Introduction to Data Analytics

COSC / DATA 301 – Command Line

MUCH OF THE WORK IN THIS COURSE HAS BEEN DEVELOPED BY PRECEDING COLLEAGUES

Why learn the Command Line?

The ***command line*** is the text interface to the computer.

Understanding the command line allows you to interact with the computer in ways that you often cannot with the user interface.

The command line is commonly used for scripting and automation of tasks and when accessing remote systems.

What is the Command Line?

The ***command line*** is the text interface to the computer that accepts commands that the computer will execute. These commands include:

- starting programs

- navigating directories and manipulating files

- searching, sorting, and editing text files

- system and environment configuration

The command line is part of the ***operating system***, which is software that manages your computer including all devices and programs.

- Common operating systems include Windows, Mac OS, and Linux/Unix.

Windows Command Line

The command line on Windows dates back to the original Microsoft operating system called **DOS** (**Disk Operating System**) in 1981.

This command line interface is still part of all modern Windows operating systems and is accessible as the "Command Prompt".

Command Prompt

It is commonly used for system administration and scripting.

Command Line - Windows

Mac OS Command Line

The command line for Mac OS uses the same commands as Linux. It can be opened using `Finder` then `Utilities` then `Terminal`.
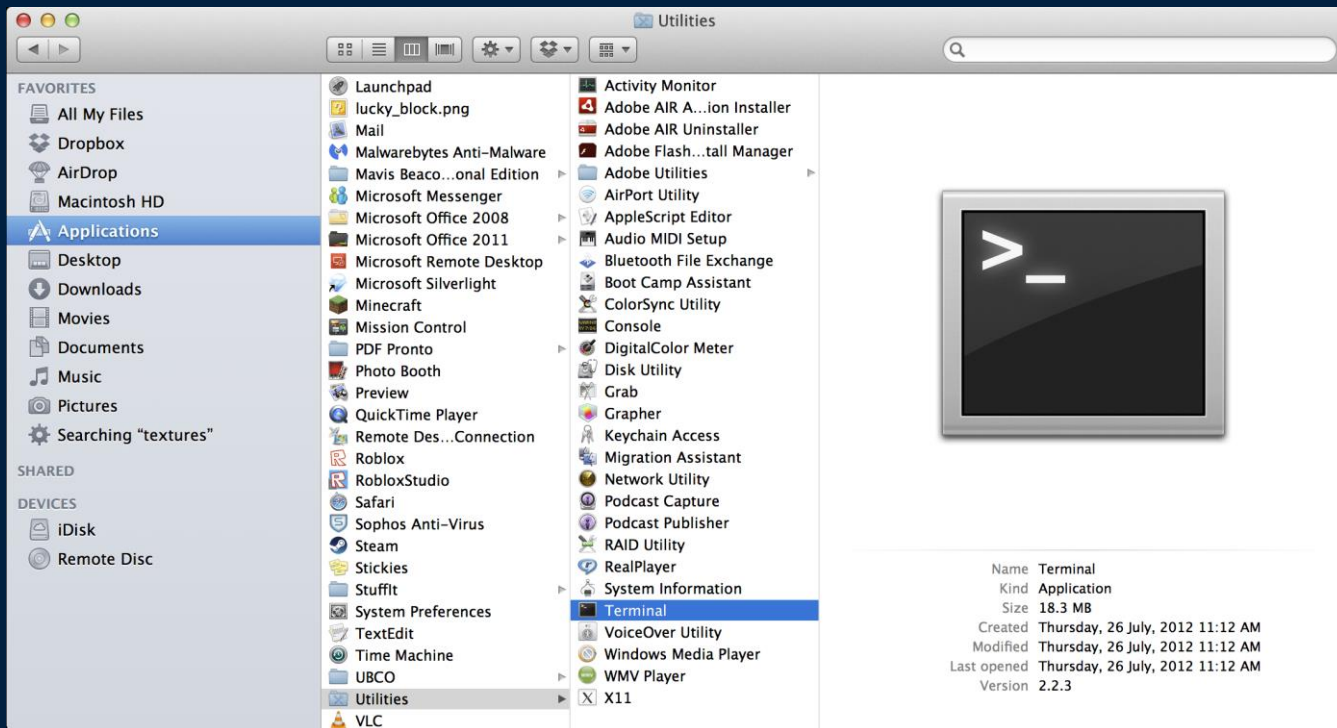
Command Line – Mac/Linux


```
A4003829:~ rlawrenc$ pwd
/Users/rlawrenc
A4003829:~ rlawrenc$ echo Hello
Hello
A4003829:~ rlawrenc$ mkdir 301
A4003829:~ rlawrenc$ cd 301
A4003829:301 rlawrenc$ nano test.txt
A4003829:301 rlawrenc$ ls
test.txt
A4003829:301 rlawrenc$ cat test.txt
This is a test!
A4003829:301 rlawrenc$ rm test.txt
A4003829:301 rlawrenc$ cd ..
A4003829:~ rlawrenc$ rmdir 301
A4003829:~ rlawrenc$
```

