



UNIVERSIDAD DE COSTA RICA

INTRODUCCIÓN AL SHELL DE LINUX: BASH

Prof. Marlon Brenes y Prof. Federico Muñoz
Escuela de Física, Universidad de Costa Rica

Qué es un shell?

- Un shell es un “super”-programa
- Su propósito es proveer la habilidad de ejecutar otros programas
- Todas las computadoras en general poseen algún tipo de shell

```
Last login: Tue Jan 30 13:50:45 on console  
[mbrenes@hxckid:~$ sn  
Last login: Thu Jan 25 14:22:34 2024 from 100.64.47.228  
=====  
SciNet welcomes you to the NIAGARA supercomputer.  
  
This is a Niagara login node. Use this node to develop and compile code,  
to run short tests, and to submit computations to the scheduler.  
  
Remember that /scratch is never backed-up.  
  
Documentation: https://docs.scinet.utoronto.ca/index.php/Niagara\_Quickstart  
Support: support@scinet.utoronto.ca or niagara@tech.alliancecan.ca  
=====  
* Oct 11 2023: Optional multifactor authentication available  
  
Multifactor authentication is now available to all users. Please  
help us secure Canada's digital research by activating it for your  
account. More details available here:  
https://docs.alliancecan.ca/wiki/Multifactor\_authentication  
  
If you need unattended connections, please contact our technical  
support at support@tech.alliancecan.ca before activating multifactor  
authentication.  
=====  
Welcome mbrenes, your access to this system has been logged.  
If you are not mbrenes, please disconnect immediately.  
=====  
* Scratch Purging Warning (2024-02-01) *  
  
Some of your files on $SCRATCH are slated for purging. Purging  
starts on the 15th of February and can take a few days to finish.  
The list of these files can be found in the file:  
  
/scratch/t/todelete/current/3114286_mbrenes_dvira_641.38K_7files  
[mbrenes@nia-login03:~$ pwd  
/home/d/dvira/mbrenes
```

Acerca de interfaces

- Es una mala decisión utilizar interfaces de usuario gráficas (GUI, por sus siglas en inglés) para ambientes de computación científica de alto rendimiento
- Este no es necesariamente el caso en programación de alto nivel científico, e.g., Matlab, Mathematica, entre otros.
- Razones
 - Computación científica de alto rendimiento se refiere a aplicaciones diseñadas para operar en arquitecturas de supercómputo (*clusters*)
 - El acceso a dichas computadoras ocurre de manera remota
 - Mantener GUIs para cada usuario resulta en condiciones poco óptimas para el usuario. Las GUIs son **lentas**
 - Esto implica que en computación de alto rendimiento, es necesario conocer la interfaz adaptada para el sistema

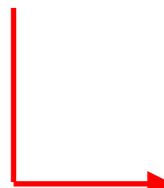
GUI vs CLI

- *Graphical User Interfaces (GUI)*



- Diseñadas para utilizar funcionalidad ya existente, controles ya existentes
- Mucha funcionalidad, difícil realizar extensiones
- Difícil de utilizar comandos en serie para replicar funcionalidad
- Fácil de aprender. Difícil de utilizar para tareas complejas

- *Command Line Interfaces (CLI)*



- Canvas en blanco. Se programa directamente la tarea que se desea completar
- Perfectas para crear y diseñar tareas nuevas
- Los comandos existentes suelen ser buenos para realizar **una** tarea
- Complicadas de aprender. Facilidad para tareas complejas

BASH

- El shell mas común en Linux es *BASH (Bourne Again Shell)*



- El shell provee acceso a archivos, redes y programas
- Existen otros shells (zsh, ksh, csh...). Su uso es el mismo, la sintaxis y funcionalidad pueden cambiar
- El shell es el ente con el cual se interactúa cuando se escriben comandos

- El shell se accesa mediante una terminal (xterm, eterm, konsole...)



- En la terminal se escriben los comandos
- El shell los interpreta
- El shell realiza acciones bajo su propia cuenta o ejecuta otros programas

