Basic commands

Karen Cristine Goncalves

2023-01-26

Basics

- Commands are separated by a new line (Enter, Return) or a semicolon (;)

```
ls --help
man ls
```

Basics

- Commands are separated by a new line (Enter, Return) or a semicolon (;)
- The first word in a command is what you are asking the computer to do (a function)

```
ls --help
man ls
```

Basics

- Commands are separated by a new line (Enter, Return) or a semicolon (;)
- The first word in a command is what you are asking the computer to do (a function)
- Spaces are used to separate file names, commands, etc.

```
ls --help
man ls
```

Basics

- Commands are separated by a new line (Enter, Return) or a semicolon (;)
- The first word in a command is what you are asking the computer to do (a function)
- Spaces are used to separate file names, commands, etc.
- Some commands allow you to customize their output with options

```
ls --help
man ls
```

Basics

- Commands are separated by a new line (Enter, Return) or a semicolon (;)
- The first word in a command is what you are asking the computer to do (a function)
- Spaces are used to separate file names, commands, etc.
- Some commands allow you to customize their output with options
  - These are single letters, a word, or several words preceded by - or --

```
ls --help
man ls
```

Basics

- Commands are separated by a new line (Enter, Return) or a semicolon (;)
- The first word in a command is what you are asking the computer to do (a function)
- Spaces are used to separate file names, commands, etc.
- Some commands allow you to customize their output with options
  - These are single letters, a word, or several words preceded by - or --
- You can get help for a command by using the help option (either `-h` , `--help`) or the command `man` (for manual)

```
ls --help
man ls
```

About spaces and file names

- Put name in quotes if it has spaces (test code below)

```
touch Programming Class.txt
ls

touch "Programming Class.txt"
ls
```

```
echo "This is good"
echo 'This is good'
```

About spaces and file names

- Put name in quotes if it has spaces (test code below)

```
touch Programming Class.txt
ls

touch "Programming Class.txt"
ls
```

- Quotes are NORMALLY (not always) interchangeable (test code below)

```
echo "This is good"
echo 'This is good'
```

About spaces and file names

- Put name in quotes if it has spaces (test code below)

```
touch Programming Class.txt
ls

touch "Programming Class.txt"
ls
```

- Quotes are NORMALLY (not always) interchangeable (test code below)

```
echo "This is good"
echo 'This is good'
```

Shortcuts

- `~` or `$HOME` : your home folder (can be defined by the user)

Shortcuts

- `~` or `$HOME` : your home folder (can be defined by the user)
- `.` : the folder that you are currently in

Shortcuts

- `~` or `$HOME` : your home folder (can be defined by the user)
- `.` : the folder that you are currently in
- `..`: the folder that contains the one you are currently in

Shortcuts

- `~` or `$HOME` : your home folder (can be defined by the user)
- `.` : the folder that you are currently in
- `..`: the folder that contains the one you are currently in
- `Ctrl+C`: cancel a command

Shortcuts

- `~` or `$HOME` : your home folder (can be defined by the user)
- `.` : the folder that you are currently in
- `..`: the folder that contains the one you are currently in
- `Ctrl+C`: cancel a command
- In MobaxTerm, find and modify useful shortcuts by clicking on Settings -> Keyboard shortcuts

Shortcuts

- `whoami`

Shortcuts

- `whoami`
  - prints your username (if saved in the computer)

Shortcuts

- `whoami`
  - prints your username (if saved in the computer)
- Use `tab` to complete words

Shortcuts

- `whoami`
  - prints your username (if saved in the computer)
- Use `tab` to complete words
- In a current command or in a text file, move the cursor faster by using `Ctrl+Arrow` (right or left arrow in a command line, in a text file up and down arrow too) (this works everywhere!!!)

