Pablo Vinuesa (vinuesa [at] ccg . unam . mx)

Centro de Ciencias Genómicas-UNAM, México

http://www.ccg.unam.mx/~vinuesa/ @pvinmex1

Tema 1: Introducción al biocómputo en sistemas Unix/Linux

https://github.com/vinuesa/TIB-filoinfo

1. ¿Qué son la bioinformática y el biocómputo, y cómo pueden ayudarme para mi trabajo en biología?
2. ¿Qué es UNIX y Linux?
3. Tengo una PC que corre Windows, ¿cómo puedo correr Linux en mi máquina?
4. ¿Cómo hago trabajar a UNIX/Linux? – el shell: comandos y conceptos básicos
5. Sesión práctica: uso de tuberías de comandos UNIX/Linux para procesar archivos de secuencias

Introducción al biocómputo en sistemas UNIX/Linux

- **¿Qué es UNIX? - fuente: Wikipedia**

- **Unix** (registrado oficialmente como **UNIX**®) es un sistema operativo portable, multitarea y multiusuario
- su desarrollo inicia en 1969 por un grupo de empleados de los laboratorios Bell de AT&T, entre los que figuran Ken Thompson, Dennis Ritchie y Douglas McIlroy
- UNIX es un Sistema Operativo no libre muy popular, porque está basado en una arquitectura que ha demostrado ser técnicamente estable.
- **MacOS X** es un derivado de UNIX BSD!

Ken Thompson y Dennis Ritchie
Fuente: Wikipedia

Introducción al biocómputo en sistemas UNIX/Linux

- **¿Qué es Linux? - Evolución de sistemas UNIX y similares a UNIX (fuente: Wikipedia)**

GNU: En 1983, Richard Stallman anunció el Proyecto GNU, un ambicioso esfuerzo para crear un **sistema similar a Unix, que pudiese ser distribuido libremente** El software desarrollado por este proyecto -por ejemplo, GNU Emacs y GCC - también han sido parte fundamental de otros sistemas UNIX. (**vean conferencias de R. Stallman en youtoube** ...)

Richard Stallman

POWERED BY Linux

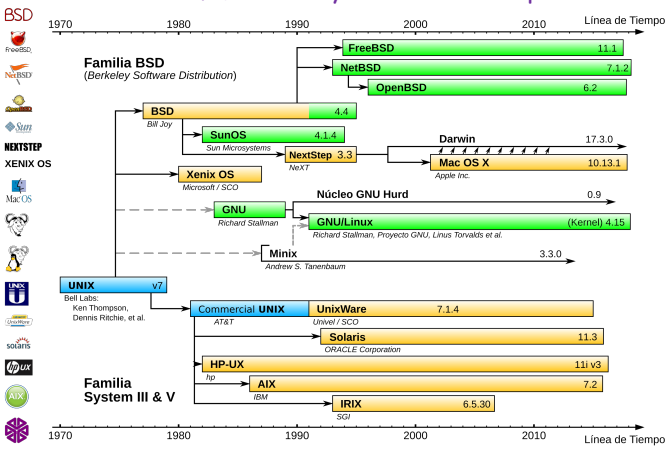
Linus Torvalds

Linux: En 1991, cuando Linus Torvalds empezó a proponer el *núcleo* Linux y a reunir colaboradores, las herramientas GNU eran la elección perfecta. Al combinarse ambos elementos, conformaron la base del sistema operativo (basado en POSIX) que hoy se conoce como GNU/Linux.

Las distribuciones basadas en el núcleo, el software GNU y otros agregados como Red Hat Linux y Debian GNU/Linux, se han hecho populares tanto entre los aficionados a la computación como en el mundo empresarial y científico. Linux tiene un origen independiente, por lo que se considera un 'clon' de UNIX y no un UNIX en el sentido histórico.

Introducción al biocómputo en sistemas UNIX/Linux

- La evolución de las familias UNIX y clones – Fuente Wikipedia



Introducción al biocómputo en sistemas UNIX/Linux

- Tengo una PC que corre windows, ¿cómo puedo correr Linux en mi máquina?

1. **Puedes instalar Linux en una nueva partición (Lo más recomendable)**.
   descargas gratuitas de distribuciones desde:

=> 1. Ubuntu 18.04 LTS - http://www.ubuntu.com/getubuntu/download

Ver instrucciones de instalación aquí:

http://www.ubuntu.com/download/desktop/install-desktop-latest

Centos - https://www.centos.org/download/

Fedora - http://fedoraproject.org/es/get-fedora

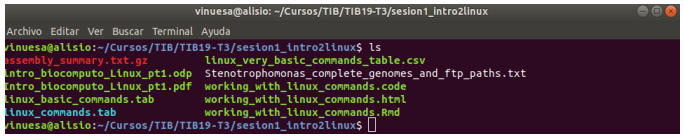
Biolinux - http://environmentalomics.org/bio-linux/

2. **Puedes instalar MobaXterm**, que proporciona una terminal para Windows con un servidor de ambiente gráfico X11, un cliente SSH para establecer sesiones remotas seguras con un servidor, diversas herramientas de red y más.
   https://mobaxterm.mobatek.net/download.html

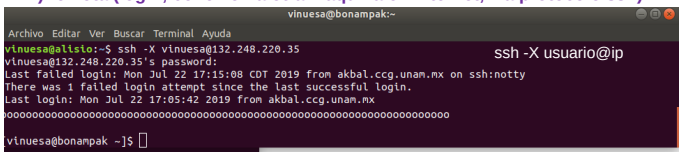
Introducción al biocómputo en sistemas UNIX/Linux

- Tipos de sesiones shell

1) local (acceder a tu computadora)



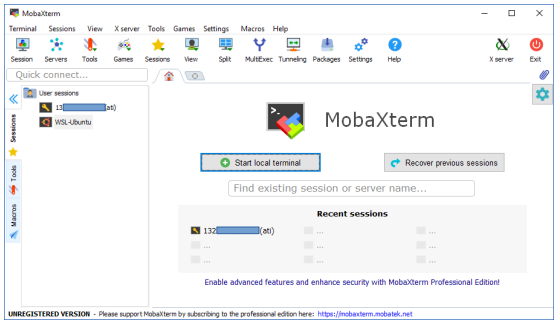
2) remota (login; conexión a otra máquina en Internet, vía protocolo ssh)



Si tu máquina corre Windows, descarga el MobaXterm installer desde:

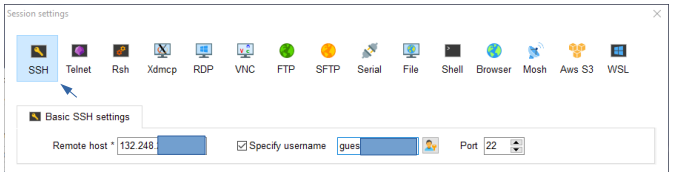
https://download.mobatek.net/1102018093083521/MobaXterm_Installer_v11.0.zip

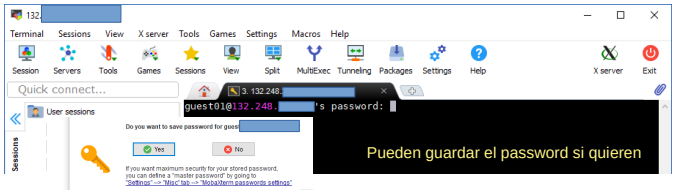
- Descomprime **MobaXterm_Installer_v11.0.zip** e instálalo en tu máquina (x2click)



Uso de MobaXterm para establecer sesiones remotas a un servidor vía protocolo SSH

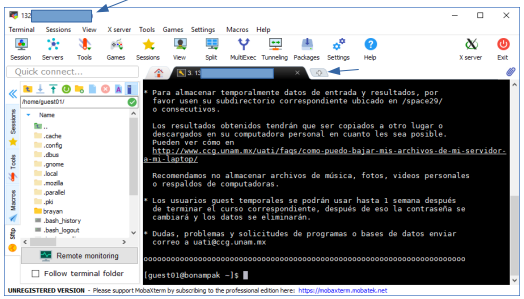
1. Para establecer una conexión remota vía SSH elige el protocol SSH y usa la IP, usuario y contraseña indicados en el taller





Uso de MobaXterm para establecer sesiones remotas a un servidor vía protocolo SSH

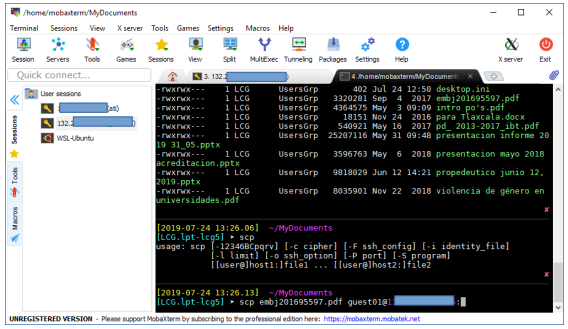
Una vez establecida la sesión remota al servidor, estarán trabajando en él!



