

# WEEK 5

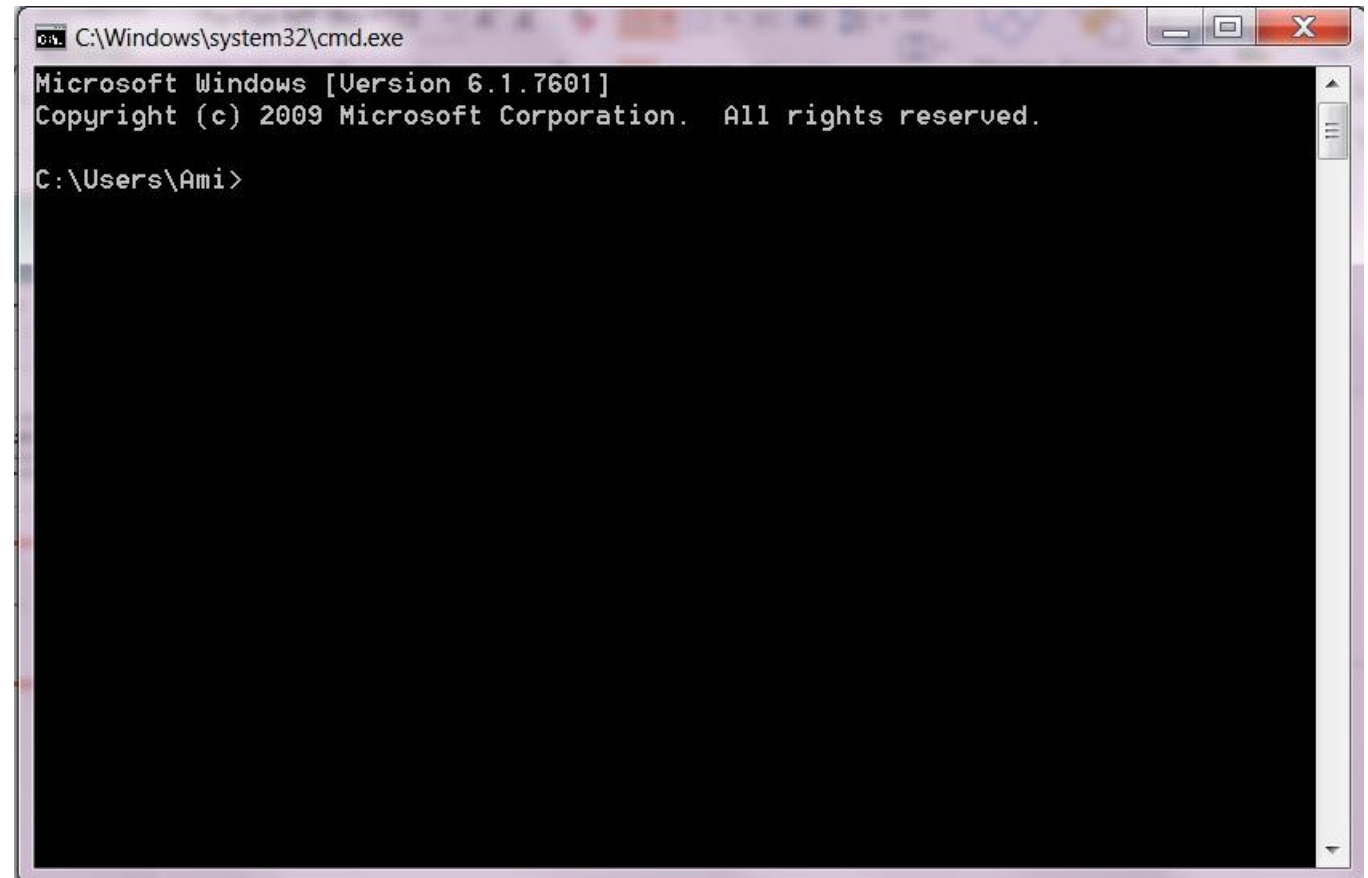
Gates

# COMMAND LINE OPERATIONS

- 1) Command line operations are often necessary whether using Windows, MAC, or Unix/Linux.
- 2) Throughout the Data Analytics Master's Degree and in many classes, you will need to perform command line activities either on your computer or on Unix/Linux.
- 3) Some people own MAC computers and some people own PC computers. For this reason, I will have a few slides on each.
- 4) Everyone will have to learn and know how to perform basic command line and other activities in Unix/Linux.

# COMMAND LINE OPERATIONS ON A PC

- 1) On a Windows machine, it is easy to access the command line.
- 2) Go to the Start menu (lower left) and type in “cmd”. Then press enter.
- 3) In Windows, this command line is DOS.
- 4) There are 100s of commands you can perform from this location.

A screenshot of a Windows Command Prompt window. The title bar at the top reads "C:\Windows\system32\cmd.exe". The window has standard Windows window controls (minimize, maximize, close) on the right. The main area is black with white text. It displays "Microsoft Windows [Version 6.1.7601]" and "Copyright (c) 2009 Microsoft Corporation. All rights reserved." followed by a blank line and the prompt "C:\Users\Ami>".