Entering a Command

Enter a *command* at a *prompt*.

- The prompt may be a > or a $ or customized by the user.

Press ENTER to execute the command.

On Windows, commands are mostly case-insensitive while on Mac/Linux they are case-sensitive.

```
A4003829:~ rlawrenc$ pwd
/Users/rlawrenc
A4003829:~ rlawrenc$ echo Hello
Hello
A4003829:~ rlawrenc$ mkdir 301
A4003829:~ rlawrenc$ cd 301
A4003829:301 rlawrenc$ nano test.txt
A4003829:301 rlawrenc$ ls
test.txt
A4003829:301 rlawrenc$ cat test.txt
This is a test!
A4003829:301 rlawrenc$ rm test.txt
A4003829:301 rlawrenc$ cd ..
A4003829:~ rlawrenc$ rmdir 301
A4003829:~ rlawrenc$
```
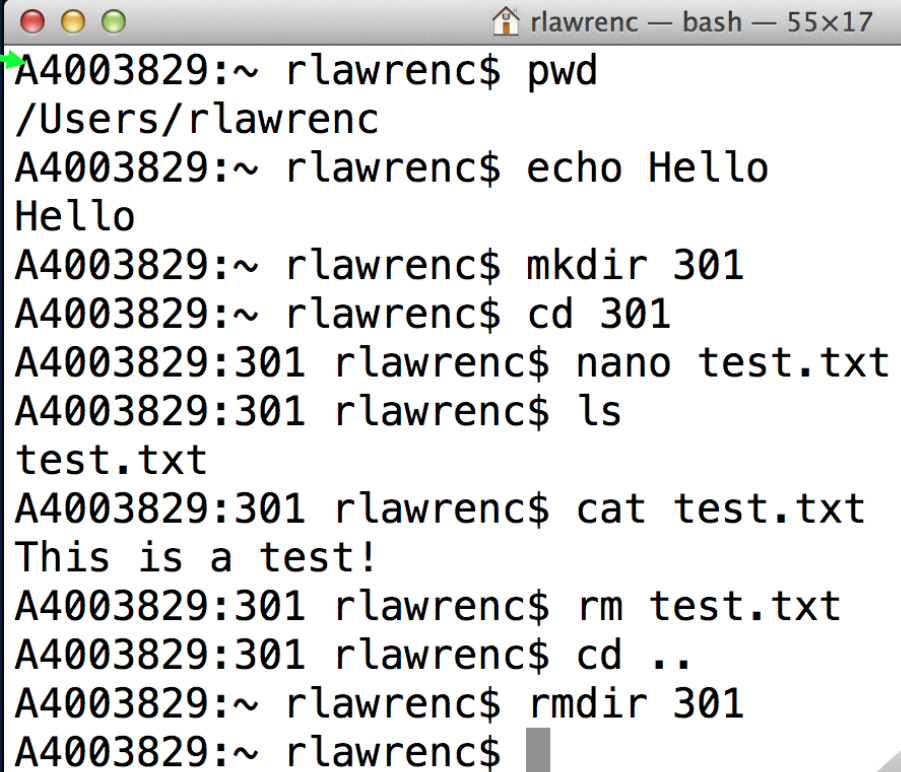
File System
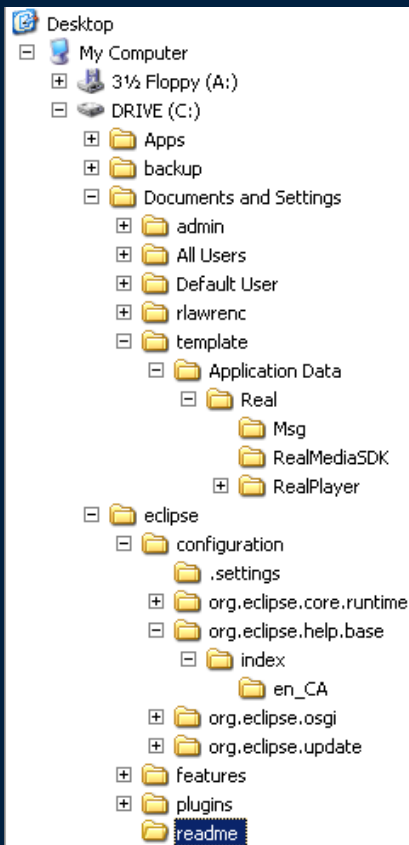
The **file system** organizes data on a device as a hierarchy of directories and files.