BASH

```
[idp] mbrenes@nia-login03:~$ pwd
/home/d/dvira(mbrenes
[idp] mbrenes@nia-login03:~$ which bash
/usr/bin/bash
[idp] mbrenes@nia-login03:~$ cat .bash_profile
# .bash_profile

# User specific environment and startup programs

export PS1="\[\033[32m\]\u\[\033[m\]@\[\033[33m\]\h:\[\033[37m\]\W\[\033[m\]\$ "
export PETSC_DIR=/home/d/dvira(mbrenes/petsc-3.16.0
export PETSC_ARCH=arch-complex
export SLEPC_DIR=/home/d/dvira(mbrenes/slepc-3.16.0
export BOOST_DIR=/home/d/dvira(mbrenes/boost_1_77_0

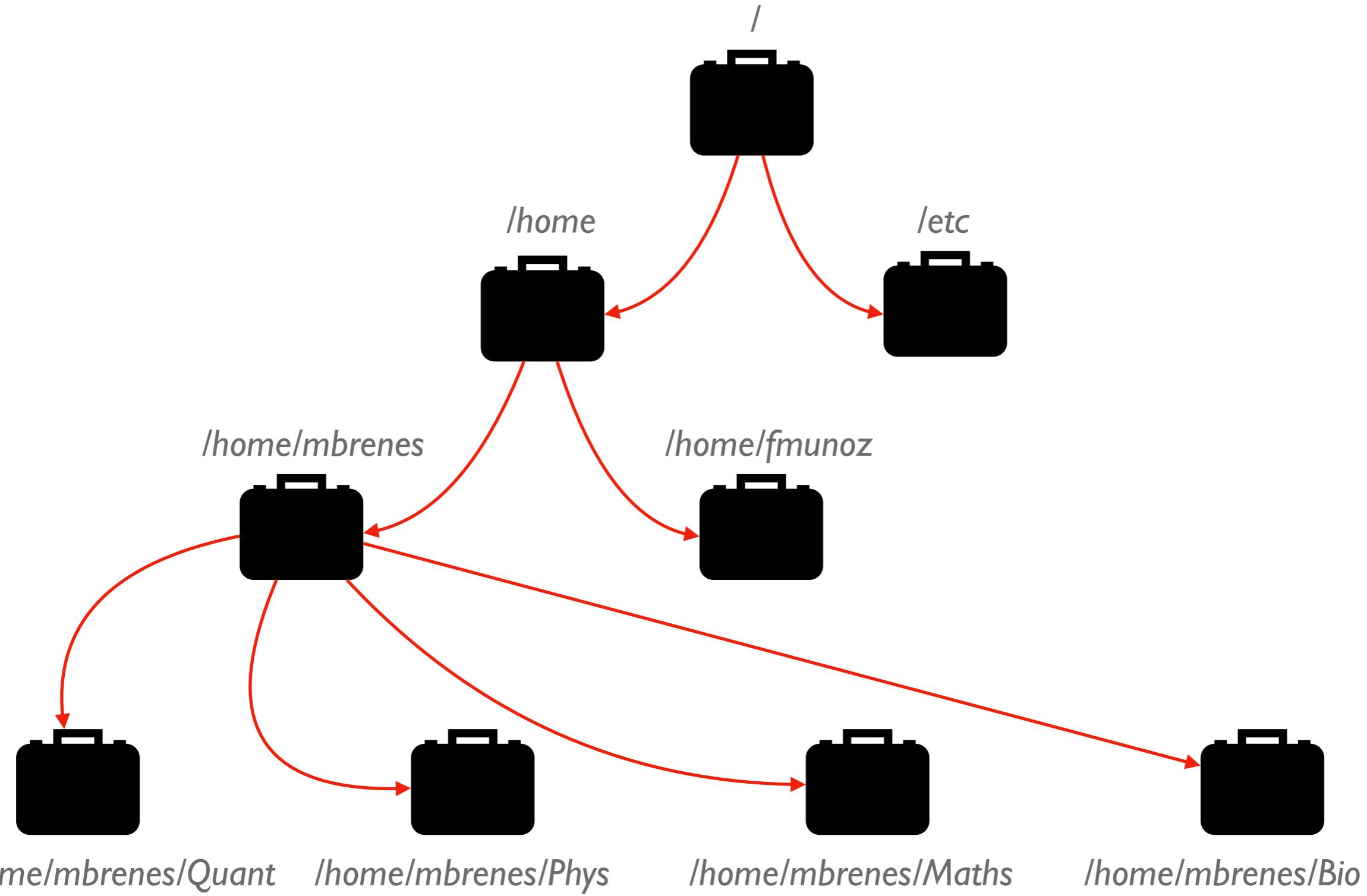
export XDG_RUNTIME_DIR=/tmp/runtime-mbrenes
export LIBGL_ALWAYS_INDIRECT=1
(idp) mbrenes@nia-login03:~$
```

Hola mundo!

```
[mbrenes@nia-login03:~$  
[mbrenes@nia-login03:~$ hello="world"  
[mbrenes@nia-login03:~$ echo Hello, world  
Hello, world  
[mbrenes@nia-login03:~$ echo Hello, $hello  
Hello, world  
[mbrenes@nia-login03:~$ ]
```

- El símbolo “=” indica **asignación**, no equivalencia
- “=” indica al shell que cree una variable llamada ‘hello’ con valor ‘world’. La variable se accesa con el símbolo \$
- echo es una función primitiva de bash que imprime dependiendo de su argumento

Filesystem



Bash básico: filesystem

```
[mbrenes@nia-login06:~$ clear  
[mbrenes@nia-login06:~$ pwd  
/home/d/dvira/mbrenes  
[mbrenes@nia-login06:~$ ls  
boost_1_77_0          ED          HEOM      job_omp.sh  pennylane-lightning  PovM-Quantum-Simulation  Redfield  
BoundaryDriving-PovM-Transformer EntanglementStabilisation job_mpi.sh  job.sh     petsc-3.16.0        QuDyn  
[mbrenes@nia-login06:~$ ls ~/QuDyn/  
drivers  job.sh  LICENSE  Makefile  obj  QuDyn.x  repl  src  
mbrenes@nia-login06:~$ ]
```

Our commands

echo **arg**

echo the argument

pwd

present working directory

ls [**dir**]

list the directory contents

arg

mandatory argument

[**arg**]

optional argument

pwd significa *print working directory*

ls proviene de *list*

Los datos de trabajo de hoy

```
[mbrenes@nia-login06:workspace$ ls
[mbrenes@nia-login06:workspace$ git clone https://github.com/mbrenesn/FisicaComputacional.git
Cloning into 'FisicaComputacional'...
remote: Enumerating objects: 10, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 10 (delta 1), reused 4 (delta 0), pack-reused 0
Receiving objects: 100% (10/10), done.
Resolving deltas: 100% (1/1), done.
[mbrenes@nia-login06:workspace$ ls
FisicaComputacional
[mbrenes@nia-login06:workspace$ cd FisicaComputacional/Semana_01/
[mbrenes@nia-login06:Semana_01$ ls
commandline.tar.gz
mbrenes@nia-login06:Semana_01$ ]
```

cd significa *change directory*

Los datos de trabajo de hoy

```
[mbrenes@nia-login06:Semana_01$ ls
commandline.tar.gz
[mbrenes@nia-login06:Semana_01$ tar -z -x -f commandline.tar.gz
[mbrenes@nia-login06:Semana_01$ ls
commandline.tar.gz  data
[mbrenes@nia-login06:Semana_01$ ls data/
alexander  Bert  Frank_Richard  gerdal  jamesm  Lawrence  THOMAS
mbrenes@nia-login06:Semana_01$ ]
```