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

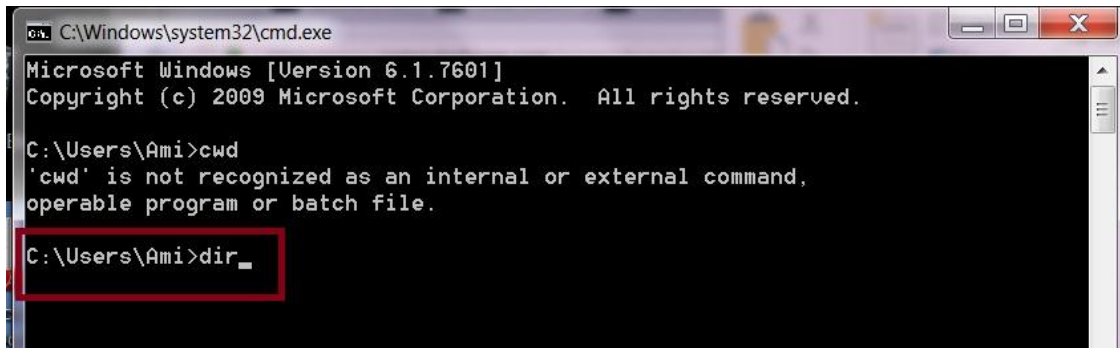
C:\Users\Ami>
```

# WINDOWS/DOS COMMAND LINE OPTIONS

## dir

The “dir” command will list all the files and folders that are in the current directory.

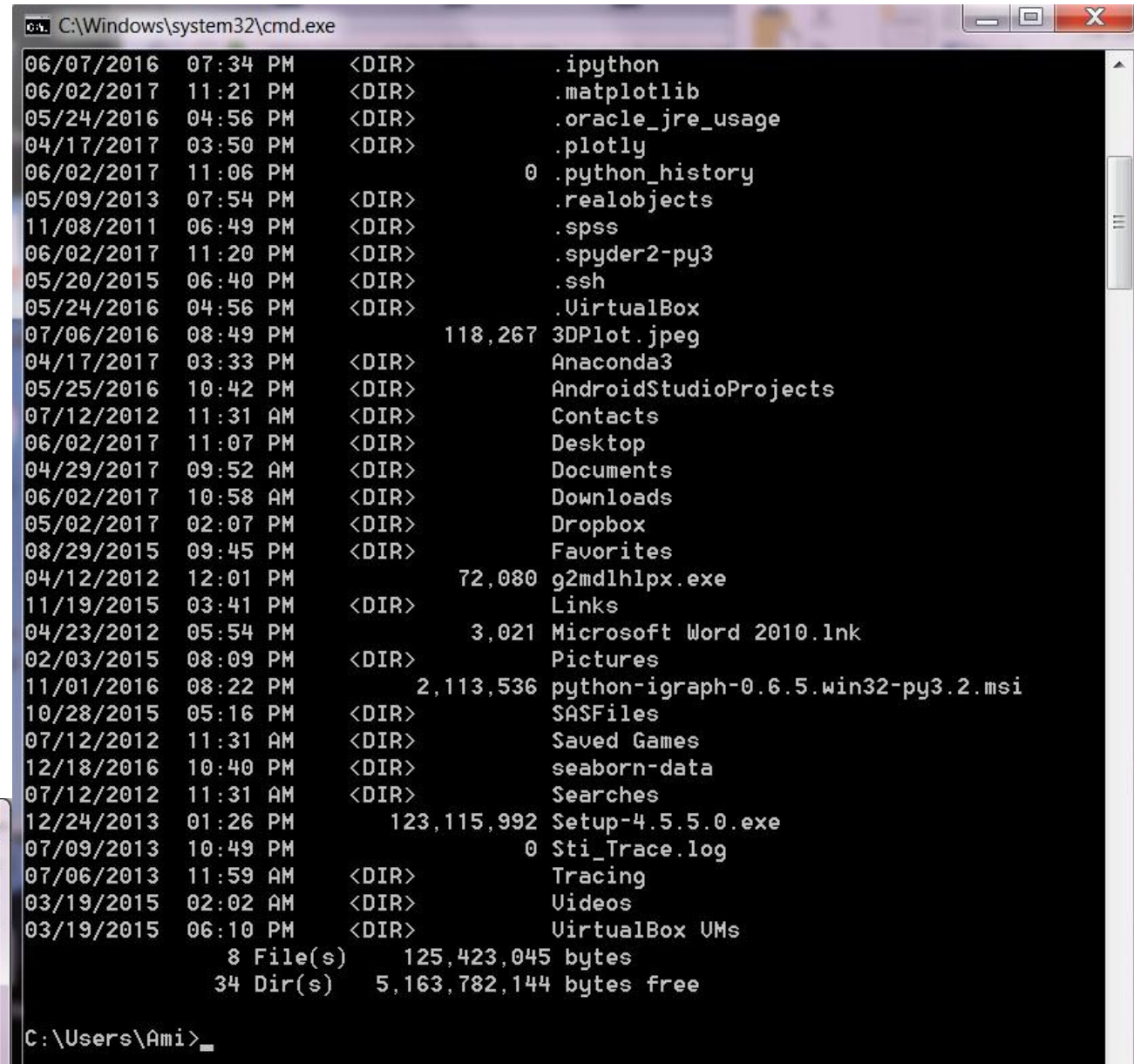
You can see the current dir by looking at the path.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Ami>cd
'cd' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\Ami>dir_
```



```
C:\Windows\system32\cmd.exe
06/07/2016 07:34 PM <DIR> .ipython
06/02/2017 11:21 PM <DIR> .matplotlib
05/24/2016 04:56 PM <DIR> .oracle_jre_usage
04/17/2017 03:50 PM <DIR> .plotly
06/02/2017 11:06 PM 0 .python_history
05/09/2013 07:54 PM <DIR> .realobjects
11/08/2011 06:49 PM <DIR> .spss
06/02/2017 11:20 PM <DIR> .spyder2-py3
05/20/2015 06:40 PM <DIR> .ssh
05/24/2016 04:56 PM <DIR> .VirtualBox
07/06/2016 08:49 PM 118,267 3DPlot.jpeg
04/17/2017 03:33 PM <DIR> Anaconda3
05/25/2016 10:42 PM <DIR> AndroidStudioProjects
07/12/2012 11:31 AM <DIR> Contacts
06/02/2017 11:07 PM <DIR> Desktop
04/29/2017 09:52 AM <DIR> Documents
06/02/2017 10:58 AM <DIR> Downloads
05/02/2017 02:07 PM <DIR> Dropbox
08/29/2015 09:45 PM <DIR> Favorites
04/12/2012 12:01 PM 72,080 g2mdlhlpx.exe
11/19/2015 03:41 PM <DIR> Links
04/23/2012 05:54 PM 3,021 Microsoft Word 2010.lnk
02/03/2015 08:09 PM <DIR> Pictures
11/01/2016 08:22 PM 2,113,536 python-igraph-0.6.5.win32-py3.2.msi
10/28/2015 05:16 PM <DIR> SASFiles
07/12/2012 11:31 AM <DIR> Saved Games
12/18/2016 10:40 PM <DIR> seaborn-data
07/12/2012 11:31 AM <DIR> Searches
12/24/2013 01:26 PM 123,115,992 Setup-4.5.5.0.exe
07/09/2013 10:49 PM 0 Sti_Trace.log
07/06/2013 11:59 AM <DIR> Tracing
03/19/2015 02:02 AM <DIR> Videos
03/19/2015 06:10 PM <DIR> VirtualBox VMs
      8 File(s)      125,423,045 bytes
     34 Dir(s)      5,163,782,144 bytes free

C:\Users\Ami>
```