- El panel mostrado a la izquierda les sirve para navegar el sistema remoto,pero lo haremos desde la línea de comandos que nos ofrece la terminal (área negra).
- Pueden abrir más pestañas en la terminal. Estas serán sesiones locales (si no hacen ssh)

Uso de MobaXterm para trabajo con sesiones remotas y locales

Una vez establecida la sesión remota al servidor, estarán trabajando en él!



- El panel mostrado a la izquierda tiene abierta una sesión remota.
- La terminal de la derecha corre una sesión local (en tu computadora)
- Pueden copiar un archivo de su máquina local al servidor con:

```
$ scp file user@xxx.xxx.xxx.xxx:/path/to/final/dir
```

Expande MobaXterm con plugins

https://mobaxterm.mobatek.net/plugins.html

- **Instalen sin falta Git** para poder clonar y actualizar el repositorio GitHub del taller
- Recomendamos instalar también GCC dev. Tools, Nedit, Perl y Zip

Introducción al biocómputo en sistemas UNIX/Linux

- explorando el nuevo ambiente ... El shell y la interpretación de comandos, unos ejemplos

1. – ¿Qué máquina es ésta a la que estoy conectado?

- **hostname** - read or set the hostname or the NIS domain name

```
vinuesa@bonampak:/$ hostname
bonampak.ccg.unam.mx
vinuesa@bonampak:/$ hostname –i # corre también hostname –-help
132.248.*.*
```

- **uname** - Print certain system information

```
vinuesa@bonampak:~$ uname
Linux
vinuesa@bonampak:~$ uname -a
Linux bonampak.ccg.unam.mx 3.10.0-862.9.1.el7.x86_64 #1 SMP Mon Jul 16
16:29:36 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux
```

Introducción al biocómputo en sistemas UNIX/Linux

- explorando el nuevo ambiente ... El shell y la interpretación de comandos, unos ejemplos

1. – ¿Qué procesos se están corriendo en el sistema?

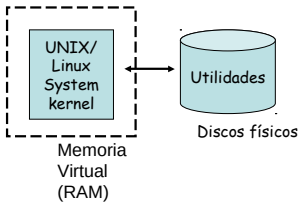
- **top** - monitor system load

```
vinuesa@bonampak:/$ top

top - 19:13:22 up 184 days,  3:14,  9 users,  load average: 1.93, 2.10, 2.13
Tasks: 669 total,   2 running, 667 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.9 us,  2.0 sy,  0.0 ni, 96.1 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 52807612+total,  4045748 free, 11101945+used, 41301091+buff/cache
KiB Swap: 13421772+total, 13331803+free,   899696 used. 41411052+avail Mem

   PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND

 84375  llozano   20   0 9587984   5.1g   1388 R 139.7  1.0  48721:34 spades-hammer
191547 ati       20   0   67.3g  63.0g  63.0g S 109.9 12.5 322566:44 VBoxHeadless
181971 ati       20   0   34.4g  32.2g  32.2g S   1.3  6.4 124996:03 VboxHeadless
 53501  ropomari  20   0  158872   2684   1224 S   0.7  0.0   0:03.54 sshd
 1507   root      20   0   13216    524    524 S   0.3  0.0  26:48.50 rngd
 54660  vinuesa   20   0  162500   2824   1576 R   0.3  0.0   0:00.26 top

...
```

Introducción al biocómputo en sistemas UNIX/Linux

- explorando el nuevo ambiente ... El shell (consola) y la interpretación de comandos

- ¿Qué es el shell?



Los sistemas UNIX/Linux se dividen lógicamente en dos piezas: el **kernel y las utilidades**.

El kernel es el corazón del sistema y reside en la memoria de la computadora desde el momento que se arranca y hasta que se apaga.

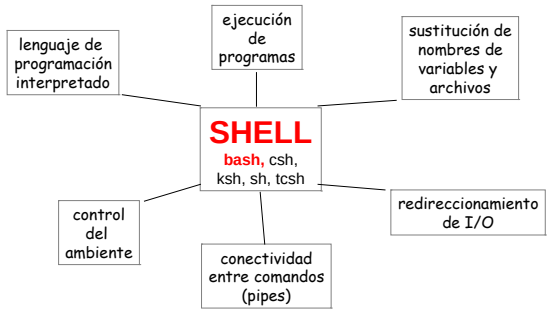
Las utilidades (comandos), residen en el disco físico y se cargan en memoria sólo cuando son llamadas.

El shell también es un programa. Se carga automáticamente en memoria desde que uno hace el login a una máquina para que el usuario pueda interactuar con ella.

Introducción al biocómputo en sistemas UNIX/Linux

- explorando el nuevo ambiente ... El shell y la interpretación de comandos

- Las responsabilidades del **shell –**



Introducción al biocómputo en sistemas UNIX/Linux

• **explorando el nuevo ambiente … El shell y la interpretación de comandos**

– ¿Dónde encuentro una lista y descripción básica de los comandos disponibles ?

1. Mira estas entradas en Wikipedia:
   http://en.wikibooks.org/wiki/Linux_Guide/Linux_commands
   http://en.wikipedia.org/wiki/List_of_Unix_programs

2. Y corre estos comandos para ver parte de los programas del sistema instalados en el servidor o en tu máquina:
   ls /bin
   ls /usr/bin

3. Un sencillo tutorial que todos deberían haber revisado ya (caps 1-5):
   http://www.ee.surrey.ac.uk/Teaching/Unix/

Introducción al biocómputo en sistemas UNIX/Linux

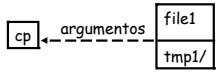
• **explorando el nuevo ambiente … El shell y la interpretación de comandos**

- **Ejecución de programas por el shell – ejemplos**

• **formato básico de un comando**

comando [argumento1 arg2 arg3 …]

```
-bash-3.1$
-bash-3.1$ cp file1 tmp1/
-bash-3.1$
```



¡los espacios separan argumentos!
necesito al menos 1 espacio entre comandos y argumentos!
Nombres de archivos en UNIX/Linux de preferencia no deben contener espacios