- `tar` es una función primitiva de bash cuya función es compresión/descompresión de archivos en formato *tarball*. Un *tarball* es un conjunto de archivos agrupados para su fácil manejo



- `-z` indica compresión en formato gunzip
- `-x` indica extraer los archivos
- `-f` continuado del nombre del tarball indica el archivo

Directarios

```
[mbrenes@nia-login06:Semana_01$ ls
commandline.tar.gz  data
[mbrenes@nia-login06:Semana_01$ pwd
/home/d/dvira/mbrenes/workspace/FisicaComputacional/Semana_01
[mbrenes@nia-login06:Semana_01$ mkdir firstdir
[mbrenes@nia-login06:Semana_01$ ls
commandline.tar.gz  data  firstdir
[mbrenes@nia-login06:Semana_01$ cd ~/workspace/
[mbrenes@nia-login06:workspace$ ls
FisicaComputacional
[mbrenes@nia-login06:workspace$ pwd
/home/d/dvira/mbrenes/workspace
mbrenes@nia-login06:workspace$
```

Our commands

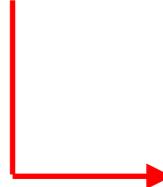
echo arg	echo the argument
pwd	present working directory
ls [dir]	list the directory contents
mkdir dir	create a directory
cd [dir]	change directory
arg	mandatory argument
[arg]	optional argument

mkdir significa *make directory*

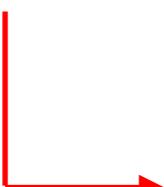


- El símbolo ~ funciona para denotar el directorio ‘home’
- mkdir crea nuevos directorios
- cd se utiliza para moverse a través de distintos directorios

Tips

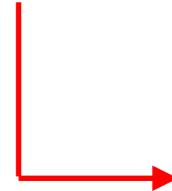


- ‘cd ..’ nos lleva un directorio hacia arriba
- ‘cd ../../’ nos lleva dos directorios hacia arriba
- ‘cd ../otherdir’ nos lleva al directorio otherdir ubicado en el directorio anterior
- ‘cd firstdir/seconddir/../../’ nos deja en la misma posición
- En general, ‘.’ significa ‘este directorio’, mientras que ‘..’ significa el directorio anterior

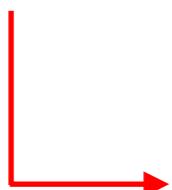


- Se pueden usar direcciones absolutas: ‘cd /usr/src/kernels’
- ‘cd’ nos lleva a ~
- ‘cd -’ nos lleva al directorio accesado anterior al actual (muy útil)

Entradas de teclado

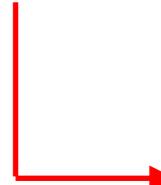


- **Tab** completa las opciones dependiendo del comando (extremadamente útil)
- **Arriba y abajo** rotan los comandos usados anteriormente
- **ctrl-a** nos lleva al inicio del comando
- **ctrl-e** nos lleva al final del comando
- **ctrl-r** recibe líneas de comando y las compara con comandos usados anteriormente (extremadamente útil)



- ‘history’ es un comando primitivo que enlista los comandos utilizados en el pasado

Páginas man



- **man** provee la funcionalidad de un comando
- La mayoría de programas tienen una página en man que detalla los parámetros de uso y funcionalidad

```
TAR(1)                               User Commands                               TAR(1)

NAME
    tar - manual page for tar 1.26

SYNOPSIS
    tar [OPTION...] [FILE]...

DESCRIPTION
    GNU `tar' saves many files together into a single tape or disk archive, and can restore individual files from the archive.

    Note that this manual page contains just very brief description (or more like a list of possible functionality) originally generated by the help2man utility. The full documentation for tar is maintained as a Texinfo manual. If the info and tar programs are properly installed at your site, the command `info tar' should give you access to the complete manual.

EXAMPLES
    tar -cf archive.tar foo bar
        # Create archive.tar from files foo and bar.

    tar -tvf archive.tar
        # List all files in archive.tar verbosely.

    tar -xf archive.tar
        # Extract all files from archive.tar.

DEFAULTS
    *This* tar installation defaults to:

    --format=gnu -f- -b20 --quoting-style=escape --rmt-command=/sbin/rmt --rsh-command=/usr/bin/rsh

Main operation mode:
    -A, --catenate, --concatenate
```

Wildcards * (extremadamente útil)

- • El símbolo '*' es conocido como un *wildcard*, cuya utilidad es capturar **todas** las posibles combinaciones que calzan con cierta descripción

```
[mbrenes@nia-login06:Semana_01$ pwd  
/home/d/dvira/mbrenes/workspace/FisicaComputacional/Semana_01  
[mbrenes@nia-login06:Semana_01$ cd data/  
[mbrenes@nia-login06:data$ ls  
alexander Bert Frank_Richard gerdal jamesm Lawrence THOMAS  
[mbrenes@nia-login06:data$ ls -d *er*  
alexander Bert gerdal  
[mbrenes@nia-login06:data$ ls -d *e  
Lawrence
```

- • Actúa de manera *iterativa*
- • El shell expande todas las posibilidades y alimenta estos argumentos al comando

Manipulación de archivos

```
[mbrenes@login06:FisicaComputacional$ ls  
README.md  Semana_01  
[mbrenes@login06:FisicaComputacional$ mkdir test  
[mbrenes@login06:FisicaComputacional$ ls  
README.md  Semana_01  test  
[mbrenes@login06:FisicaComputacional$ cp Semana_01/data/alexander/data_216.DATA .  
[mbrenes@login06:FisicaComputacional$ ls  
data_216.DATA README.md Semana_01  test  
[mbrenes@login06:FisicaComputacional$ mv data_216.DATA test/  
[mbrenes@login06:FisicaComputacional$ ls test/  
data_216.DATA  
[mbrenes@login06:FisicaComputacional$ ]
```