# COMMAND: CD

cd will allow you to “change directories”.

You can change to any other **folder**. In DOS, a folder is a directory.

Here is typed, cd Documents. Then I typed cd Python Scripts.

Now I am in location  
C:\Users\Ami\Documents\Python Scripts

```
09/06/2016 11:33 PM 1,346 ViewAirFile0zone.csv
09/06/2016 07:52 PM 2,679 ViewAirFile0zone.txt
09/06/2016 11:33 PM 1,558 ViewAirFile0zone_2.csv
09/06/2016 11:33 PM 1,335 ViewAirFilePM25.csv
09/06/2016 07:45 PM 3,026 ViewAirFilePM25.txt
09/06/2016 11:33 PM 1,541 ViewAirFilePM25_2.csv
08/27/2016 07:38 PM 250 WebScrape1.py
01/31/2017 04:29 PM 4,242 Week2Examples2017.py
09/10/2016 08:07 PM 2,132 Week2InClassActivitySolutionLisa.py
10/02/2016 11:16 AM 4,402 Week3ICAdat.csv
02/10/2017 11:59 AM 699 Week4Ex2.py
10/03/2016 02:19 PM 5,783 Week4ICA_nytimesSOLUTION.py
10/03/2016 06:56 PM 8,636 Week4_5ICA_Clustering.py
10/04/2016 10:00 PM 53,865 Week4_5ICA_Graph_for_Clicks.png
10/04/2016 10:00 PM 50,346 Week4_5ICA_Graph_for_Gender.png
10/04/2016 10:00 PM 51,666 Week4_5ICA_Graph_for_Impressions.png
10/04/2016 10:00 PM 54,951 Week4_5ICA_Graph_for_Signed_In.png
10/04/2016 10:00 PM 8,016 Week4_5ICA_StudentHANDOUT_FINALVERSION.py

10/11/2016 02:41 PM 1,738 Week6ClassExample.py
10/25/2016 02:53 PM 1,101 Week8Examples2.py
06/11/2016 06:58 PM 292 WhileBreak.py
06/10/2016 11:47 PM 244 WhileLoop1.py
07/24/2016 08:26 PM 530 WhileLoop2.py
09/01/2016 04:30 PM 1,292 WIKIAPI.py
09/01/2016 04:30 PM 208,836 WikiDataFile.txt
09/20/2016 06:36 PM 792 WithOPenExampleCh8.py
03/19/2017 09:45 PM 905 WordCloud.py
03/10/2017 08:29 PM 155 WordCloud_2.py
03/11/2017 02:42 PM 95 WordFrequencies.txt
08/28/2016 11:46 PM 613 WorkingRE.py
09/10/2016 08:10 PM 33 zip.txt
09/20/2016 11:26 PM <DIR> __pycache__
368 File(s) 159,063,101 bytes
10 Dir(s) 5,163,393,024 bytes free

C:\Users\Ami\Documents\Python Scripts>
```

# PATHS AND LOCATIONS

It is critical that you know where files and folders are on your computer.

You should be able to access Python and R files and folders from the command line as well as from the IDEs.

When installing new modules/packages into Python, you may often use the command line to do this.

# OTHER COMMAND LINE OPERATIONS: WINDOWS

1) `cd ..`

Moves you backwards one directory (or folder)

2) `notepad Testfile.txt`

This will create a new text file and will open the editor.

3) `.bat` files

`.bat` files are called batch files. They can contain command line commands that you want to complete.

Example:

On the command line, type `notepad Mybat.bat`

In that file, type:

`@echo on`

`cls`

`dir`

# CONTINUED

To run the batch file, on the command line in the directory where the batch file was saved, type the name of the batch file.

This will run (execute) all the lines in the batch file.

Batchfiles can be used to run a series of command line commands.

By typing, *Mybat*, I will run all the commands in the batchfile.