Basic commands

- `cd`

Basic commands

- `cd`
  - acronym for “change directory” (directory = folder)

Basic commands

- `cd`
  - acronym for “change directory” (directory = folder)
  - If used alone, opens your home folder

Basic commands

- `cd`
  - acronym for “change directory” (directory = folder)
  - If used alone, opens your home folder
  - The name of the folder to which you want to go comes after `cd`

Basic commands

- `cd`
  - acronym for "change directory" (directory = folder)
  - If used alone, opens your home folder
  - The name of the folder to which you want to go comes after `cd`
    - `cd` , `cd ~` and `cd $HOME` are synonyms

Basic commands

- `cd`
  - acronym for "change directory" (directory = folder)
  - If used alone, opens your home folder
  - The name of the folder to which you want to go comes after `cd`
    - `cd` , `cd ~` and `cd $HOME` are synonyms
  - `cd -` - takes you to the previous folder

Basic commands

- `pwd`

Basic commands

- `pwd`
  - acronym for “print working directory” (directory = folder)

Basic commands

- `pwd`
  - acronym for "print working directory" (directory = folder)
  - equivalent to the R function `getwd()` or python's `os.getcwd()`

Basic commands

- `pwd`
  - acronym for "print working directory" (directory = folder)
  - equivalent to the R function `getwd()` or python's `os.getcwd()`
  - prints the full path to your current folder

Basic commands

- `pwd`
  - acronym for "print working directory" (directory = folder)
  - equivalent to the R function `getwd()` or python's `os.getcwd()`
  - prints the full path to your current folder
  - A full path always starts from the root (/)

Basic commands

- `ls`

```
pwd > myFolder
cat myFolder
```

```
ls >> myFolder
cat myFolder
```

Basic commands

- `ls`
  - lists the contents of your current folder

```
pwd > myFolder
cat myFolder
```

```
ls >> myFolder
cat myFolder
```

Basic commands

- `ls`
  - lists the contents of your current folder
  - Check slide 2 where we used this command

```
pwd > myFolder
cat myFolder
```

```
ls >> myFolder
cat myFolder
```

Basic commands

- `ls`
  - lists the contents of your current folder
  - Check slide 2 where we used this command
- Use > after a command to save the output

```
pwd > myFolder
cat myFolder
```

```
ls >> myFolder
cat myFolder
```

Basic commands

- `ls`
  - lists the contents of your current folder
  - Check slide 2 where we used this command
- Use > after a command to save the output

```
pwd > myFolder
cat myFolder
```

- Use >> to add the current output to a previous file

```
ls >> myFolder
cat myFolder
```

Managing text files

- `cat` : prints the contents of the file to the screen (check slide 5)

Managing text files

- `cat` : prints the contents of the file to the screen (check slide 5)
  - Do not use it with files are that not text (images, pdfs, compressed files) or is too big

Managing text files

- `cat` : prints the contents of the file to the screen (check slide 5)
  - Do not use it with files are that not text (images, pdfs, compressed files) or is too big
  - If several files names are put after the command, one file is printed followed by the next (conCATenation)

Managing text files

- `cat` : prints the contents of the file to the screen (check slide 5)
  - Do not use it with files are that not text (images, pdfs, compressed files) or is too big
  - If several files names are put after the command, one file is printed followed by the next (conCATenation)
- `head` and `tail`: prints to the screen the first/last 10 lines of the file

Managing text files

- `less` : opens the file as “read-only”

Managing text files

- `less` : opens the file as “read-only”
  - Search for a word in a file inside less by typing “/” followed by the word

Managing text files

- `less` : opens the file as “read-only”
  - Search for a word in a file inside less by typing “/” followed by the word
  - To close less, press Q

Managing text files

- `less` : opens the file as “read-only”
  - Search for a word in a file inside less by typing “/” followed by the word
  - To close less, press Q
- `more` : opens the file as “read-only”, when the file is closed, prints it to the screen

Managing text files

- `nano` : open a text file to edit it.

Managing text files

- `nano` : open a text file to edit it.
- `grep` : searches for a word/phrase in the file and prints the lines that match

Managing text files

- `nano` : open a text file to edit it.
- `grep` : searches for a word/phrase in the file and prints the lines that match
  - Can search for several phrases (one per line) in a file by using the option `-f`

Managing text files

- `nano` : open a text file to edit it.
- `grep` : searches for a word/phrase in the file and prints the lines that match
  - Can search for several phrases (one per line) in a file by using the option `-f`
  - If you don't care about the upper/lower case, use the option `-i` or `--ignore-case`

Managing text files

Search for lines that have the word programming in the file myFolder created in slide7
`grep "Programming" myFolder`

- `wc` : word count. Counts the number of characters, words and lines in a file

Managing text files

Search for lines that have the word programming in the file myFolder created in slide7
`grep "Programming" myFolder`

- `wc` : word count. Counts the number of characters, words and lines in a file
- `echo` : repeats the text that follows it (check slide 2)

Exercise

- Save a fasta file into your home folder with the name myFasta.fa

Exercise

- Save a fasta file into your home folder with the name myFasta.fa
- Use `grep` to find all the lines with sequence IDs.