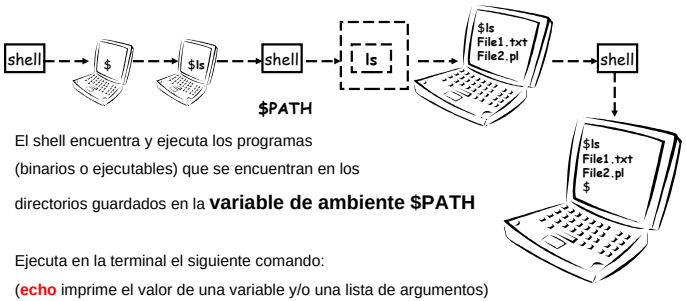
• Además de **argumentos**, los comandos pueden tener **opciones**, las cuales preceden a los argumentos y llevan un guión sencillo delante de una o más opciones.

```
-bash-3.1$ ls -l /home    # prueba también solo ls /home
drwxr-xr-x.  2 aagarcia students  4096 Oct 30  2014 aagarcia
drwxr-xr-x.  2 aanaya   students  4096 Oct 30  2014 aanaya
drwxr-xr-x.  2 acampos  students  4096 Oct 30  2014 acampos
drwxr-xr-x.  2 acarmona students  4096 Oct 30  2014 acarmona
drwxr-xr-x.  4 agodinez students  4096 Jul  1  2016 agodinez
drwxr-xr-x.  2 aguzman  students  4096 Oct 30  2014 aguzman
```

Introducción al biocómputo en sistemas UNIX/Linux

• **explorando el nuevo ambiente … El shell y la interpretación de comandos**

- **Tecleando comandos para el shell – el ciclo de comandos**



El shell encuentra y ejecuta los programas (binarios o ejecutables) que se encuentran en los directorios guardados en la **variable de ambiente $PATH**

Ejecuta en la terminal el siguiente comando:

(**echo** imprime el valor de una variable y/o una lista de argumentos)

```
-bash-3.1$
vinuesa@bonampak:/$ echo $PATH
```

Commandos Básicos

- ls (list)
  - $ ls -l
  - $ ls -a
  - $ ls -la
  - $ ls -l --sort=time # ls -ltr
  - $ ls -l --sort=size -r #ls -lSr
  - $ ls *.txt
- cd (change directory)
  - $ cd dir
  - $ cd ./dir
  - $ cd ../
  - $ cd ../../
  - $ cd /export/space2/tib/filo
  - $ cd == cd $HOME
- pwd (print working directory)
  - $ pwd
- ~
  - $ cd ~
- *~user*
  - $ cd ~vinuesa
- ¿Qué hará "cd ~vinuesa"?
- ¿y cd?

- which

$ which blastn

- locate
  - $ locate get_homologues.pl
  - $ locate mi_archivo
- find
  - $ find / | grep stdio.h
  - $ find /usr/include | grep stdio.h
  - $ find . -type d
  - $ find /home/vinuesa -name *TIB*
- man (manual pages for command)
  - $ man ls
  - $ man find
  - $ man man

Commandos Básicos (cont.)

- echo (print to STDOUT)
  - $ echo “Hello World”
  - $ echo -n “Hello World”
- cat (concatenate)
  - $ cat /proc/cpuinfo
  - $ cat arch1 arch2
- cp (copy)
  - $ cp arch1 dir1
  - $ cp -r dir1 ~vinuesa/tmp
- mv (move or rename)
  - $ mv arch1 archivo1
  - $ mv arch1 ~vinuesa/tmp
- mkdir (make directory)
  - $ mkdir dir2
  - $ mkdir -p dir2/practica1
- rm (remove)
  - $ rm arch1
  - $ rm -rf dir2
- less (paginador)
  - $ less archivo.txt # q para salir
- head (ver cabecera del archivo)
  - $ head -5 archivo1.txt
- tail (ver cola del archivo)
  - $ tail -1 archivo.txt
  - $ tail -f logfile.txt
- sed (stream editor)
  - sed 's/esto/aquello/' archivo.txt
- vim (vi improved; a powerful command line text editor in Linux)
- gedit (editor de texto con interfaz gráfica en gnome)
- nedit (otro editor de texto con interfaz gráfica)

Commandos Básicos (cont.)

- tar & gzip
  - $ tar cvzf dir1.tar.gz dir1
  - $ tar cvzf dir1.tgz dir1
  - $tar cvpf dir1.tar dir1
- gzip (gnuzip, compress)
  - $ gzip -9 dir1.tar
  - # genera dir1.tar.gz
- untar & ungzip
  - $ tar xvfz dir1.tar.gz
- touch
  - $ touch foo
- head
  - $ head archivo.txt
- tail
  - $ tail archivo.txt
  - $ tail -f archivo2.txt
- Pipe, >, grep, sort, cut, uniq
  - $ ls -l /home | less
  - $ cat /etc/passwd | cut -d: -f1 | sort
  - $ ls -l /home | grep stud | wc -l > stud.txt
  - $ ls -l /home | grep -v stud | sort | unic -c
- backticks
  - $ echo “The date is `date`”
  - $ echo `seq 1 10`
- Hard, soft (symbolic) link
  - ln -s /path/to/gbk_files/*.gbk .
- scp (secure copy)
  - scp arch1 vinuesa@buluc.lcg.unam.mx:$HOME/tmp

Commandos Básicos (cont)

- Uso del disco duro
  - $ df -h /
- Uso discro de archivos
  - $ du -sxh ~/
- **Uso avanzado: programación del Shell ☺**

- Asignación de variables y bucles for

- var1=123; echo $var1; var2=/home/pepin && echo $var2
- for file in *faa; do muscle < $file > ${file%.*}_aln.faa; done
- for file in $(ls *faa | grep rpoB); do echo -n $file; grep -c '>' $file; done

Linux text editors

- con entorno gráfico
  - **gedit**
  - nedit
- Sin entorno gráfico
  - **Vim**
  - emacs
  - nano
  - pico

Vim – the Linux power text editor

- 2 modes
  - Input mode
    - ESC to back to cmd mode
  - Command mode
    - Cursor movement
      - h (left), j (down), k (up), l (right)
      - ^f (page down)
      - ^b (page up)
      - ^ (first char.)
      - $ (last char.)
      - G (bottom page)
      - :1 (goto first line)
    - Swtch to input mode
      - a (append)
      - i (insert)
      - o (insert line after)
      - O (insert line before)
- Delete
  - dd (delete a line)
  - d10d (delete 10 lines)
  - d$ (delete till end of line)
  - dG (delete till end of file)
  - x (current char.)
- Paste
  - p (paste after)
  - P (paste before)
- Undo
  - u
- Search
  - /
- Save/Quit
  - :w (write)
  - :q (quit)
  - :wq (write and quit)
  - :q! (give up changes)

Introducción al biocómputo en sistemas UNIX/Linux

- **explorando el nuevo ambiente ... Trabajando eficientemente con el shell**

¿Cómo me muevo en la línea de comandos?

- Usa **ctrl-e** para ir al final de la línea
- Usa **ctrl-a** para ir al principio de la línea

¿Cómo edito la línea de comandos?

- Usa la techa **backspace** para eliminar uno a uno caracteres (del final hacia el principio)
- Usa **ctrl-w** para eliminar una palabra completa (del final hacia el principio)
- Usa **ctrl-u** para eliminar la línea completa (del final hacia el principio)

¿Cómo aborto o suspendo la ejecución de un comando?

Usa **ctrl-c** para abortar la ejecución del último comando

- Usa **ctrl-z** para suspender la ejecución del último comando
- Usa **bg** para poner este último comando a correr en el fondo (background)

Repetición de la ejecución de un comando y completado de nombres de comandos/archivos

- Unix recuerda los comandos ejecutados: usa flecha arriba o abajo para moverte por el historial de comandos "history" file o escribe **history | grep comando**
- Usa **TAB** para completar automáticamente el nombre de comandos, archivos o directorios

Prácticas – asegúrate de tener clonado y actualizado el repositorio GitHub

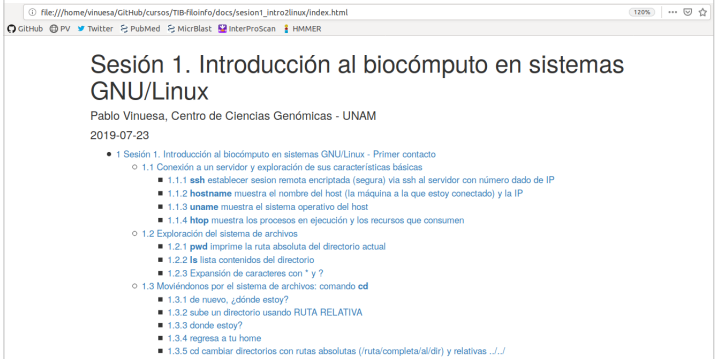
https://github.com/vinuesa/TIB-filoinfo

1. Instala git en tu máquina (**sesión local**) con `$ sudo apt install git`
2. genera un directorio GitHub en tu $HOME y clona el repositorio así:

```
$ cd && mkdir GitHub && cd GitHub
  git clone https://github.com/vinuesa/TIB-filoinfo.git
```

Una vez clonado el repositorio, puedes abrir localmente los archivos html con las instrucciones para los ejercicios en tu navegador web de preferencia

Así evitamos consumir innecesariamente el siempre limitado ancho de banda.
¡Gracias por tu colaboración!