A screenshot of a Windows command prompt window. The window has a black background with white text. The title bar is partially visible at the top, showing "PERATION". The command prompt shows the current directory as "C:\Users\Ami\Documents". The first command entered is "notepad MyBat.bat", and the second command entered is "Mybat".

```
PERATION C:\Users\Ami\Documents>notepad MyBat.bat  
C:\Users\Ami\Documents>Mybat
```



# MORE COMMON COMMANDS IN WINDOWS/DOS

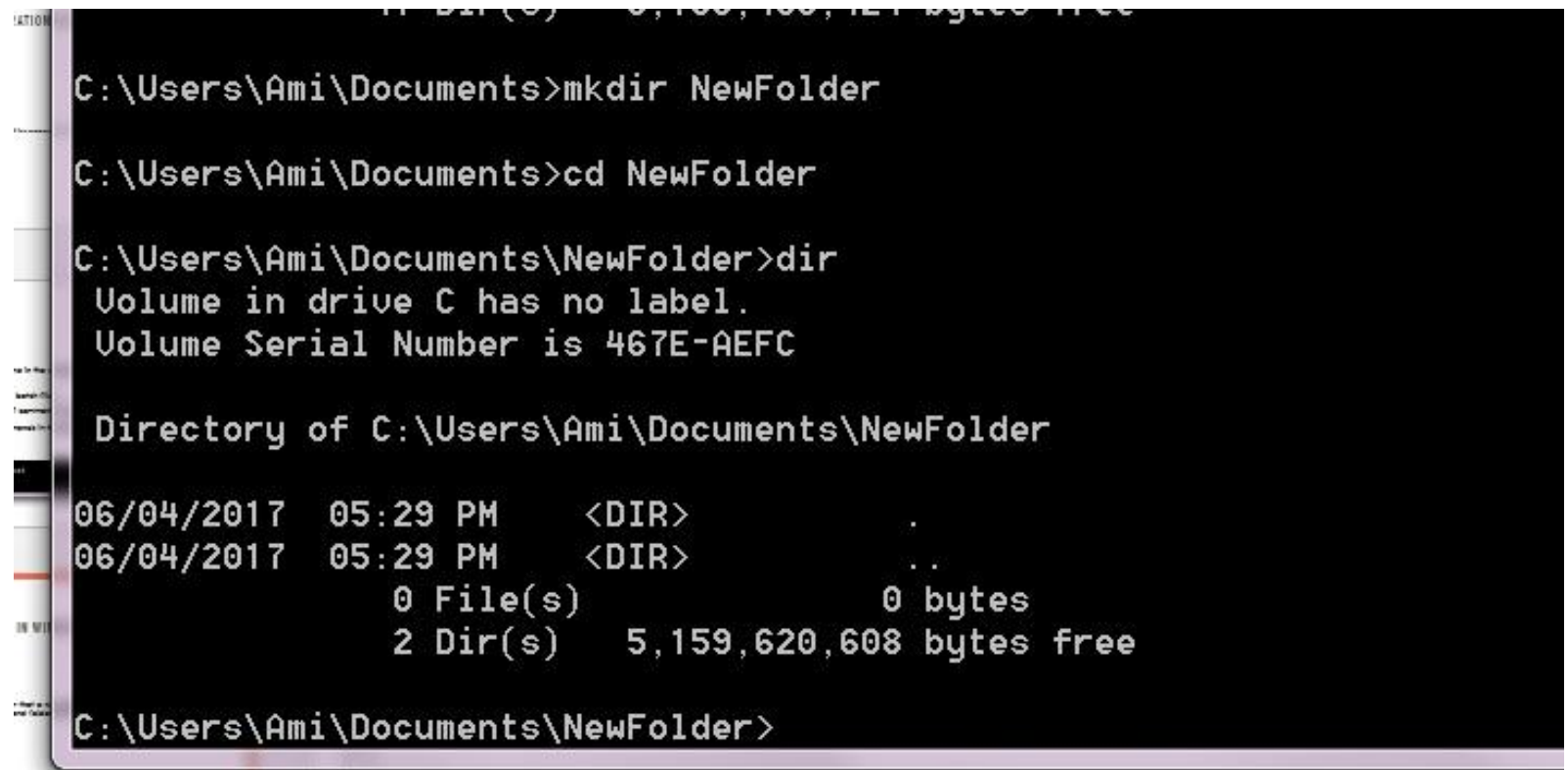
## 1) mkdir

This will create a new folder.

2) If I type, mkdir NewFolder, I can confirm that a new folder (directory) can be created. I can then create or add files and folders to that new folder.

Here – I created a new folder called NewFolder.

I then used “cd” to change directories to that new folder location.



```
C:\Users\Ami\Documents>mkdir NewFolder

C:\Users\Ami\Documents>cd NewFolder

C:\Users\Ami\Documents\NewFolder>dir
Volume in drive C has no label.
Volume Serial Number is 467E-AEFC

Directory of C:\Users\Ami\Documents\NewFolder

06/04/2017  05:29 PM    <DIR>          .
06/04/2017  05:29 PM    <DIR>          ..
               0 File(s)                0 bytes
               2 Dir(s)  5,159,620,608 bytes free

C:\Users\Ami\Documents\NewFolder>
```

# MOVE AND COPY

You can **move** or **copy** files from one location to another.