Exercise

- Save a fasta file into your home folder with the name myFasta.fa
- Use `grep` to find all the lines with sequence IDs.
  - Note - put the word or phrase you will search for inside ""

Exercise - Solution

- Save a fasta file into your home folder with the name myFasta.fa

In all fasta files, the sequence ID line is indicated by the symbol >, so we just need to look for it

```
grep ">" myFasta.fa
```

Exercise - Solution

- Save a fasta file into your home folder with the name myFasta.fa
- Use `grep` to find all the lines with sequence IDs.

In all fasta files, the sequence ID line is indicated by the symbol >, so we just need to look for it

```
grep ">" myFasta.fa
```

Exercise - Solution

- Save a fasta file into your home folder with the name myFasta.fa
- Use `grep` to find all the lines with sequence IDs.
  - Note - put the word or phrase you will search for inside `""`

In all fasta files, the sequence ID line is indicated by the symbol >, so we just need to look for it

```
grep ">" myFasta.fa
```

Exercise - Solution

- Save a fasta file into your home folder with the name myFasta.fa
- Use `grep` to find all the lines with sequence IDs.
  - Note - put the word or phrase you will search for inside `""`

In all fasta files, the sequence ID line is indicated by the symbol >, so we just need to look for it

```
grep ">" myFasta.fa
```

- Normally, if you just search for one word, the quotes are not needed, by in this case, the symbol `">"` could also mean "send the output to", which would replace the myFasta.fa file

Managing files

- `cp` : acronym for copy

Managing files

- `cp` : acronym for copy
  - `cp file file2` : creates a copy of the file “file” and saves it in the file “file2”

Managing files

- cp : acronym for copy
  - `cp file file2` : creates a copy of the file "file" and saves it in the file "file2"
  - `cp file folder` : creates a copy of the file "file" and saves it with the same name in the folder "folder"

Managing files

- cp : acronym for copy
  - `cp file file2` : creates a copy of the file "file" and saves it in the file "file2"
  - `cp file folder` : creates a copy of the file "file" and saves it with the same name in the folder "folder"
  - `cp file file2 folder`: both files "file" and "file2" are copied into the folder "folder" with the same names

Managing files

- cp : acronym for copy
  - `cp file file2` : creates a copy of the file “file” and saves it in the file “file2”
  - `cp file folder` : creates a copy of the file “file” and saves it with the same name in the folder “folder”
  - `cp file file2 folder`: both files “file” and “file2” are copied into the folder “folder” with the same names
  - `cp folder folder2 -r` : the option -r allows the copy of the entire folder.

Managing files

- cp : acronym for copy
  - `cp file file2` : creates a copy of the file "file" and saves it in the file "file2"
  - `cp file folder` : creates a copy of the file "file" and saves it with the same name in the folder "folder"
  - `cp file file2 folder`: both files "file" and "file2" are copied into the folder "folder" with the same names
  - `cp folder folder2 -r` : the option -r allows the copy of the entire folder.
    - If folder2 doesn't exist, it will be created to hold the same files as "folder"

Managing files

- cp : acronym for copy
  - `cp file file2` : creates a copy of the file "file" and saves it in the file "file2"
  - `cp file folder` : creates a copy of the file "file" and saves it with the same name in the folder "folder"
  - `cp file file2 folder`: both files "file" and "file2" are copied into the folder "folder" with the same names
  - `cp folder folder2 -r` : the option -r allows the copy of the entire folder.
    - If folder2 doesn't exist, it will be created to hold the same files as "folder"
    - If folder2 exists, a copy of "folder" will be created inside of folder2

Managing files

- `mv` : acronym for move. Move file from one place to another

Managing files

- `mv` : acronym for move. Move file from one place to another
  - `mv file file2` : renames “file” as “file2”