NAVEGACIÓN DEL SISTEMA Y MANEJO DE ARCHIVOS - detalles

Comandos básicos:

pwd — Print name of current working directory.

cd — Change directory.

ls — List directory contents.

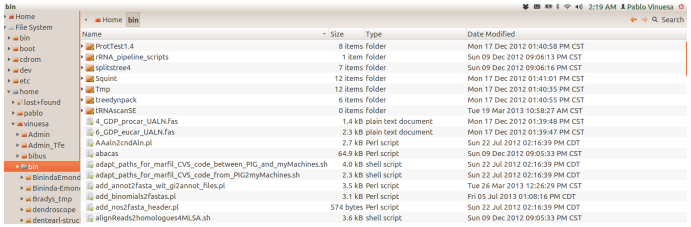
Ejecuta los comandos provistos en el archivo “working_with_linux_commands.code”

que puedes descargar desde moodle

El sistema de archivos

Igual que en Windows, los sistemas operativos tipo UNIX como Linux organiza sus archivos en una estructura jerárquica de directorios. El primer directorio del sistema Se conoce como el **directorio raíz /**, que contiene archivos y subdirectorios.

Existen interfaces gráficas para navegar y explorar el sistema como se muestra abajo, pero la manera más poderosa y flexible de hacerlo es mediante comandos como **cd**, **ls** y **pwd**



Vista parcial del árbol de directorios usando el ambiente gráfico gnome con el navegador Nautilus

El sistema de archivos – su navegación

Comandos básicos: pwd, ls, cd

```
vinuesa@yaxche[~]$ pwd # imprime directorio acutal
/home/vinuesa    # ruta absoluta, desde el directorio raíz '/'

vinuesa@yaxche[~]$ ls /  # lista contenidos directorio raíz
bin   etc     initrd.img      lib64       mnt   root  selinux  tmp  vmlinuz
boot  export  initrd.img.old  lost+found  opt   run   srv      usr  vmlinuz.old
dev   home    lib             media       proc  sbin  sys      var


vinuesa@yaxche[~]$ cd /; pwd # separa múltiples comandos con ;
/

vinuesa@yaxche[/]$ ls
bin   etc     initrd.img      lib64       mnt   root  selinux  tmp  vmlinuz
boot  export  initrd.img.old  lost+found  opt   run   srv      usr  vmlinuz.old
dev   home    lib             media       proc  sbin  sys      var

vinuesa@yaxche[/]$ cd; pwd # cd sin argumento nos regresa a $HOME
vinuesa@yaxche[~]$ pwd
/home/vinuesa
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

I.– trabajando con archivos

- **ls** – lista información sobre archivos y directorio

opciones importantes:
-a display all, including hidden files .file.txt
-d display information about directory
-l long format
-F classify
-h human readable sizes
-r reverse sorted order
-R recursively lists subdirectories
-s display sorted by size
-t display sorted by creation time
-x display files sorted by lines (default is by columns)
-1 display files one per line

Ejemplo:

ls –ltr # imprime lista de archivos ordenados reversamente por tiempo de modificación

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

1.– trabajando con archivos – sustitución de nombres de archivos

```
-bash-3.1$ ls
inscritos_forma_inscr.1liner  posters_old13Jun.html  posters_old4.html
poster_1liners.txt            posters_old3.html      posters_old.html

-bash-3.1$ ls *txt  # lista sólo los archivos terminados en .txt
poster_1liners.txt

-bash-3.1$ ls *old?.html
posters_old3.html  posters_old4.html
-bash-3.1$

-bash-3.1$ ls *[0-9]*
inscritos_forma_inscr.1liner  poster_1liners.txt  posters_old13Jun.html
posters_old3.html  posters_old4.html

-bash-3.1$ rm *[2-9]*      # Elimina todos los archivos que contien
                           # dígitos del 2-9 en sus nombres
-bash-3.1$ ls
posters_old.html poster_1liners.txt posters_old.html

-bash-3.1$ cp *.* ~/temp  # copia los archivos que quedan a ~/temp
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

1.– trabajando con archivos – atributos de archivos y cambio de permisos de acceso

```
-bash-3.1$ ls -l
drwxr-xr-x 2 vinuesa cifn-ux 1.0K Jun 20 20:13 .
drwxr-xr-x 8 vinuesa cifn-ux 1.0K Jun 20 20:13 ..
-rw-r--r-- 1 vinuesa cifn-ux   76 Jun 17 14:00 inscritos_forma_inscr.1liner
-rw-r--r-- 1 vinuesa cifn-ux  392 Jun 14 20:02 poster_1liners.txt

-bash-3.1$ chmod u+x,g-r,o-r inscritos_forma_inscr.1liner

-bash-3.1$ ls –l
-rwx------ 1 vinuesa cifn-ux     76 Jun 17 14:00 inscritos_forma_inscr.1liner
-rw-r--r-- 1 vinuesa cifn-ux    392 Jun 14 20:02 poster_1liners.txt


-bash-3.1$ chmod 760 poster_1liners.txt
-bash-3.1$ ls –l
-rwx------ 1 vinuesa cifn-ux  76 Jun 17 14:00 inscritos_forma_inscr.1liner
-rwxrw---- 1 vinuesa cifn-ux  392 Jun 14 20:02 poster_1liners.txt
-bash-3.1$
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

1.– trabajando con archivos – atributos de archivos y cambio de permisos de acceso

Para que un archivo que contiene un programa (sea un binario o un script) pueda ser ejecutado desde cualquier directorio del sistema tiene que cumplir 2 condiciones:

1.El archivo tiene que estar en el PATH

2.El usuario tiene que tener permisos de lectura y ejecución para dicho archivo

Comprueba los permisos de los binarios estándar de Linux ejecutando:

```
vinuesa@vinuesa-laptop:~$ ls -l /bin
total 9040
-rwxr-xr-x 1 root root  959120 Mar 28 12:02 bash
-rwxr-xr-x 3 root root   31112 Dec 15  2011 bunzip2
-rwxr-xr-x 1 root root 1832016 Nov 16  2012 busybox
-rwxr-xr-x 3 root root   31112 Dec 15  2011 bzcat
lrwxrwxrwx 1 root root       6 Dec 15  2011 bzcmp -> bzdiff
…
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