Our commands	
echo <i>arg</i>	echo the argument
pwd	present working directory
ls [<i>dir</i>]	list the directory contents
mkdir <i>dir</i>	create a directory
cd [<i>dir</i>]	change directory
history [<i>num</i>]	print the shell history
man <i>cmd</i>	command's man page
cp <i>file1 file2</i>	copy a file
mv <i>file1 file2</i>	move/rename a file
<i>arg</i>	mandatory argument
[<i>arg</i>]	optional argument

- • ‘mv’ significa move, para mover directorios y archivos
- • ‘cp’ significa copy, para copiar archivos y directorios
 - • Para copiar directorios: ‘cp -r’

Manipulación de archivos

```
[mbrenes@nia-login01:FisicaComputacional$ ls  
README.md Semana_01 test  
[mbrenes@nia-login01:FisicaComputacional$ cp Semana_01/data/alexander/data_216.DATA .  
[mbrenes@nia-login01:FisicaComputacional$ ls  
data_216.DATA README.md Semana_01 test  
[mbrenes@nia-login01:FisicaComputacional$ rm data_216.DATA  
[mbrenes@nia-login01:FisicaComputacional$ ls  
README.md Semana_01 test  
[mbrenes@nia-login01:FisicaComputacional$ ls test/  
data_216.DATA  
[mbrenes@nia-login01:FisicaComputacional$ rm test/  
rm: cannot remove 'test/': Is a directory  
[mbrenes@nia-login01:FisicaComputacional$ rm -rf test/  
[mbrenes@nia-login01:FisicaComputacional$ ls  
README.md Semana_01  
[mbrenes@nia-login01:FisicaComputacional$ ]
```

Our commands

echo arg	echo the argument
pwd	present working directory
ls [dir]	list the directory contents
mkdir dir	create a directory
cd [dir]	change directory
history [num]	print the shell history
man cmd	command's man page
cp file1 file2	copy a file
mv file1 file2	move/rename a file
rm file	delete a file

arg mandatory argument
[arg] optional argument

- • ‘rm’ significa remove, para borrar directorios y archivos
- • **CUIDADO:** Al eliminar un archivo no hay forma de recuperarlo
 - • *If it's gone, it's gone*
 - • Para archivos: ‘f’ fuerza ‘r’ recursivamente la eliminación

Contenido de archivos

```
mbrenes@nia-login01:FisicaComputacional$ ls Semana_01/data/alexander/
data_216.DATA data_288.DATA data_337.DATA data_379.DATA data_420.DATA data_471.D
data_242.DATA data_292.DATA data_339.DATA data_387.DATA data_421.DATA data_473.D
data_256.DATA data_297.DATA data_344.DATA data_389.DATA data_427.DATA data_498.D
data_262.DATA data_305.DATA data_346.DATA data_397.DATA data_434.DATA data_502.D
data_268.DATA data_306.DATA data_347.DATA data_402.DATA data_454.DATA data_516.D
data_277.DATA data_309.DATA data_357.DATA data_408.DATA data_462.DATA data_527.D
data_278.DATA data_318.DATA data_364.DATA data_415.DATA data_469.DATA data_530.D
mbrenes@nia-login01:FisicaComputacional$ more Semana_01/data/alexander/data_216.DATA
#
Reported: Wed Aug 17 13:56:38 2011
Subject: madonnaStarr178
Year/month of birth: 1995/02
Sex: N
CI type: 8
Volume: 7
Range: 3
Discrimination: 5
mbrenes@nia-login01:FisicaComputacional$ cat Semana_01/data/alexander/data_216.DATA
#
Reported: Wed Aug 17 13:56:38 2011
Subject: madonnaStarr178
Year/month of birth: 1995/02
Sex: N
CI type: 8
Volume: 7
Range: 3
Discrimination: 5
mbrenes@nia-login01:FisicaComputacional$
```

Our commands

<code>echo arg</code>	echo the argument
<code>pwd</code>	present working directory
<code>ls [dir]</code>	list the directory contents
<code>mkdir dir</code>	create a directory
<code>cd [dir]</code>	change directory
<code>history [num]</code>	print the shell history
<code>man cmd</code>	command's man page
<code>cp file1 file2</code>	copy a file
<code>mv file1 file2</code>	move/rename a file
<code>rm file</code>	delete a file
<code>rmdir dir</code>	delete a directory
<code>more file</code>	scroll through file
<code>less file</code>	scroll through file
<code>cat file</code>	print the file contents
<code>arg</code>	mandatory argument
<code>[arg]</code>	optional argument

- • ‘more’, ‘less’ y ‘cat’ son comandos para visualizar archivos en la terminal
- • ‘cat’ permite ‘concatenar’ archivos para usar su información como entrada y salida al stdout y stdin