Each **folder** (directory) has a name and can contain any number of files or subdirectories.

Each **file** has a name.

The user can change (navigate) directories in the hierarchy.

Absolute versus Relative Paths

The ***root*** of the file system is the directory "**/**".

- There is only one root of a directory hierarchy.

A path to a new location (from your current location) can be specified as an ***absolute path*** from the root:

```
cd /Users/rlawrenc/301/folder
```

or a ***relative path*** from your current location:

```
cd 301/folder
```

To back up one directory level, use the command: `cd ..`

Commonly Used File Navigation Commands

	Windows	Mac OS and Linux
List contents of directory	`dir`	`ls`
Change directory	`cd` *301*	`cd` *301*
Print working directory	`cd`	`pwd`
Make a directory	`mkdir` *301*	`mkdir` *301*
Remove a directory	`rmdir` *301*	`rmdir` *301*
Rename a file	`ren` *old.txt new.txt*	`mv` *old.txt new.txt*
Remove a file	`del` *file.txt*	`rm` *file.txt*
Copy a file	*copy src.txt dest.txt*	*cp src.txt dest.txt*

Commonly Used Text Related Commands

	Windows	Mac OS and Linux
Open a text editor	`notepad`	`nano`
Echo output	`echo Hello`	`echo Hello`
Output contents of a file	`more file.txt`	`cat file.txt`
Search text files	`find`	`grep`
Sort text files	`sort`	`sort`

Wildcards

A *wildcard* character allows for matching file names with more flexibility.

The ? represents any *one* character in a file name.

Example: `file?.txt` would match `file1.txt`.

The * (asterisk) matches any number of characters (including zero).

Example: `*.txt` would match anything ending with `.txt` (`a.txt`).

Navigating the Command Line

	Windows Key	Mac OS Key
Previous command in history	Up	Up
Next command in history	Down	Down
First command in history	PageUp	
Last command in history	PageDown	
Move to start of line	Home	Ctrl+A
Move to end of line	End	Ctrl+E
Auto-compete file name	Tab	Tab

Pausing or Cancelling Commands

To **pause** a command:

- Windows: Press `Ctrl+S` or the `Pause` key. To resume, press any key.
- Mac: `Control+Esc` or `Command+.`

To **cancel** a command, press `Ctrl+C` or `Ctrl+Break`.

- The command is canceled, and the command prompt returns.
- However, any actions performed before the cancel are not undone.