Explorando la estructura de directorios con el comando ls

```
pablo@Tenerife:~$ ls -F /    # exploremos el directorio raíz
'/'
bin/   cdrom@  etc/   initrd.img@      lib/    lib64@       media/  opt/
root/  selinux/  sys/  usr/  vmlinuz@      xorg.conf.new boot/  dev/    home/
initrd.img.old@  lib32/  lost+found/  mnt/    proc/  sbin/  srv/      tmp/
var/  vmlinuz.old@


pablo@Tenerife:~$ ls -F /bin  # veamos el contenido de /bin
bash*          bzless@  dbus-cleanup-sockets*  egrep*       kbd_mode*   ls*
bunzip2*       bzmore*  dbus-daemon*           false*       kill*       lsmod*
busybox*       cat*     dbus-uuidgen*          fgconsole*   ksh@        mkdir*
bzcat*         chgrp*   dd*                    fgrep*       less*       mknod*
bzcmp@         chmod*   df*                    fuser*       lessecho*   mktemp*
bzdiff*        chown*   dir*                   fusermount*  lessfile@   more*
bzegrep@       chvt*    dmesg*                 grep*        lesskey*    mount*
bzexe*         cp*      dnsdomainname*         gunzip*      lesspipe*   mountpoint*
bzfgrep@       cpio*    domainname*            gzexe*       ln*         mt@
bzgrep*        csh@     dumpkeys*              gzip*        loadkeys*   mt-gnu*
bzip2*         dash*    echo*                  hostname*    login*      mv*
bzip2recover*  date*    ed*                    ip*          lowntfs-3g*  nano*
 … y muchos más
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

1.– trabajando con archivos – atributos de archivos y directorios (permisos)

```
bash-3.1$ ls -lFtr
-rw-r--r-- 1 vinuesa cifn-ux  10061 Jun 14 00:46 profesores.html  # - archivo
drwxr-xr-x 2 vinuesa cifn-ux     96 Jun 15 18:08 perl_scripts/    # d directorio
```

modo de acceso o permisos: | links | dueño | grupo | bytes | Fecha y hora modificación | nombre de archivo –rw-r Nombre de directorio drwxr

usuario
grupo
otros

Parámetros para el comando **chmod** (change mode)

User	Type	Rights
u – user	+ add	r – read
g – group	- delete	w – write
O – others		x – execute
a - all		

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

1.– trabajando con archivos – atributos de archivos (permisos)

```
bash-3.1$ ls -lFtr
-rw-r--r-- 1 vinuesa cifn-ux  10061 Jun 14 00:46 profesores.html  # - archivo
drwxr-xr-x 2 vinuesa cifn-ux     96 Jun 15 18:08 perl_scripts/    # d directorio
```

modo de acceso o permisos: | links | dueño | grupo | bytes | Fecha y hora modificación | nombre de archivo –rw-r Nombre de directorio drwxr

usuario
grupo
otros

Parámetros para el comando **chmod** (change mode)

User	Group	Others
r w x	r w x	r w x
4 2 1	4 2 1	4 2 1

Read =4
Write =2
Execute =1

Ej. **chmod 651 archivo** -> lo hace u: r+w; g: r+x; o: x

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

1.– trabajando con archivos – atributos de archivos y cambio de permisos de acceso

Veamos un ejemplo: vamos a escribir y ejecutar nuestro primer script de shell.

Teclea lo siguiente en la terminal:

```
vinuesa@ivory:~/cursos/perl4bioinfo$ cat > hello_shell.sh # enter
echo "Hola $USER!"
echo -n 'hoy es: '; date
echo "usas el shell: $SHELL"
echo -n 'y tu computadora es: '; uname
^D # esto es Ctrl-D


vinuesa@ivory:~/cursos/perl4bioinfo$ ls -l
-rw-r--r-- 1 vinuesa vinuesa 133 Aug  7 17:52 hello_shell.sh
# (644)
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

1.– trabajando con archivos – atributos de archivos y cambio de permisos de acceso

```
vinuesa@ivory:~/cursos/perl4bioinfo$ ls -l
-rw-r--r-- 1 vinuesa vinuesa 133 Aug  7 17:52 hello_shell.sh # (644)
```

Por tanto para poder ejecutar el script necesitamos primero cambiarle los permisos,

generalment con **chmod** 755 mi_script

```
vinuesa@ivory:~/cursos/perl4bioinfo$ chmod 755
hello_shell.sh
vinuesa@ivory:~/cursos/perl4bioinfo$ ls -l
-rwxr-xr-x 1 vinuesa vinuesa 133 Aug  7 17:52
hello_shell.sh

#  y ahora corro el script
vinuesa@ivory:~/cursos/perl4bioinfo$ ./hello_shell.sh
Hola vinuesa!
hoy es: Wed Aug  7 18:03:31 CDT 2013
usas el shell: /bin/bash
y tu computadora es: Linux
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

1.– trabajando con archivos – atributos de archivos y cambio de permisos de acceso

1) Al escribir un archivo de texto con un editor estándar, como al escribir un programa en Bash o Perl, el sistema operativo por defecto le otorga los permisos

   `rw-r--r-- (644)`

   Por tanto para poder ejecutar el script necesitamos primero cambiarle los permisos, generalment con

   chmod 755 mi_script (rwxr-xr-x)

2) Si el script no queda guardado en un directorio del PATH, tendremos que indicar la ruta de acceso al mismo, sea la ruta absoluta o relativa

```
./mi_script # desde el dir actual o ruta relativa
/ruta/completa/a/mi_script # ruta absoluta
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

I.– trabajando con archivos – ¿dónde estoy en el árbol de directorios?

- **pwd** – print working directory

```
-bash-3.1$ pwd
/home/vinuesa/public_html/tlem09
-bash-3.1$ ls -lFtr
total 472
drwxr-xr-x 2 vinuesa cifn-ux   1024 Apr  3 19:21 css/
drwxr-xr-x 2 vinuesa cifn-ux   1024 Apr 13 14:12 images/
-rw------- 1 vinuesa cifn-ux  15877 May 14 01:22 index.html.save
drwxr-xr-x 2 vinuesa cifn-ux   1024 May 25 12:03 docs/
-rw-r--r-- 1 vinuesa cifn-ux  17047 Jun 10 11:36 index.html
-rw-r--r-- 1 vinuesa cifn-ux  17640 Jun 10 11:58 recursos_bioinformatica.html
-rw-r--r-- 1 vinuesa cifn-ux 121862 Jun 13 14:53 posters_new.html
-rw-r--r-- 1 vinuesa cifn-ux  10061 Jun 14 00:46 profesores.html
drwxr-xr-x 2 vinuesa cifn-ux     96 Jun 15 18:08 perl_scripts/
…
-bash-3.1$
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

1.– trabajando con archivos

- **cat** – Visualizar o concatenar el contenido de archivos

```
Usage: cat [OPTION] [FILE]...
Concatenate FILE(s), or standard input, to standard output.

  -b, --number-nonblank    number nonblank output lines
  -n, --number             number all output lines
  -s, --squeeze-blank      never more than one single blank line
  -T, --show-tabs          display TAB characters as ^I
```

- **nedit** es un buen editor gráfico;
  prueba a correr: **nedit hello_shell.sh &**
- **(vi)/vim**, son los editores estándar de UNIX/Linux (no gráfico), también **pico** y **emacs**
- **less** – es un paginador (muestra archivos por pantalla)
- **more** – es otro paginador (más viejo y con menor funcionalidad, por tanto: “less is more”

Introducción al biocómputo en sistemas UNIX/Linux

· Comandos y conceptos básicos

1.– trabajando con archivos

- **less** – un paginador con muchas opciones (ver 'man less')

```
                          MOVING

  f  ^F  ^V  SPACE  *  Forward  one window (or N lines).
  b  ^B  ESC-v      *  Backward one window (or N lines).
  nG                *  Go TO LINE no. N
-------------------------------------------------------------------------
                        SEARCHING

  /pattern          *  Search forward for (N-th) matching line.
  ?pattern          *  Search backward for (N-th) matching line.
-------------------------------------------------------------------------
                          QUIT
   q
```

Ejemplo:

```
pablo@Tenerife:~$ less /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
...
```

Introducción al biocómputo en sistemas UNIX/Linux

· Comandos y conceptos básicos

2. - trabajando con directorios: rutas absolutas y relativas

• La **ruta absoluta** de un archivo o directorio es aquella que **apunta hacia éste desde el directorio raíz /,** tal y como se ve en la salida del comando pwd.
• La **ruta relativa** es aquella que parte del directorio actual, el cual se representa por un punto **.** y el directorio parental (uno arriba) mediante dos puntos **..**