Concatenar archivos

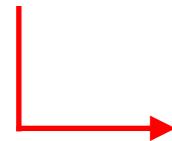
```
mbrenes@nia-login01:FisicaComputacional$ ls Semana_01/data/alexander/
data_216.DATA data_288.DATA data_337.DATA data_379.DATA data_420.DATA data_471.D
data_242.DATA data_292.DATA data_339.DATA data_387.DATA data_421.DATA data_473.D
data_256.DATA data_297.DATA data_344.DATA data_389.DATA data_427.DATA data_498.D
data_262.DATA data_305.DATA data_346.DATA data_397.DATA data_434.DATA data_502.D
data_268.DATA data_306.DATA data_347.DATA data_402.DATA data_454.DATA data_516.D
data_277.DATA data_309.DATA data_357.DATA data_408.DATA data_462.DATA data_527.D
data_278.DATA data_318.DATA data_364.DATA data_415.DATA data_469.DATA data_530.D
mbrenes@nia-login01:FisicaComputacional$ more Semana_01/data/alexander/data_216.DATA
#
Reported: Wed Aug 17 13:56:38 2011
Subject: madonnaStarr178
Year/month of birth: 1995/02
Sex: N
CI type: 8
Volume: 7
Range: 3
Discrimination: 5
mbrenes@nia-login01:FisicaComputacional$ cat Semana_01/data/alexander/data_216.DATA
#
Reported: Wed Aug 17 13:56:38 2011
Subject: madonnaStarr178
Year/month of birth: 1995/02
Sex: N
CI type: 8
Volume: 7
Range: 3
Discrimination: 5
mbrenes@nia-login01:FisicaComputacional$
```

Our commands

<code>echo arg</code>	echo the argument
<code>pwd</code>	present working directory
<code>ls [dir]</code>	list the directory contents
<code>mkdir dir</code>	create a directory
<code>cd [dir]</code>	change directory
<code>history [num]</code>	print the shell history
<code>man cmd</code>	command's man page
<code>cp file1 file2</code>	copy a file
<code>mv file1 file2</code>	move/rename a file
<code>rm file</code>	delete a file
<code>rmdir dir</code>	delete a directory
<code>more file</code>	scroll through file
<code>less file</code>	scroll through file
<code>cat file</code>	print the file contents
<code>arg</code>	mandatory argument
<code>[arg]</code>	optional argument

- • ‘more’, ‘less’ y ‘cat’ son comandos para visualizar archivos en la terminal
- • ‘cat’ permite ‘concatenar’ archivos para usar su información como entrada y salida al stdout y stdin

Redirecciones (extremadamente útil)



- cmd > file: toma la salida que hubiera sido impresa en stdout de terminal, crea un archivo llamado file y redirecciona los contenidos en el nuevo archivo llamado 'file'



- cmd >> file: toma la salida que hubiera sido impresa en stdout de terminal y *adjunta* los contenidos a 'file'. Si 'file' no existe, lo crea



- cmd < file: toma 'file' y usa su contenido como ingreso a 'cmd'

Concatenar archivos: wildcards y redirecciones

```
[mbrenes@nia-login01:FisicaComputacional$ ls
README.md  Semana_01
[mbrenes@nia-login01:FisicaComputacional$ ls Semana_01/data/alexander/
data_216.DATA  data_277.DATA  data_305.DATA  data_339.DATA  data_364.DATA  data_402.DATA  dat
data_242.DATA  data_278.DATA  data_306.DATA  data_344.DATA  data_379.DATA  data_408.DATA  dat
data_256.DATA  data_288.DATA  data_309.DATA  data_346.DATA  data_387.DATA  data_415.DATA  dat
data_262.DATA  data_292.DATA  data_318.DATA  data_347.DATA  data_389.DATA  data_420.DATA  dat
data_268.DATA  data_297.DATA  data_337.DATA  data_357.DATA  data_397.DATA  data_421.DATA  dat
[mbrenes@nia-login01:FisicaComputacional$ cat Semana_01/data/alexander/*.DATA > all_data.DATA
[mbrenes@nia-login01:FisicaComputacional$ head -n 15 all_data.DATA
#
Reported: Wed Aug 17 13:56:38 2011
Subject: madonnaStarr178
Year/month of birth: 1995/02
Sex: N
CI type: 8
Volume: 7
Range: 3
Discrimination: 5
#
Reported: Thu May 19 09:08:14 2011
Subject: paulSpice199
Year/month of birth: 1994/01
Sex: M
CI type: 24
[mbrenes@nia-login01:FisicaComputacional$ tail -n 15 all_data.DATA
Year/month of birth: 1999/05
Sex: F
CI type: 22
Volume: 6
Range: 8
Discrimination: 8
#
Reported: Sat May 7 10:50:03 2011
Subject: georgeSpice437
Year/month of birth: 1997/12
Sex: M
CI type: 20
Volume: 3
Range: 5
Discrimination:
[mbrenes@nia-login01:FisicaComputacional$
```

Our commands

echo arg	echo the argument
pwd	present working directory
ls [dir]	list the directory contents
mkdir dir	create a directory
cd [dir]	change directory
history [num]	print the shell history
man cmd	command's man page
cp file1 file2	copy a file
mv file1 file2	move/rename a file
rm file	delete a file
rmdir dir	delete a directory
more file	scroll through file
less file	scroll through file
cat file	print the file contents
cmd > file	redirect output to file
cmd >> file	append output to file
cmd < file	use file as input to cmd
head file	print first 10 lines of file
tail file	print last 10 lines of file

- ‘head’ y ‘tail’ se pueden utilizar para visualizar el contenido de las primeras/últimas n líneas del archivo

Conteo de objetos en archivos

```
[mbrenes@nia-login01:alexander$ pwd  
/home/d/dvira/mbrenes/workspace/FisicaComputacional/Semana_01/data/alexander  
[mbrenes@nia-login01:alexander$ cat *.DATA > all_data.DATA  
[mbrenes@nia-login01:alexander$ wc all_data.DATA  
 441 1173 7184 all_data.DATA  
[mbrenes@nia-login01:alexander$ wc -l all_data.DATA  
441 all_data.DATA  
[mbrenes@nia-login01:alexander$ wc -w all_data.DATA  
1173 all_data.DATA  
[mbrenes@nia-login01:alexander$ wc -c all_data.DATA  
7184 all_data.DATA  
[mbrenes@nia-login01:alexander$ wc -w *.DATA  
1173 all_data.DATA  
 24 data_216.DATA  
 24 data_242.DATA  
 24 data_536.DATA  
 23 data_542.DATA  
 23 data_546.DATA  
 24 data_547.DATA  
 24 data_548.DATA  
 24 data_550.DATA  
 23 data_560.DATA  
2346 total  
mbrenes@nia-login01:alexander$
```