Try it: Navigating Directories with Commands

Question: Using a terminal window on your computer, perform the following actions:

1) Create a directory called `301`.
2) Change into the directory `301`.
3) Echo `I am awesome!`
4) Show your current directory (print working directory).
5) Create a text file called `message.txt` with a message in it.
6) List the contents of your directory.
7) Rename the file `message.txt` to `test.txt`. Verify the name change.
8) Delete the `test.txt` file.
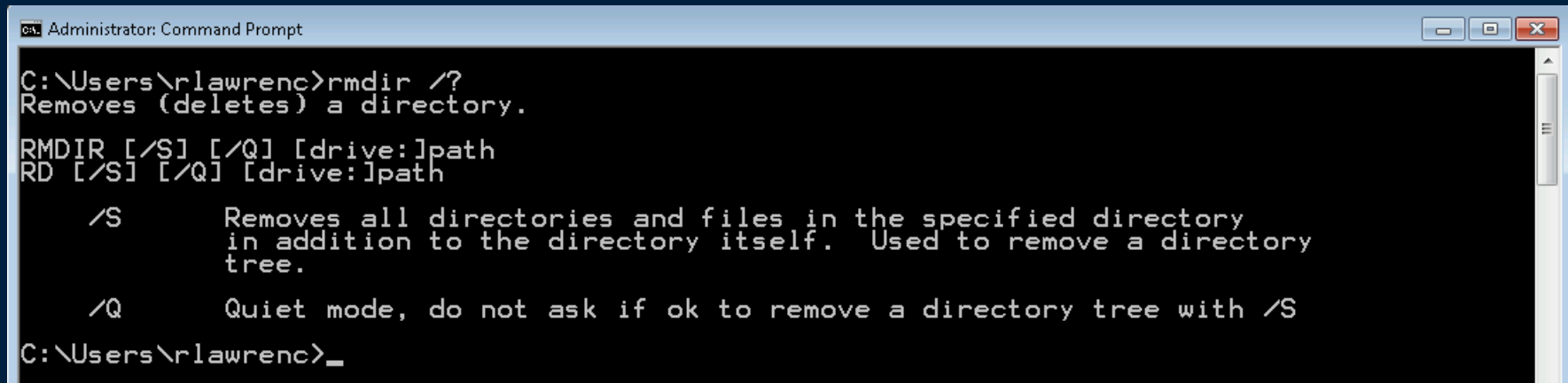9) Change directory to directory above `301`.
10) Delete directory `301`.

Command Arguments

A command can take ***arguments*** that changes its behavior.

- Example: Path was an argument for the `cd` command. e.g. `cd` *301*

On Windows, commands also can be modified by a ***switch*** (or extension) which is usually a slash then a letter (e.g. `/S`).

- To find out what is available, run the command with: `/?`

```
Administrator: Command Prompt

C:\Users\rlawrenc>rmdir /?
Removes (deletes) a directory.

RMDIR [/S] [/Q] [drive:]path
RD [/S] [/Q] [drive:]path

    /S      Removes all directories and files in the specified directory
            in addition to the directory itself.  Used to remove a directory
            tree.

    /Q      Quiet mode, do not ask if ok to remove a directory tree with /S

C:\Users\rlawrenc>_
```

Command Arguments – Mac/Linux

On Mac/Linux, arguments are separated by spaces and begin with –.

An explanation of arguments can be found by using `man` then the command name. Example: `man cp`

```
CP(1)                        BSD General Commands Manual
                  CP(1)

NAME
     cp -- copy files

SYNOPSIS
     cp [-R [-H | -L | -P]] [-fi | -n] [-apvX]
         source_file target_file
     cp [-R [-H | -L | -P]] [-fi | -n] [-apvX]
         source_file ... target_directory

DESCRIPTION
     In the first synopsis form, the cp utility
     copies the contents of the source_file to
:
```

Standard Input, Output, and Error

Standard input (`stdin`) is the default input device (usually a keyboard) into the terminal.

Standard output (`stdout`) is the location where output is sent after a command is run. The default is the terminal window.

Standard error (`stderr`) is the location where error messages are displayed (typically the terminal window).

Redirecting Input

By default, a command gets its input from standard input and outputs results to standard output.

A command can get its input from the output of another command by using the **pipe** (|) symbol. Example:

```
cat test.txt | wc
```

Also can use redirect input (<) to send input to a command. Example:

```
cat < test.txt
```

Note that can chain together multiple pipes.