Managing files

- `mv` : acronym for move. Move file from one place to another
  - `mv file file2` : renames “file” as “file2”
  - `mv file folder` : moves “file” into the folder “folder”

Managing files

- `mv` : acronym for move. Move file from one place to another
  - `mv file file2` : renames “file” as “file2”
  - `mv file folder` : moves “file” into the folder “folder”
  - `mv file file2 folder`: both files “file” and “file2” are moved into the folder “folder” with the same names

Managing files

- `mv` : acronym for move. Move file from one place to another
  - `mv file file2` : renames “file” as “file2”
  - `mv file folder` : moves “file” into the folder “folder”
  - `mv file file2 folder`: both files “file” and “file2” are moved into the folder “folder” with the same names
  - `mv folder folder2` :

Managing files

- `mv` : acronym for move. Move file from one place to another
  - `mv file file2` : renames “file” as “file2”
  - `mv file folder` : moves “file” into the folder “folder”
  - `mv file file2 folder`: both files “file” and “file2” are moved into the folder “folder” with the same names
  - `mv folder folder2` :
    - If folder2 doesn't exist, it is the same as renaming “folder” as “folder2”

Managing files

- `mv` : acronym for move. Move file from one place to another
  - `mv file file2` : renames “file” as “file2”
  - `mv file folder` : moves “file” into the folder “folder”
  - `mv file file2 folder`: both files “file” and “file2” are moved into the folder “folder” with the same names
  - `mv folder folder2` :
    - If folder2 doesn't exist, it is the same as renaming “folder” as “folder2”
    - If folder2 exists, “folder” is moved to that folder

Managing files

- `rm` : acronym for remove. Deletes files. **They are PERMANENTELY deleted, there is no trash bin here!!!!!!!**

Managing files

- `rm` : acronym for remove. Deletes files. **They are PERMANENTELY deleted, there is no trash bin here!!!!!!!**
  - `rm file file2` : deletes both files

Managing files

- `rm` : acronym for remove. Deletes files. **They are PERMANENTELY deleted, there is no trash bin here!!!!!!!**
  - `rm file file2` : deletes both files
  - `rm file file2 -i` : asks the user if they really want to delete each file, if `y` is pressed, the file is deleted (`-i` for interactive)

Managing folders

- `mkdir` : acronym for “make directory”. Creates new folders with the specified names.

Managing folders

- `mkdir` : acronym for “make directory”. Creates new folders with the specified names.
  - Gives an error if something with the same name already exists in the current folder.

Managing folders

- `mkdir` : acronym for “make directory”. Creates new folders with the specified names.
  - Gives an error if something with the same name already exists in the current folder.
  - If many names are given (separated by spaces, creates all names folders)

Managing folders

- `mkdir` : acronym for "make directory". Creates new folders with the specified names.
  - Gives an error if something with the same name already exists in the current folder.
  - If many names are given (separated by spaces, creates all names folders)
- `rmdir` : acronym for "remove directory". Deletes ***empty*** folders.

Managing folders

- `mkdir` : acronym for "make directory". Creates new folders with the specified names.
  - Gives an error if something with the same name already exists in the current folder.
  - If many names are given (separated by spaces, creates all names folders)
- `rmdir` : acronym for "remove directory". Deletes ***empty*** folders.
  - If the folder is not empty, gives an error.

Managing folders

- `mkdir` : acronym for "make directory". Creates new folders with the specified names.
  - Gives an error if something with the same name already exists in the current folder.
  - If many names are given (separated by spaces, creates all names folders)
- `rmdir` : acronym for "remove directory". Deletes ***empty*** folders.
  - If the folder is not empty, gives an error.
  - If many names are given (separated by spaces, deletes all names folders **if they are empty**

Managing folders

- `mkdir` : acronym for "make directory". Creates new folders with the specified names.
  - Gives an error if something with the same name already exists in the current folder.
  - If many names are given (separated by spaces, creates all names folders)
- `rmdir` : acronym for "remove directory". Deletes ***empty*** folders.
  - If the folder is not empty, gives an error.
  - If many names are given (separated by spaces, deletes all names folders **if they are empty**
- `rm -r folder` : as in `cp`, the option `-r` allows the `rm` to work with a folder. It deletes everything in the folder, then deletes the folder itself

Resources for help

Glossary of commands

Book on basic unix commands