Our commands

echo <i>arg</i>	echo the argument
pwd	present working directory
ls [<i>dir</i>]	list the directory contents
mkdir <i>dir</i>	create a directory
cd [<i>dir</i>]	change directory
history [<i>num</i>]	print the shell history
man <i>cmd</i>	command's man page
cp <i>file1 file2</i>	copy a file
mv <i>file1 file2</i>	move/rename a file
rm <i>file</i>	delete a file
rmdir <i>dir</i>	delete a directory
more <i>file</i>	scroll through file
less <i>file</i>	scroll through file
cat <i>file</i>	print the file contents
<i>cmd</i> > <i>file</i>	redirect output to file
<i>cmd</i> >> <i>file</i>	append output to file
<i>cmd</i> < <i>file</i>	use file as input to cmd
head <i>file</i>	print first 10 lines of file
tail <i>file</i>	print last 10 lines of file
wc <i>file</i>	word count data of file

- ‘wc’ (*word count*) es una función primitiva que se utiliza para contar caracteres, líneas y palabras

Tipo de archivo y ubicación

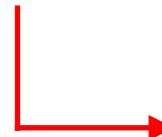
- ‘file’ es un comando para intentar determinar el tipo de archivo en cuestión

```
[mbrenes@nia-login01:alexander$ clear  
  
[mbrenes@nia-login01:alexander$ cd ~/workspace/FisicaComputacional/Semana_01/data/  
[mbrenes@nia-login01:data$ file Lawrence/  
Lawrence/: directory  
[mbrenes@nia-login01:data$ file Lawrence/Data0554  
Lawrence/Data0554: ASCII text  
mbrenes@nia-login01:data$ ]
```

- ‘which’ es una función para identificar la ubicación de cierto comando o programa en ejecución (muy útil cuando existen múltiples instalaciones)

```
[mbrenes@nia-login01:data$ which wget  
/usr/bin/wget  
[mbrenes@nia-login01:data$ file /bin/wget  
/bin/wget: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.32,
```

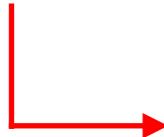
Comparar archivos



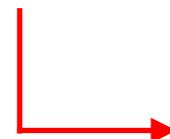
- ‘diff’ es un comando para comparar archivos y ubicar sus diferencias

```
[mbrenes@nia-login02:FisicaComputacional$ ls
README.md  Semana_01
[mbrenes@nia-login02:FisicaComputacional$ cd Semana_01/data/
[mbrenes@nia-login02:data$ diff alexander/data_216.DATA alexander/data_242.DATA
2,9c2,9
< Reported: Wed Aug 17 13:56:38 2011
< Subject: madonnaStarr178
< Year/month of birth: 1995/02
< Sex: N
< CI type: 8
< Volume: 7
< Range: 3
< Discrimination: 5
---
> Reported: Thu May 19 09:08:14 2011
> Subject: paulSpice199
> Year/month of birth: 1994/01
> Sex: M
> CI type: 24
> Volume: 4
> Range: 9
> Discrimination: 8
mbrenes@nia-login02:data$
```

Encontrar archivos



- ‘find’ es un comando para encontrar archivos en subdirectorios



- Suele no ser tan útil y tiende a ser mejor opción usar ‘ls’ con wildcards ‘*’. Sin embargo, si puede tener usos interesantes

```
[mbrenes@nia-login02: data$ pwd  
/home/d/dvira/mbrenes/workspace/FisicaComputacional/Semana_01/data  
[mbrenes@nia-login02: data$ ls  
alexander Bert Frank_Richard gerdal jamesm Lawrence THOMAS  
[mbrenes@nia-login02: data$ find . -type f -name "*09*"  
. ./jamesm/data_509.txt  
. ./alexander/data_309.DATA  
. ./gerdal/Data0409  
mbrenes@nia-login02: data$ ]
```

Encontrar archivos

- • ‘find’ es un comando para encontrar archivos en subdirectorios
- • Suele no ser tan útil y tiende a ser mejor opción usar ‘ls’ con wildcards ‘*’. Sin embargo, si puede tener usos interesantes

```
[mbrenes@nia-login02: data$ pwd  
/home/d/dvira/mbrenes/workspace/FisicaComputacional/Semana_01/data  
[mbrenes@nia-login02: data$ ls  
alexander Bert Frank_Richard gerdal jamesm Lawrence THOMAS  
[mbrenes@nia-login02: data$ find . -type f -name "*09"  
.jamesm/data_509.txt  
.alexander/data_309.DATA  
.gerdal/Data0409  
mbrenes@nia-login02: data$
```

Our commands

echo arg	echo the argument
pwd	present working directory
ls [dir]	list the directory contents
cd [dir]	change directory
history [num]	print the shell history
man cmd	command's man page
cp file1 file2	copy a file
mv file1 file2	move/rename a file
rm file	delete a file
mkdir dir	create a directory
rmdir dir	delete a directory
more file	scroll through file
less file	scroll through file
cat file	print the file contents
cmd > file	redirect output to file
cmd >> file	append output to file
cmd < file	use file as input to cmd
head file	print first 10 lines of file
tail file	print last 10 lines of file
wc file	word count data of file
find dir	find files

Pipelines

- • Los comandos se pueden entrelazar de manera secuencial mediante *pipes*
 - • Los *pipes* se utilizan mediante el símbolo ‘|’
- • Por ejemplo, en lugar de

```
[mbrenes@nia-login02:~]$ pwd  
/home/d/dvira/mbrenes/workspace/FisicaComputacional/Semana_01/data  
[mbrenes@nia-login02:~]$ cd Lawrence/  
[mbrenes@nia-login02:Lawrence]$ wc Data* > all-wcs  
[mbrenes@nia-login02:Lawrence]$ more all-wcs
```