- Note the example commands are Mac OS/Linux only: `wc` is not on Windows.

Redirecting Output

Redirect output using > which will overwrite the file:

```
sort test.txt > sorted.txt
```

Use >> to append to the existing file:

```
sort test.txt >> sorted.txt
```

Redirection Summary

	Symbol
Redirect input	<
Redirect output	>
Redirect output (append)	>>
Pipe output to input of next command	|

Escape Symbol

An ***escape symbol*** is used when a command requires input that contains a character with a special meaning. The escape symbol indicates this character is data not part of the command.

- On Windows, the caret (^) indicates that whatever character that follows it is data rather than part of the command. Example:

  ```
  cp test.txt a^&b.txt
  ```

- On Linux, use the backslash (\).

This is especially common when dealing with spaces in a file name. The other way to handle file names with spaces is to enclose them in double quotes:

```
cp test.txt "c:\program files\file spaces.txt"
```

Environment Variables

Environment variables allow for customization and control of the command and system environment.

Current variables are seen using the `set` or `env` command.

Important variables:
- `$PATH` – list of directories where commands/applications will be found
- `$HOME` – user home directory

Finding Text in Files

The `grep` command allows for searching for text in files that match a pattern (Mac/Linux only, `find` on Windows).

- `grep` stands for "global regular expression print"
- Search is case-sensitive (use `-i` for case-insensitive) and can contain regular expressions.

Example:

`grep er *.txt`

- search for `er` in any file that ends in `.txt`

Windows: `find "er" *.txt`

Batch Files

A ***batch program*** (also commonly called a *batch file or command file*) is a text file that contains a sequence of commands to be executed.

You define the sequence of commands, name the sequence, and then execute the commands by entering the name at a command prompt. Any action you can take by typing a command at a command prompt can be encapsulated in a batch program.

In Windows files typically end in `.bat` or `.cmd` and on Mac/Linux with `.sh`.

Batch files can take arguments like other commands.

Try it: Using Batch Files

Question: Using a terminal window on your computer, create a batch file that performs these actions:

Before creating the batch file, create a file called `numbers.txt` that has the numbers `one, two, three, …, ten.`

In the batch file, called `myscript.bat` (or `.sh`):

1) Write a command to sort `numbers.txt` and output as `sorted.txt.`

2) Write a command to output the word count on `numbers.txt` to `count.txt.` (only for Linux or Mac command line)

3) Write commands to take `numbers.txt` and append its data three times into the file `output.txt.`

4) Use `grep (or find)` to search for "e" in `output.txt` and write results as file `search.txt.`

5) Output the contents of `sorted.txt, output.txt,` and `search.txt.`

6) Run your batch file.

27

Connecting to Another Computer using SSH

Secure shell or SSH is a protocol allowing remote login to other machines to execute commands.

- The network communication is encrypted for security.
- An open-source program on campus is Putty.

Using SSH allows you to connect and execute commands on another machine even when you do not have physical access to that machine.

SSH may be used to send or retrieve data from other computers for analysis.

Conclusion

The **_command line_** is the text interface to the computer that accepts commands that the computer will execute including running programs, manipulating files, and running scripts.

The command line allows for automation and more control than may be available in the user interface. It may also be the only way to interact with the machine if connecting via SSH.

The command environment allows for redirecting the standard input and output using input/output redirection and pipes.

Objectives

- Define command line and list some of its uses
- Explain the purpose of an operating system
- Know how to open the command line window on Mac OS and Windows
- Be able to enter commands and stop them
- Define: file system, folder, file
- Explain the difference between an absolute and relative path
- Use command line shortcuts to save time
- Be able to match wildcards involving ? and *
- Be able to cancel a command
- Explain standard input, standard output, and standard error
- Be able to use input and output redirection and pipes (?, >, < , >>)
- Explain the reason for an escape symbol
- Define and explain the purpose of environment variables.

Objectives (2)

- Be able to use grep to search text files.
- Explain the purpose of a batch program.
- Be able to connect to another machine using SSH.

THE UNIVERSITY OF BRITISH COLUMBIA