```
# iniciamos desde este directorio: /home/vinuesa/cursos/intro2bioinfo

 # vamos a nuestro $HOME usando la ruta relativa, es decir, 2 dirs arriba
vinuesa@ivory:~/cursos/intro2bioinfo$ cd ../../
vinuesa@ivory:~$ pwd
/home/vinuesa
vinuesa@ivory:~$ cd cursos/intro2bioinfo/  # volvemos al dir intro2bioinfo

# vamos a nuestro $HOME usando la ruta absoluta, es decir, desde /
vinuesa@ivory:~/cursos/intro2bioinfo$ cd /home/vinuesa
vinuesa@ivory:~$ pwd
/home/vinuesa
vinuesa@ivory:~$
```

Introducción al biocómputo en sistemas UNIX/Linux

· Comandos y conceptos básicos

2. - trabajando con directorios: rutas absolutas y relativas

• Siempre podemos regresar a nuestro home tecleando cualquiera de las siguientes órdenes:

iniciamos desde este directorio: /home/vinuesa/cursos/intro2bioinfo

```
 # 1) vamos a nuestro $HOME usando cd sin argumento
vinuesa@ivory:~/cursos/intro2bioinfo$ cd
vinuesa@ivory:~$ pwd
/home/vinuesa
vinuesa@ivory:~$ cd cursos/intro2bioinfo/  # volvemos al dir perl4bioinfo

# 2) vamos a nuestro $HOME usando cd ~
vinuesa@ivory:~/cursos/intro2bioinfo$ cd ~
vinuesa@ivory:~$ pwd
/home/vinuesa
vinuesa@ivory:~$ cd cursos/intro2bioinfo/  # volvemos al dir perl4bioinfo

# 3) haciendo cd $HOME
vinuesa@ivory:~$ cd $HOME
vinuesa@ivory:~$ pwd
/home/vinuesa
```

Introducción al biocómputo en sistemas UNIX/Linux

· Comandos y conceptos básicos

2. - trabajando con directorios: rutas absolutas y relativas

• Más ejemplos del uso de rutas absolutas y relativas:

estamos en: /home/vinuesa/cursos/perl4bioinfo

```
# 1) queremos ver contenido del directorio /usr/bin
vinuesa@ivory:~$ ls /usr/bin  # [ó: ls ../../../../usr/bin]

# 2) queremos copiar el archivo hello_shell.sh a $HOME/bin
vinuesa@ivory:~$ cp hello_shell.sh $HOME/bin
[ó: cp hello_shell.sh ../../bin]
[ó: cp hello_shell.sh ~/bin]
```

El usuario decide qué es más práctico, usar rutas relativas o absolutas. El punto está en que podemos ejecutar cualquier comando que lee o escribe archivos desde un directorio diferente al actual, y poner el resultado del comando en el directorio que nos convenga.

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

2. - trabajando con directorios: generación de directorios

Podemos en UNIX/Linux ejecutar varios comandos en una sola línea, separándolos con ;
Si la línea es muy larga, podemos introducir saltos de línea (return), escapándolo con \

mkdir => genera directorio; no dejar espacios en blanco en el nombre, usar guión bajo!!!
cd => cambia al directorio
touch => genera archivos vacíos, para nuestra práctica
mv => mueve o renombra un archivo o directorio

```
vinuesa@ivory:~$ mkdir practicas_UNIX; cd practicas_UNIX; \
touch file1.txt file2.txt file3.txt; cd ../;  ls -laF practicas_UNIX;
total 0
drwxrwxr-x 2 vinuesa vinuesa 4096 Aug  8 11:52 ./
drwxr-xr-x 6 vinuesa vinuesa 4096 Aug  8 11:52 ../
-rw-rw-r-- 1 vinuesa vinuesa    0 Aug  8 11:52 file1.txt
-rw-rw-r-- 1 vinuesa vinuesa    0 Aug  8 11:52 file2.txt
-rw-rw-r-- 1 vinuesa vinuesa    0 Aug  8 11:52 file3.txt

vinuesa@ivory:~$ ls -d practicas_UNIX/
practicas_UNIX/
vinuesa@ivory:~$ ls practicas_UNIX/
file1.txt  file2.txt file3.txt
vinuesa@ivory:~$ mkdir borrame; mv practicas_UNIX/ borrame/
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

2. - trabajando con directorios: copiado y borrado de directorios

```
# estamos en /home/vinuesa y revisamos el contenido del dir
vinuesa@ivory:~$ ls
borrame  cursos

# veamos el contenido de borrame
vinuesa@ivory:~$ ls -F borrame/
practicas_unix/

# ahora copiamos practicas_unix al directorio actual
($HOME)
vinuesa@ivory:~$ cp -r borrame/practicas_unix .

# borramos el directorio borrame
vinuesa@ivory:~$ rm -rf borrame

# vemos contenido del dir practicas_unix y borramos todos
los archivos que contiene
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

2. - trabajando con directorios: copiado y borrado de directorios

```
# estamos en /home/vinuesa y revisamos el contenido del dir
# practicas_unix y borramos todos los archivos que contiene

vinuesa@ivory:~$ ls -lF practicas_unix/
total 0
-rw-rw-r-- 1 vinuesa vinuesa 0 Aug  8 12:15 file1.txt
-rw-rw-r-- 1 vinuesa vinuesa 0 Aug  8 12:15 file2.txt
-rw-rw-r-- 1 vinuesa vinuesa 0 Aug  8 12:15 file3.txt

vinuesa@ivory:~$ rm practicas_unix/*.*
vinuesa@ivory:~$ ls -lF practicas_unix/
total 0
vinuesa@ivory:~$

# una vez vacío, podemos usar rmdir para eliminar el dir vacío
vinuesa@ivory:~$ rmdir practicas_unix/ # o usar rm -rf pract*
vinuesa@ivory:~$ ls
Cursos
vinuesa@ivory:~$
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos**

2. - trabajando con archivos y directorios: resumen

- Podemos ejecutar varios comandos en una sola línea, separándolos con **;**
- Si la línea es muy larga, podemos introducir saltos de línea, escapándolo con ****

mkdir => genera directorio; no dejar espacios en blanco en el nombre, usar guión bajo!!!

cd => cambia al directorio

touch => genera archivos vacíos, para nuestra práctica

mv file2 dir1; mv file3 file1 => mueve o renombra un archivo o directorio

cp file1 dir1 => copia file1 a dir1

cp –r dir1 dir2 => copia dir1 y su contenido (-r recursivamente) a dir2

rm dir1/file1 => borra file1 en dir1/

rm –rf dir1 dir2 => elimina los directorios dir1 y dir2

find . –type d => muestra recursivamente los subdirectorios del directorio indicado

ls –d dir* => muestra sólo los nombres de los directorios que empiecen por dir

ls dir1 => muestra el contenido del directorio dir1

tar (-cvzf|-tvzf|-xvzf) => crea, lista contenidos o extrae archivos/dirs de un tarro comprimido

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos – I/O**

3.– Standard Input / Standard Output

```
# salida del comando who a STDOUT (pantalla por lo general)
-bash-3.1$ who
root     pts/1        2009-03-17 12:39 (:0.0)
heladia  pts/3        2009-05-18 23:55 (itzamna.ccg.unam.mx)
vinuesa  pts/4        2009-06-20 19:36 (uxmal.ccg.unam.mx)
rzayas   pts/7        2009-06-02 10:57 (kay.ccg.unam.mx)
rzayas   pts/8        2009-06-02 10:58 (kay.ccg.unam.mx)

# redireccionamos la salida de who a un archivo con >
-bash-3.1$ who > users.out

# podemos ver el contenido de users.out con cat ó less ó more
-bash-3.1$ cat users.out
root     pts/1        2009-03-17 12:39 (:0.0)
heladia  pts/3        2009-05-18 23:55 (itzamna.ccg.unam.mx)
vinuesa  pts/4        2009-06-20 19:36 (uxmal.ccg.unam.mx)
rzayas   pts/7        2009-06-02 10:57 (kay.ccg.unam.mx)
rzayas   pts/8        2009-06-02 10:58 (kay.ccg.unam.mx)
-bash-3.1$
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos – I/O**

3.– Standard Input / Standard Output