- • Podemos
 - • De esta forma podemos evitar la creación de un archivo sin utilidad
 - • Con los pipes podemos hacer cadenas de comandos

Sort

- Sort se utiliza para ordenar datos de una manera deseada
 - Modificadores: '-n' (por número), '-k' (por columna), '-r' (reverso)

More commands

wget (curl) url	downloads the url
tar file	handles tar files
cmd1 cmd2	pipe cmd1 output to cmd2
sort file	sorts the lines of file
arg	mandatory argument
[arg]	optional argument

```
[mbrenes@nia-login02:Lawrence$ sort -n -k 3 all-wcs
      9   24   144 Data0234
      9   24   144 Data0310
      9   24   144 Data0382
      9   24   144 Data0430
      9   24   144 Data0506
```

```
[mbrenes@nia-login02:Lawrence$ wc Data* | sort -n -k 3
      9   24   144 Data0234
      9   24   144 Data0310
      9   24   144 Data0382
      9   24   144 Data0430
      9   24   144 Data0506
```

- Los pipes se pueden utilizar de manera secuencial

```
[mbrenes@nia-login02:Lawrence$ wc Data* | sort -n -k 3 | head -n 3
      9   24   144 Data0234
      9   24   144 Data0310
      9   24   144 Data0382
[mbrenes@nia-login02:Lawrence$ wc Data* | sort -n -r -k 3 | head -n 3
  468 1246 7644 total
      9   24   153 Data0214
      9   24   150 Data0515
[mbrenes@nia-login02:Lawrence$ ]
```

Edición de documentos y archivos

- • Existen diversos programas para editar documentos (*IDEs*)
 - • Para terminal: vi, emacs, nano...
 - • Con GUI: sublime text, pycharm, xcode, komodo, gedit...
- • Sus usos dependen de muchas variables pero particularmente:
preferencia personal

```
VIM - Vi IMproved
version 7.4.629
by Bram Moolenaar et al.
Modified by <bugzilla@redhat.com>
Vim is open source and freely distributable

      Help poor children in Uganda!
type :help iccf<Enter>      for information

type :q<Enter>          to exit
type :help<Enter> or <F1> for on-line help
type :help version7<Enter> for version info
```

Primera Tarea:
Buscar un editor
de texto en
terminal y en GUI

Bash scripting

- • Bash permite infinidad de posibilidades con respecto a manejo de datos
 - Una forma muy común de usar bash es por medio de *scripts*. Podemos crear un archivo llamado ‘biggest.sh’

```
mbrenes@nia-login02:Lawrence$ vim biggest.sh
```

```
1#!/bin/bash
2
3wc * | sort -n -k 3 | tail -n 2 | head -n 1
~
```

- • Usando vi, el archivo se guarda mediante tecla ‘ESC’ y digitado ‘:wq’
 - El script se puede ejecutar usando el comando ‘source’

```
mbrenes@nia-login02:Lawrence$ vim biggest.sh
mbrenes@nia-login02:Lawrence$ source biggest.sh
9 24 153 Data0214
mbrenes@nia-login02:Lawrence$
```

Permisos

- • Los archivos en Linux tienen permisos asociados de lectura, escritura y ejecución
- • 'ls -l' muestra los tres permisos en el siguiente orden: lectura-escritura-ejecución
- • Los permisos se muestran tres veces en el siguiente orden: dueño-miembros del grupo-otros

```
[mbrenes@nia-login02:Lawrence$ ls -l biggest.sh  
-rw-r--r-- 1 mbrenes dvira 57 Feb 15 17:40 biggest.sh]
```

Dueño Miembros Otros

- • 'chmod' permite modificar permisos

```
[mbrenes@nia-login02:Lawrence$ ls -l biggest.sh  
-rw-r----- 1 mbrenes dvira 57 Feb 15 17:40 biggest.sh  
[mbrenes@nia-login02:Lawrence$ chmod u+x biggest.sh  
[mbrenes@nia-login02:Lawrence$ ls -l biggest.sh  
-rwxr----- 1 mbrenes dvira 57 Feb 15 17:40 biggest.sh  
[mbrenes@nia-login02:Lawrence$ ./biggest.sh  
9 24 153 Data0214  
mbrenes@nia-login02:Lawrence$ ]
```

grep (extremadamente útil)

- • ‘grep’ es una función de bash que busca patrones
- • Si encuentra el patrón, devuelve la hilera correspondiente

```
[mbrenes@nia-login02:Lawrence$ pwd  
/home/d/dvira/mbrenes/workspace/FisicaComputacional/Semana_01/data/Lawrence  
[mbrenes@nia-login02:Lawrence$ grep 'Range' Data0352  
Range: 2  
[mbrenes@nia-login02:Lawrence$ grep 'Range' *\nData0214:Range: 7  
Data0225:Range: 8  
Data0234:Range: 6  
Data0240:Range: 9
```

- • ‘grep -v’ devuelve las hileras que *no* contienen el patrón

```
[mbrenes@nia-login02:Lawrence$ grep -v 'Range' *\nbiggest.sh:#!/bin/bash\nbiggest.sh:\nbiggest.sh:wc * | sort -n -k 3 | tail -n 2 | head -n 1\nData0214:#\nData0214:Reported: Tue May 24 13:29:57 2011\nData0214:Subject: madonnaBackstreet176\nData0214:Year/month of birth: 1999/10\nData0214:Sex: M\nData0214:CT type: 19
```

cut

- • ‘cut’ permite truncar las salidas de los comandos y/o archivos

```
[mbrenes@nia-login02:Lawrence$ grep 'Ra' * | sort -n -k 2 | tail -n 1  
Data0531:Range: 9  
[mbrenes@nia-login02:Lawrence$ grep 'Ra' * | sort -n -k 2 | tail -n 1 | cut -c -8  
Data0531  
[mbrenes@nia-login02:Lawrence$ grep 'Ra' * | sort -n -k 2 | tail -n 1 | cut -c 10-  
Range: 9  
[mbrenes@nia-login02:Lawrence$ grep 'Ra' * | sort -n -k 2 | tail -n 1 | cut -c 2-5  
ata0  
mbrenes@nia-login02:Lawrence$ ]
```