See to the right where I moved Mybat.bat to NewFolder.

```
C:\Users\Ami\Documents>move Mybat.bat NewFolder
1 file(s) moved.

C:\Users\Ami\Documents>cd NewFolder

C:\Users\Ami\Documents\NewFolder>dir
Volume in drive C has no label.
Volume Serial Number is 467E-AEFC

Directory of C:\Users\Ami\Documents\NewFolder

06/04/2017  05:31 PM    <DIR>          .
06/04/2017  05:31 PM    <DIR>          ..
06/04/2017  05:23 PM                18 MyBat.bat
                    1 File(s)                18 bytes
                    2 Dir(s)  5,159,321,600 bytes free

C:\Users\Ami\Documents\NewFolder>
```

# DELETING IN WINDOWS/DOC: DEL

There are 100s of commands.

It is possible to delete one or more files at once.

You can use **wildcards** to delete many files.

For example,

```
del *.gif
```

will delete all files that have a .gif extension.

Be careful with delete!

Here is a good reference for other commands and methods:

<https://www.computerhope.com/issues/chusedos.htm>

# GET HELP USING “HELP”

```
C:\Windows\system32\cmd.exe - help dir
C:\Users\Ami\Documents\NewFolder->help dir
Displays a list of files and subdirectories in a directory.

DIR [drive:][path][filename] [/A[[:]attributes]] [/B] [/C] [/D] [/L] [/N]
  [/O[[:]sortorder]] [/P] [/Q] [/R] [/S] [/T[[:]timefield]] [/W] [/X] [/4]

[drive:][path][filename]
    Specifies drive, directory, and/or files to list.

/A      Displays files with specified attributes.
attributes  D Directories                      R Read-only files
              H Hidden files                    A Files ready for archiving
              S System files                     I Not content indexed files
              L Reparse Points                   - Prefix meaning not

/B      Uses bare format (no heading information or summary).
/C      Display the thousand separator in file sizes. This is the
        default. Use /-C to disable display of separator.
/D      Same as wide but files are list sorted by column.
/L      Uses lowercase.
/N      New long list format where filenames are on the far right.
/O      List by files in sorted order.
sortorder  N By name (alphabetic)                S By size (smallest first)
              E By extension (alphabetic)         D By date/time (oldest first)
              G Group directories first           - Prefix to reverse order

/P      Pauses after each screenful of information.
/Q      Display the owner of the file.
/R      Display alternate data streams of the file.
/S      Displays files in specified directory and all subdirectories.
/T      Controls which time field displayed or used for sorting
timefield  C Creation
              A Last Access
              W Last Written

/W      Uses wide list format.
/X      This displays the short names generated for non-8dot3 file
        names. The format is that of /N with the short name inserted
        before the long name. If no short name is present, blanks are
```

# MAC COMMAND LINE: CALLED A TERMINAL

The MAC offers the same type of command line options as Windows.

While Windows runs on DOS, MAC runs on Unix.

As such, methods on MACs are similar (if not the same) as methods on Unix/Linux.

As I have not tested every computer or every OS – be prepared for some differences 😊

# OPENING THE MAC COMMAND LINE TERMINAL

## How to open the command line.

Before you can use it, you need to be able to find it.

So what we need to do is open the terminal. On OS X, open your Applications folder, then open the Utilities folder. Open the Terminal application. You may want to add this to your dock. I like to launch terminal by using Spotlight search in OS X, searching for “terminal”.



<http://blog.teamtreehouse.com/introduction-to-the-mac-os-x-command-line>



# Anatomy of the Console

First let's clarify a few terms.

**Console:** This is the system as a whole. This is both the command line as well as the output from previous commands.

**Command Line:** This is the actual line in a console where you type your command.

**Prompt:** This is the beginning of the command line. It usually provides some contextual information like who you are, where you are and other useful info. It typically ends in a **\$** . After the prompt is where you will be typing commands.

**Terminal:** This is the actual interface to the console. The program we use to interact with the console is actually a “terminal emulator”, providing us the experience of typing into an old school terminal from the convenience of our modern graphical operating system.

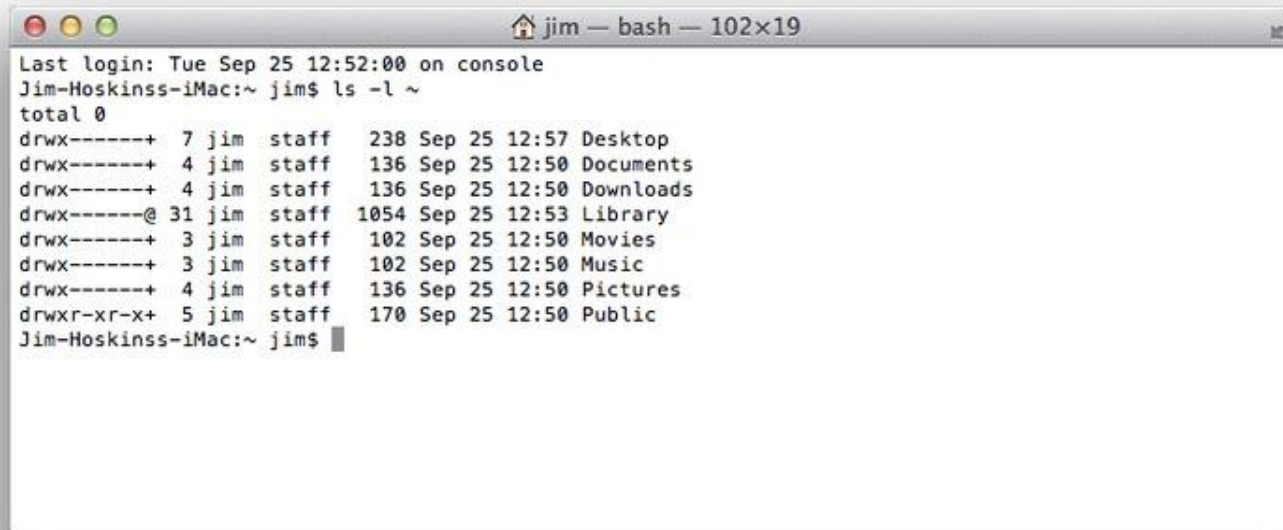
<http://blog.teamtreehouse.com/introduction-to-the-mac-os-x-command-line>