```
# uso del comando cat para conCATenar texto o archivos al final de otro usando >>
# tecleamos el comando y enter; escribimos el texto y salicmos con Ctrl-D
-bash-3.1$ cat >> users.out
estas son líneas adicionadas al final del archivo gracias a '>>'
y una segunda linea
y una tercera

# veamos el contenido de users.out con less
bash-3.1$ less users.out # (equivalente a less < users.out)
root     pts/1        2009-03-17 12:39 (:0.0)
heladia  pts/3        2009-05-18 23:55 (itzamna.ccg.unam.mx)
vinuesa  pts/4        2009-06-20 19:36 (uxmal.ccg.unam.mx)
rzayas   pts/7        2009-06-02 10:57 (kay.ccg.unam.mx)
rzayas   pts/8        2009-06-02 10:58 (kay.ccg.unam.mx)
estas son lineas adicionadas al final del archivo gracias a '>>'
y una segunda linea
y una tercera
(END)

# de esta manera añadimos el contenido de file1 al final de users.out
-bash-3.1$ cat file1 >> users.out
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos – I/O**

3.– Standard Input / Standard Output – más ejemplos de I/O con cat

```
# el comando echo imprime a STOUT su argumento(s); redirigimos salida a archivo1.txt
# simplemente para tener un archivo de texto con contenido

-bash-3.1$ echo 'linea uno' > archivo1.txt
-bash-3.1$ less archivo1.txt
linea uno

# lo mismo lo podemos hacer con el comando cat, como ya hemos visto anteriormente
-bash-3.1$ cat > archivo2.txt
linea dos
^D # usen CTRL-D para interrumpir la escritura a archivo2.txt con cat
-bash-3.1$ cat archivo2.txt archivo1.txt # concatena ambos archivos
linea dos
linea uno

# redirigimos con '>' la salida de cat a un archivo
-bash-3.1$ cat archivo2.txt archivo1.txt > archivos2-1_concatenados.txt

-bash-3.1$ cat archivos2-1_concatenados.txt
linea dos
linea uno
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos – Pipes (tuberías)**

4.– pipes '|' – conecta la salida (stdout) de un comando directamente con la entrada estándar (stdin) de otro comando, filtrando la salida del primero por el segundo programa. Típicos programas de filtrado son grep, cut, sort, sed, awk, head, tail …

```
# veamos estos comandos de filtrado en acción usando el archivo
# /etc/passwd
vinuesa@ivory:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
...

# cuantas entradas o líneas hay en dicho archivo? => contamos con wc
vinuesa@ivory:~$ cat  /etc/passwd | wc
    115     147    5876  # líneas   palabras  caracteres

# con opción –l cuenta sólo las líneas
vinuesa@ivory:~$ cat /etc/passwd | wc -l
115
```

Introducción al biocómputo en sistemas UNIX/Linux

- Comandos y conceptos básicos – herramientas de filtrado usadas en pipes

4.– Veremos las siguientes herramientas: grep, cut, sort y uniq (exploren uds. tr y sed)

- **grep**

```
-bash-3.1$ grep --help  # selección de opciones (hay muchas más)
Usage: grep [OPTION]... PATTERN [FILE] ...
Search for PATTERN in each FILE or standard input.
Example: grep -i 'hello world' menu.h main.c

Regexp selection and interpretation:
  -P, --perl-regexp         PATTERN is a Perl regular expression
  -e, --regexp=PATTERN      use PATTERN as a regular expression
  -f, --file=FILE           obtain PATTERN from FILE
  -i, --ignore-case         ignore case distinctions
  -w, --word-regexp         force PATTERN to match only whole words
  -x, --line-regexp         force PATTERN to match only whole lines

Miscellaneous:
  -v, --invert-match        select non-matching lines

Output control:
  -n, --line-number         print line number with output lines
      --line-buffered       flush output on every line
  -L, --files-without-match only print FILE names containing no match
  -l, --files-with-matches  only print FILE names containing matches
```

Introducción al biocómputo en sistemas UNIX/Linux

- Comandos y conceptos básicos – herramientas de filtrado usadas en pipes

4.– Veremos las siguientes herramientas: grep, cut, sort y uniq (exploren uds. tr y sed)

- **cut**

```
-bash-3.1$ cut --help # selección de opciones (hay muchas más)

Usage: cut [OPTION]... [FILE]...
Print selected parts of lines from each FILE to standard output.

Mandatory arguments to long options are mandatory for short options too.
  -c, --characters=LIST   select only these characters
  -d, --delimiter=DELIM   use DELIM instead of TAB for field delimiter
  -f, --fields=LIST       select only these fields;  also print any line
                            that contains no delimiter character, unless
                            the -s option is specified
With no FILE, or when FILE is -, read standard input.
```

Introducción al biocómputo en sistemas UNIX/Linux

- Comandos y conceptos básicos – herramientas de filtrado usadas en pipes

4.– Veremos las siguientes herramientas: grep, cut, sort y uniq (exploren uds. tr y sed)

- **sort**

```
-bash-3.1$ sort --help # selección de opciones (hay algunas más)
Usage: sort [OPTION]... [FILE]...
Write sorted concatenation of all FILE(s) to standard output.

Mandatory arguments to long options are mandatory for short options too.
Ordering options:

  -b, --ignore-leading-blanks  ignore leading blanks
  -d, --dictionary-order      consider only blanks and alphanumeric characters
  -f, --ignore-case           fold lower case to upper case characters
  -g, --general-numeric-sort  compare according to general numerical value
  -i, --ignore-nonprinting    consider only printable characters
  -M, --month-sort            compare (unknown) < `JAN' < ... < `DEC'
  -n, --numeric-sort          compare according to string numerical value
  -r, --reverse               reverse the result of comparisons

Other options:

  -m, --merge               merge already sorted files; do not sort
  -o, --output=FILE         write result to FILE instead of standard output
  -t, --field-separator=SEP  use SEP instead of non-blank to blank transition

  -u, --unique              with -c, check for strict ordering;
                            without -c, output only the first of an equal run
```

Introducción al biocómputo en sistemas UNIX/Linux

- Comandos y conceptos básicos – herramientas de filtrado usadas en pipes

- **uniq**

```
-bash-3.1$ uniq --help
Usage: uniq [OPTION]... [INPUT [OUTPUT]]
Discard all but one of successive identical lines from INPUT (or
standard input), writing to OUTPUT (or standard output).

Mandatory arguments to long options are mandatory for short options too.
  -c, --count           prefix lines by the number of occurrences
  -d, --repeated        only print duplicate lines
  -D, --all-repeated[=delimit-method] print all duplicate lines
                        delimit-method={none(default),prepend,separate}
                        Delimiting is done with blank lines.
  -f, --skip-fields=N   avoid comparing the first N fields
  -i, --ignore-case     ignore differences in case when comparing
  -s, --skip-chars=N    avoid comparing the first N characters
  -u, --unique          only print unique lines
  -w, --check-chars=N   compare no more than N characters in lines
      --help     display this help and exit
      --version  output version information and exit

A field is a run of whitespace, then non-whitespace characters.
Fields are skipped before chars.

