters, if you like to extract a whole field, you can combine options -f and -d. Option -f specifies which field you want to extract, and option -d specifies the field delimiter that is used in the input file.

The following example displays only the first field of each line from the /etc/passwd file using the field delimiter : (colon). In this case, the 1st field is the username.

```
$> cut -d":" -f1 /etc/passwd
```

You can also extract more than one fields from a file. The example below displays the username and home directory of users.

```
$> cut -d":" -f1,6 /etc/passwd
```

To display a range of fields specify start field and end field as shown below. In this example, we are selecting fields 1 through 4, 6 and 7 (fields 1, 2, 3, 4, 6 and 7).

```
$> cut -d":" -f1-4,6,7 /etc/passwd
```

Sorting. *The Last Shall Be First*

The sort command rearranges the lines in a text file so that they are sorted lexicographically.

```
$> sort /etc/passwd
```

These are the default sorting rules:

- a number is before a letter.
- letters follow their order in the alphabet.
- a lowercase letter is before a same uppercase letter.

You can change the default sorting rules by providing a suitable parameter. E.g..

- Reverse sort:

  ```
  $> sort -r /etc/passwd
  ```

- Ignore case:

  ```
  $> sort -f /etc/passwd
  ```

You can also concatenate and sort several files at once:

```
$> sort /etc/passwd /etc/group
```

You can also save the result of the sort in a another file:

```
$> sort /etc/passwd -o result.txt
```

Duplicates. *Castor Troy: It's like looking in a mirror, only not.*

This command filters duplicate adjacent lines.

```
$> uniq /etc/passwd
```

It can filter them even by ignoring the case:

```
$> uniq -i /etc/passwd
```

You can report the duplicate lines by:

```
$> uniq -d /etc/passwd
```

You can also print the number of occurrences of each line:

```
$> uniq -c /etc/passwd
```

Grep. *Look it up!*

Grep prints out the lines containing a certain string.

```
$> grep root /etc/passwd
```

Options:

- -c: only gives the number of matching lines
- -v: Shows only the lines that do not match the pattern. Inverted search.
- -i: ignore case
- -n: gives the line number as well as the matching lines.

Getting files on and from the server

At times you will need to upload files from your own machine to the server, or download some of the files you have on the server to your local machine. Here are some ways of doing this.

scp

The scp command works just as cp but lets you transfer files from one machine to another one. You can use it if you have a bash terminal on your machine. The syntax is as follows:

```
$> scp source target
```

For example, if we want to download the file unirank.csv from the home folder ($\tilde{()}$) of user thomas on the csserver to the directory we are currently in (./), we would write:

```
$> scp thomas@csserver.ucd.ie:~/unirank.csv ./
```

If we want to upload unirank.csv from the current folder to the home folder of user thomas on the csserver then :

```
$> scp unirank.csv thomas@csserver.ucd.ie:~/
```

To copy directories, we use he -r option.

FileZilla

FileZilla offers a graphical interface that allows you to do the same thing as scp. It is available on windows, macOS and linux at `https://filezilla-project.org/download.php`.

Using FileZilla is pretty straightforward. If needed you can find instructions at `https://wiki.filezilla-project.org/FileZilla_Client_Tutorial_(en)`

public_html

You might need to share files with others who do not have access to your home directory. In order to do this you can put the files in a directory called public_html and they will be publicly available at the URL `http://csserver.ucd.ie/~username/fileName`.

Exercise 1

- What is the output of the commands:
  - `echo {0..9}`
  - `echo 1.{0..9}`
  - `echo {A..C}{0..2}`
- Use sort to sort the content of `/etc/passwd`
- Use grep to list only the line from `/proc/meminfo` that shows total memory (the name of the variable is MemTotal).
- Use grep to find all the lines with "to" in the `syslog` file (`/var/log/alternatives.log.1`). Then redirect this output into a file in your home directory.
- List all of the files in a directory in reverse date order, so that the most recent is at the bottom (ls).

Exercise 2

In this exercise, we will be exploring a dataset containing information regarding universities. Follow these steps using the commands described above.

1. Download the dataset as unirank.csv using wget:

   ```
   wget -O unirank.csv http://csserver.ucd.ie/~thomas/unirank.csv
   ```

2. Explore the file and its structure (`cat`, `head`, `tail`, `less`, ...)
3. Find all the "colleges" in the list (`grep`).
4. List and group all the universities by state (`cut`, `sort`).
5. Find the number of institutes per state (`cut`, `sort`, `uniq`). (which state has the most institutes in the dataset?)

6. Create a list (x, y) of couples containing the Tuitions and Fees (x) and the rank (y) for each institute. Plot the values using Excel, gnuplot, etc. What do you notice?