# RUNNING A COMMAND ON A MAC

## Running a Command.

Nearly all commands follow a common pattern with 3 main parts. The program, the options, and the arguments. Let's see an example.

```
$ ls -l ~
```



```
jim — bash — 102x19
Last login: Tue Sep 25 12:52:00 on console
Jim-Hoskinss-iMac:~ jim$ ls -l ~
total 0
drwx-----+ 7 jim  staff   238 Sep 25 12:57 Desktop
drwx-----+ 4 jim  staff   136 Sep 25 12:50 Documents
drwx-----+ 4 jim  staff   136 Sep 25 12:50 Downloads
drwx-----@ 31 jim  staff  1054 Sep 25 12:53 Library
drwx-----+ 3 jim  staff   102 Sep 25 12:50 Movies
drwx-----+ 3 jim  staff   102 Sep 25 12:50 Music
drwx-----+ 4 jim  staff   136 Sep 25 12:50 Pictures
drwxr-xr-x+ 5 jim  staff   170 Sep 25 12:50 Public
Jim-Hoskinss-iMac:~ jim$
```



# Where Are You?

In the console, you are always working in a directory, or folder, on your computer. We call this your working directory. You can see where you are using `pwd` (short for print working directory)

```
$ pwd
```

A screenshot of a macOS terminal window. The title bar at the top reads "jim — bash — 102x19". The terminal content shows the user "jim" at the prompt "Jim-Hoskinss-iMac:~ jim\$". They have entered the command "pwd", and the output is "/Users/jim". The prompt "Jim-Hoskinss-iMac:~ jim\$" is shown again on the next line, followed by a cursor.

```
Jim-Hoskinss-iMac:~ jim$ pwd
/Users/jim
Jim-Hoskinss-iMac:~ jim$
```

This command will print out your current location. You can change your directory with `cd` (short for change directory). If you pass it an argument, it will change your to that location, if it exists. Without an argument, it will take you to your home directory (`~`).

```
$ cd Documents
```

You'll notice that I just passed it a directory named Documents, because I was in my home directory, that contains a directory called Documents. This is relative path, because I specified my destination relative to my current directory. I can provide an absolute path by providing the full path beginning with the `/`, or starting with my home directory (`~`) such as:

```
$ cd /Users/jim/Documents
```

or

```
$ cd ~/Documents
```

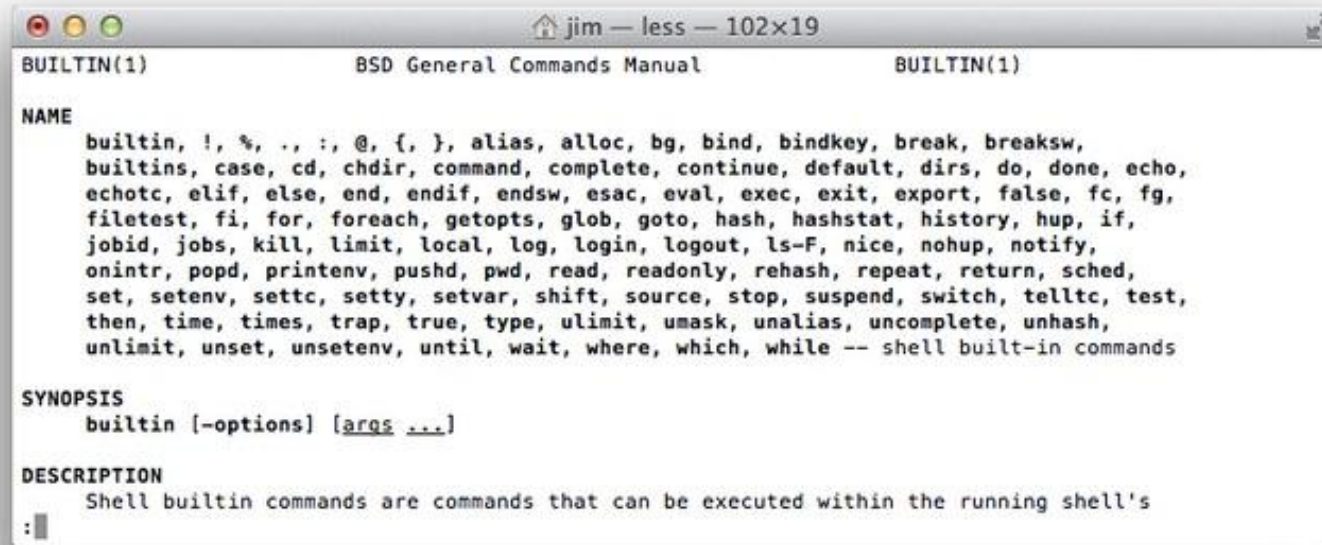
If you want to navigate “up”, that is to the directory that contains your current directory, you can use the special name `..` which you can even use separated by slashes to navigate several levels up.