More commands

wget (curl) url downloads the url
tar file handles tar files
cmd1 | cmd2 pipe cmd1 output to cmd2
sort file sorts the lines of file
source file run the cmds in file
chmod p file change file permissions
grep arg file search for arg in file
cut flags output cut part of output

arg mandatory argument
[arg] optional argument

Guardar resultados en variables

- • Los resultados de un comando se pueden guardar en variables de ambiente

```
[mbrenes@nia-login02:Lawrence$ grep 'Ra' * | sort -n -k 2 | tail -n 1 | cut -c -8  
Data0531  
[mbrenes@nia-login02:Lawrence$ i=`grep 'Ra' * | sort -n -k 2 | tail -n 1 | cut -c -8`  
[mbrenes@nia-login02:Lawrence$ echo $i  
Data0531  
mbrenes@nia-login02:Lawrence$ ]
```

- Nótese el símbolo ` para indicar que el comando se ejecuta, en lugar de guardar el string
- Para guardar el string se utilizan otros símbolos, tales como " " o '''

More commands

wget (curl) url	downloads the url
tar file	handles tar files
cmd1 cmd2	pipe cmd1 output to cmd2
sort file	sorts the lines of file
source file	run the cmds in file
chmod p file	change file permissions
grep arg file	search for arg in file
cut flags output	cut part of output

arg mandatory argument
[arg] optional argument

Argumentos (extremadamente útil)

- • Los scripts de bash pueden recibir argumentos de la terminal
 - • Creamos un script que realiza la operación de 'sort' bajo el parámetro 'Range'
 - • Utilizamos el denominador `${1}` para indicar *la primera variable de entrada del script*
 - • `${2}, ${3}, ...` indican las siguientes entradas

```
[mbrenes0@hxckid:Lawrence$ pwd  
/Users/mbrenes0/FisicaComputacional/Semana_01/data/Lawrence  
[mbrenes0@hxckid:Lawrence$ vim biggestRange.sh  
[mbrenes0@hxckid:Lawrence$ chmod +x biggestRange.sh  
[mbrenes0@hxckid:Lawrence$ mv biggestRange.sh ..  
[mbrenes0@hxckid:Lawrence$ cd ..  
[mbrenes0@hxckid:data$ ls  
Bert          Frank_Richard    Lawrence      THOMAS      alexander    biggestRange.sh gerdal      jamesm  
mbrenes0@hxckid:data$
```

```
1 #!/bin/bash  
2  
3 grep 'Range'  ${1}/* | sort -n -k 2 | tail -n 1
```

Argumentos (extremadamente útil)

- • Ahora podemos usar el script con el argumento que queramos

```
[mbrenes0@hxckid:~/data$ pwd  
/Users/mbrenes0/FisicaComputacional/Semana_01/data  
[mbrenes0@hxckid:~/data$ ls  
Bert          Frank_Richard  Lawrence      THOMAS      alexander    biggestRange.sh gerdal      jamesm  
[mbrenes0@hxckid:~/data$ ./biggestRange.sh Bert  
Bert/audiorecord-00384.txt:Range: 10  
[mbrenes0@hxckid:~/data$ ./biggestRange.sh THOMAS  
THOMAS/0336:Range: 10  
[mbrenes0@hxckid:~/data$ ./biggestRange.sh james  
grep: james/*: No such file or directory  
[mbrenes0@hxckid:~/data$ ./biggestRange.sh jamesm  
jamesm/data_517.txt:Range: 10  
[mbrenes0@hxckid:~/data$ ]
```

For loops

- • Bash permite iteración mediante *for loops* como programación standard
- • Los *for loops* en bash tienen su propia sintaxis

```
1 #!/bin/bash
2
3 for dir in alexander Bert Frank_Richard
4 do
5   echo "The biggest range in " ${dir} " is:" ./biggestRange.sh ${dir}
6 done
```



loop_biggestRange.sh

- • Mucha utilidad para manejo de archivos y datos numéricos

```
[mbrenes0@hxckid:data$ ./loop_biggestRange.sh
The biggest range in alexander is: ./biggestRange.sh alexander
The biggest range in Bert is: ./biggestRange.sh Bert
The biggest range in Frank_Richard is: ./biggestRange.sh Frank_Richard
mbrenes0@hxckid:data$ ]
```

- • Así como también se pueden usar *if statements*

```
1 #!/bin/bash
2
3 cd ${1}
4 for filename in *
5 do
6   if [ "$filename" == "NOTES" ]
7   then
8     echo "I'm a NOTES file."
9   fi
10 done
```

if_test.sh

- • Dichos statements se pueden utilizar para control de flujos

```
[mbrenes0@hxckid:~/data$ ./if_test.sh jamesm
I'm a NOTES file.
[mbrenes0@hxckid:~/data$ ./if_test.sh alexander
[mbrenes0@hxckid:~/data$ ]
```

Conclusión



- Estos comandos son los más básicos para empezar en bash



- Algunos se usan rara vez, a menudo o absolutamente todo el tiempo



- Existen funciones mucho más complejas (e.g.: `awk`)



- El mejor recurso para bash: google, stack overflow, etc