From my Documents directory, this command will take me up to my home directory.

```
$ cd ..
```

# MAN PAGES FOR HELP

```
$ man ls
```



The screenshot shows a terminal window titled "jim — less — 102x19". The window displays the man page for "builtin" commands. The page is divided into sections: "NAME", "SYNOPSIS", and "DESCRIPTION". The "NAME" section lists various builtin commands. The "SYNOPSIS" section shows the command syntax. The "DESCRIPTION" section explains that shell builtin commands can be executed within the running shell's environment.

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

NAME
    builtin, !, %, ., :, @, {, }, alias, alloc, bg, bind, bindkey, break, breaksw,
    builtins, case, cd, chdir, command, complete, continue, default, dirs, do, done, echo,
    echotc, elif, else, end, endif, endsw, esac, eval, exec, exit, export, false, fc, fg,
    filetest, fi, for, foreach, getopts, glob, goto, hash, hashstat, history, hup, if,
    jobid, jobs, kill, limit, local, log, login, logout, ls-F, nice, nohup, notify,
    onintr, popd, printenv, pushd, pwd, read, readonly, rehash, repeat, return, sched,
    set, setenv, settc, setty, setvar, shift, source, stop, suspend, switch, telltc, test,
    then, time, times, trap, true, type, ulimit, umask, unalias, uncomplete, unhash,
    unlimited, unset, unsetenv, until, wait, where, which, while -- shell built-in commands

SYNOPSIS
    builtin [-options] [args ...]

DESCRIPTION
    Shell builtin commands are commands that can be executed within the running shell's
    :|
```

The manual can be scrolled with the arrow keys or space bar. Pressing q will quit.

# OTHER COMMON MAC COMMANDS

- `mkdir` - Make a new directory
- `touch` - Make a new empty file
- `cp` - Copy a file
- `mv` - Move a file
- `rm` - Remove a file or directory (learn about the `-r` option)
- `less` - Show the contents of a file in a scrolling buffer

# RESOURCES AND REFERENCES

<https://www.davidbaumgold.com/tutorials/command-line/>

<http://mally.stanford.edu/~sr/computing/basic-unix.html>

<https://www.lifewire.com/dos-commands-4070427>

# UNIX/LINUX

Unix/Linux has many many command line options.

Here is a good link that shows many of them:

<http://mally.stanford.edu/~sr/computing/basic-unix.html>

<http://mally.stanford.edu/~sr/computing/more-unix.html>

<https://www.osc.edu/supercomputing/unix-cmds>

<https://www.tutorialspoint.com/unix/unix-useful-commands.htm>

# WHAT IS PUTTY/SSH?

RE: <http://www.putty.org/>

<https://the.earth.li/~sgtatham/putty/0.60/htmldoc/Chapter1.html>

PuTTY is a Telnet and SSH terminal software for Unix and Windows platforms that enables any users to remotely access computers over the internet.

PuTTY is free 😊

<https://the.earth.li/~sgtatham/putty/0.60/htmldoc/index.html>

# WHAT IS SSH, TELNET, RLOGIN?

SSH, Telnet and Rlogin are three ways of doing the same thing: **logging in to a multi-user computer from another computer, over a network.**

Multi-user operating systems, such as Unix and VMS, usually present a **command-line** interface to the user, much like the 'Command Prompt' or 'MS-DOS Prompt' in Windows. The system prints a prompt, and you type commands which the system will obey.

Using this type of interface, there is no need for you to be sitting at the same machine you are typing commands to. **The commands, and responses, can be sent over a network,** so you can sit at one computer and give commands to another one, or even to more than one.

**SSH, Telnet and Rlogin are *network protocols*** that allow you to do this. On the computer you sit at, you run a ***client***, which makes a network connection to the other computer (the ***server***). The network connection carries your keystrokes and commands from the client to the server, and carries the server's responses back to you.



# SSH, TELNET, RLOGIN: DIFFERENCES

## How do SSH, Telnet and Rlogin differ?

This list summarizes some of the differences between SSH, Telnet and Rlogin.

**SSH** (which stands for 'secure shell') is a recently designed, high-security protocol. It uses strong cryptography to protect your connection against eavesdropping, hijacking and other attacks. Telnet and Rlogin are both older protocols offering minimal security.

**SSH and Rlogin** both allow you to log in to the server without having to type a password. (Rlogin's method of doing this is insecure, and can allow an attacker to access your account on the server. SSH's method is much more secure, and typically breaking the security requires the attacker to have gained access to your actual client machine.)

**SSH** allows you to connect to the server and automatically send a command, so that the server will run that command and then disconnect. So you can use it in automated processing.

# WHAT IS CYGWIN?

RE: <https://www.cygwin.com/>

<https://physionet.org/physiotools/cygwin/>

<https://en.wikipedia.org/wiki/Cygwin>

<https://www.howtogeek.com/howto/41382/how-to-use-linux-commands-in-windows-with-cygwin/>

1) Cygwin is free software that provides a Unix-like environment and software tool set to users of any modern x86 32-bit and 64-bit versions of MS-Windows.

2) Cygwin provides native integration of Windows-based applications, data, and other system resources with applications, software tools, and data of the Unix-like environment.

# UNDERSTANDING OS, BASH, AND CSH

Class – this article will give you a good overview of operating systems, BASH, and CSH.

<http://www.differencebetween.net/technology/software-technology/difference-between-csh-and-bash/>

# UNDERSTANDING ENVIRONMENT VARIABLES

1) Environment variables are OS dependent. I will talk about MAC and Windows 7. However, you can search the Internet if you have a different OS.

2) Environment variables in Windows 7: command line

Environment variables are not often seen directly when using Windows. However there are cases, especially when using the command line, that setting and updating environment variables is a necessity.

In windows from cmd, you can type “set”. This will show all the current environment variables....

See next slide for image...

# “SET” IN WINDOWS CMD

```
C:\Windows\system32\cmd.exe

C:\Users\Ami\Documents>set
ALLUSERSPROFILE=C:\ProgramData
APPDATA=C:\Users\Ami\AppData\Roaming
ATISTREAMSDKROOT=C:\Program Files (x86)\ATI Stream\
CommonProgramFiles=C:\Program Files\Common Files
CommonProgramFiles(x86)=C:\Program Files (x86)\Common Files
CommonProgramW6432=C:\Program Files\Common Files
COMPUTERNAME=AMI-UAIO
ComSpec=C:\Windows\system32\cmd.exe
configsetroot=C:\Windows\ConfigSetRoot
EMC_AUTOPLAY=C:\Program Files (x86)\Common Files\Roxio Shared\
FP_NO_HOST_CHECK=NO
HOMEDRIVE=C:
HOMEPATH=\Users\Ami
LOCALAPPDATA=C:\Users\Ami\AppData\Local
LOGONSERVER=\\AMI-UAIO
MpConfig_ProductAppDataPath=C:\ProgramData\Microsoft\Windows Defender
MpConfig_ProductCodeName=AntiSpyware
MpConfig_ProductPath=C:\Program Files\Windows Defender
MpConfig_ProductUserAppDataPath=C:\Users\Ami\AppData\Local\Microsoft\Windows Def
ender
MpConfig_ReportingGUID=4AE70FC6-D8C7-463F-BC3A-0F70B32BAF62
NUMBER_OF_PROCESSORS=4
OS=Windows_NT
Path=C:\ProgramData\Oracle\Java\javapath;C:\Program Files\Common Files\Microsoft
Shared\Windows Live;C:\Program Files (x86)\Common Files\Microsoft Shared\Window
s Live;C:\Program Files (x86)\ATI Stream\bin\x86_64;C:\Program Files (x86)\ATI S
tream\bin\x86;C:\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\Windows
\System32\WindowsPowerShell\v1.0\;C:\Program Files (x86)\ATI Technologies\ATI.AC
E\Core-Static;C:\Program Files\Intel\WiFi\bin\;C:\Program Files\Common Files\Int
el\WirelessCommon\;C:\Program Files (x86)\Intel\Services\IPT\;C:\Program Files (
```

# MORE ON ENVIRONMENT VARIABLES

In a *MAC*, the method for accessing the current environment variables is similar.

You will have to access the “terminal”.

From there, I believe that “printenv” will show a list of all the current environment variables.

NOTE: I do not have a *MAC* – so you may have to double-check this.

# SETTING PATH ENVIRONMENT VARIABLES

Class – changing environment variables should not be done lightly.

It is always a good idea to understand what you are doing and why you are doing it.

It is also a good idea to review several web pages on the topic.

Here is a link to changing PATH variables in Windows:

<https://www.nextofwindows.com/how-to-addedit-environment-variables-in-windows-7>

Here is a link for changing PATH variables on a MAC:

<http://hathaway.cc/post/69201163472/how-to-edit-your-path-environment-variables-on-mac>

# REGULAR EXPRESSIONS

Regular expressions (re) can be used in R, in Python, and on the command line.

Regular expressions are an excellent methods to search through text, to find or identify things that you are looking for, and to grab, alter, or update text.

R and Python have their own versions of regular expressions. The Twitter example in Python offers a very good example of regular expressions in Python.

Regular expressions deal with “patterns” that allow one to search through text and to take certain actions.



# GREP AND EGREP

References:

<http://ryanstutorials.net/linuxtutorial/grep.php>

<https://www.digitalocean.com/community/tutorials/using-grep-regular-expressions-to-search-for-text-patterns-in-linux>

One of the most useful and versatile commands in a Linux/Unix terminal environment is the "grep" command.

The name "grep" stands for "**global regular expression print**".

This means that grep can be used to see if the input it receives **matches a specified pattern**.

## GREP BASIC EXAMPLE

```
grep -i "license" GPL-3
```

```

                                GNU GENERAL PUBLIC LICENSE
of this license document, but changing it is not allowed.
The GNU General Public License is a free, copyleft license for
The licenses for most software and other practical works are designed
the GNU General Public License is intended to guarantee your freedom to
GNU General Public License for most of our software; it applies also to
price. Our General Public Licenses are designed to make sure that you
(1) assert copyright on the software, and (2) offer you this License
    "This License" refers to version 3 of the GNU General Public License.
    "The Program" refers to any copyrightable work licensed under this
...
...
```

As you can see, we have been given results that contain: "LICENSE", "license", and "License". If there was an instance with "LiCeNsE", that would have been returned as well.

# REGULAR EXPRESSIONS HAVE RULES

Each language has its own rules and symbols for using regular expressions to match patterns in text.

These sites offer several regular expression examples for grep and for unix/linux.

<https://www.cyberciti.biz/faq/grep-regular-expressions/>

<http://www.robelle.com/smugbook/regexpr.html>

[https://www.gnu.org/software/findutils/manual/html\\_node/find\\_html/egrep-regular-expression-syntax.html](https://www.gnu.org/software/findutils/manual/html_node/find_html/egrep-regular-expression-syntax.html)

Again – remember that most languages, including R and Python, offer regular expressions for pattern matching. Each has its own “rules” and methods.

# EGREP REFERENCE

The following site explains grep and egrep

<http://ryanstutorials.net/linuxtutorial/grep.php>