Report bugs to <bug-coreutils@gnu.org>.
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos – Pipes (tuberías)**

comandos de filtrado (grep, wc) en acción usando el archivo /etc/passwd (cont.)

```
# cuantas entradas en /etc/passwd corresponden a cuentas de usuario ($HOME)
vinuesa@ivory:~$ cat /etc/passwd | grep home | wc -l
89

# cuantas entradas en /etc/passwd corresponden a cuentas de usuario
# ($HOME) que NO correspondan a curso?
vinuesa@ivory:~$ cat /etc/passwd | grep home | grep –v curso | wc -l
39

# cuantos usuarios usan el bash y cuántos usan otro shell?
vinuesa@ivory:~$ cat /etc/passwd | grep home | grep -c bash
81
vinuesa@ivory:~$ cat /etc/passwd | grep home | grep -vc bash
8
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos – Pipes (tuberías)**

comandos de filtrado (grep, cut, sort, uniq) en acción usando el archivo /etc/passwd (cont.)

```
# muestra los usuarios que no usan bash como shell ordenados alfabéticamente
vinuesa@ivory:~$ cat /etc/passwd |grep home |grep -v bash |cut –d: -f1,7 |sort
alemc:/bin/tcsh
cemg:/bin/tcsh
javiermb:/bin/tcsh
jmanuel:/bin/tcsh
#nan:/bin/tcsh
syslog:/bin/false
#viri:/bin/tcsh
zuemy:/bin/tcsh

# genera estadísticas de uso de shell para todas las entradas en /etc/passwd
vinuesa@ivory:~$ cat /etc/passwd | cut -d: -f7 | sort | uniq -c
     82 /bin/bash
      6 /bin/false
     17 /bin/sh
      1 /bin/sync
      7 /bin/tcsh
      2 /usr/sbin/nologin
```

Introducción al biocómputo en sistemas UNIX/Linux

- **Comandos y conceptos básicos – Introducción a la programación en bash**

- Uso de variables e impresión de su contenido desde línea de comandos

```
-bash-3.1$  STR='Hello World'; echo $STR
```

- Uso de condicionales y su ejecución desde un "script". Hacerlo ejecutable con chmod +x script

```
#!/bin/bash
# program: simple_conditionals.sh

# 1)check that two arguments are passed to the script from the command
line
if [ $# != 2 ]; then
   echo "# $0 needs two string arguments to compare"
   echo "# usage: $0 string1 string2"
   exit 1
fi

# 2) assign positional parameters to named variables
string1="$1"
string2="$2"

# 3) make the string comparisons within an if-else-fi structure
if [ "$string1" = "$string2" ]; then
    echo "$string1 = $string2, therefore expression evaluated as true"
else
    echo "$string1 != $string2, therefore expression evaluated as false"
fi
```

Introducción al biocómputo en sistemas UNIX/Linux

- Comandos y conceptos básicos

1. - trabajando con archivos: descarga de archivos y tarros comprimidos
2. de la web con wget y su descompresión y extacción con tar

```
# estamos en $HOME/practicas_unix y queremos descargar datos.tgz
# de http://www.ccg.unam.mx/~vinuesa/perl4bioifno/data
wget -c http://www.ccg.unam.mx/~vinuesa/perl4bioifno/data/datos.tgz

# para ver el contenido del tarro sin extraer su contenido
tar -tvzf datos.tgz # t=lisT contents v=Verbose z=Zipped f=File

# uso de tar y gunzip para descomprimir tarro y extraer su contenido
tar -xvzf datos.tgz # x=eXtract v=Verbose z=Zipped f=File

# uso de tar y gunzip para extraer un archivo particular de un tarro
# comprimido
tar -xvzf datos.tgz archivo1 # x=eXtract v=Verbose z=Zipped f=File

# generación de un tarro comprimido, al que metemos dir1 dir2
# y todos los archivos *pl *sh y *tab
tar -cvzf nombre_de_mi_tarro.tgz dir1/ dir2/ *pl *sh *tab
# c=Create v=Verbose z=Zipped f=File
```

Introducción al biocómputo en sistemas UNIX/Linux

- Comandos y conceptos básicos – miscelánea de comandos muy importantes

- generar un tarro (tar file o "carpeta") , añadirle archivos, comprimir el tarro y enviarlo a una máquina remota mediante scp:

```
# vamos a generar un tarro comprimido con gzip, que contenga todos los *.sh scripts
# presentes en el directorio actual

pablo@Tenerife:~/WinXP/Cursos/TLEM11$ ls *sh
align_seqs_clustal_or_muscle.sh  find_directories.sh  simple_conditionals.sh
pablo@Tenerife:~/WinXP/Cursos/TLEM11$ tar -cvzf sample_bash_scripts.tgz *sh
align_seqs_clustal_or_muscle.sh
find_directories.sh
simple_conditionals.sh


# ahora vamos a copiar el tarro comprimido mediante scp de mi una máquina a otra

pablo@Tenerife:~/WinXP/Cursos/TLEM11$ ls *tgz
sample_bash_scripts.tgz
pablo@Tenerife:~/WinXP/Cursos/TLEM11$ scp sample_bash_scripts.tgz \
vinuesa@132.248.34.3:/home/vinuesa/public_html/tlem
sample_bash_scripts.tgz          100% 1485     1.5KB/s   00:00
pablo@Tenerife:~/WinXP/Cursos/TLEM11$
```

Introducción al biocómputo en sistemas UNIX/Linux

- Comandos y conceptos básicos – miscelánea de comandos muy importantes

- bajar archivos de la web desde la terminal:

```
# si no lo han hecho aún, generen los siguientes directorios en su $HOME
 mkdir practicas_unix bin seq_data;
 [vinuesa@xibalba ~]$ cd bin/
[vinuesa@xibalba bin]$ wget –c \
http://www.ccg.unam.mx/~vinuesa/tlem/shell_scripts/sample_bash_scripts.tgz
--19:56:13--  http://www.ccg.unam.mx/~vinuesa/tlem/shell_scripts/sample_bash_scripts.tgz
          => `sample_bash_scripts.tgz'
Resolving www.ccg.unam.mx... 132.248.34.17
Connecting to www.ccg.unam.mx|132.248.34.17|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1,485 (1.5K) [application/x-gzip]

100%[=====================================================>] 1,485         --.--K/s
19:56:13 (59.26 MB/s) - `sample_bash_scripts.tgz' saved [1485/1485]
# ahora podemos desempacar y descomprimir los archivos contenidos en el tarro
comprimido
[vinuesa@xibalba bin]$ tar xvzf sample_bash_scripts.tgz
align_seqs_clustal_or_muscle.sh
find_directories.sh
simple_conditionals.sh
```

Introducción al biocómputo en sistemas UNIX/Linux

- Comandos y conceptos básicos – miscelánea de comandos muy importantes

- bajar archivos de la web desde la terminal:

```
# y ahora vamos a bajar un archivo de configuración ".bashrc" del ambiente para
#  que puedan trabajar más agusto en su casa $HOME
# vayan a su directorio home
 cd ~;
 [vinuesa@xibalba ~]$ wget -c \
http://www.ccg.unam.mx/~vinuesa/tlem/docs/sample_bashrc.txt
Resolving www.ccg.unam.mx... 132.248.34.17
Connecting to www.ccg.unam.mx|132.248.34.17|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1,812 (1.8K) [text/plain]

100%[=============================================>] 1,812         --.--K/s

20:51:52 (63.30 MB/s) - `sample_bashrc.txt' saved [1812/1812

# ahora lo renombramos a .bashrc
[vinuesa@xibalba ~]$ mv sample_bashrc.txt .bashrc

# y activamos el ambiente, ejecutando el script de configuración
[vinuesa@xibalba ~]$ source .bashrc .profile
```

Introducción al biocómputo en sistemas UNIX/Linux

- Comandos y conceptos básicos – Introducción a la programación en bash
  - Uso de bucles y condicionales desde un script

```
#!/bin/bash

# program: find_directories.sh

# 1) inicializamos variables; var=$(comando) guarda salida de comando UNIX en var
workdir=$(pwd)
counter=0

# 2) recorremos cada archivo/dir en pwd; si es un dir, imprimimos e incrementamos
contador
for i in $( ls ); do
    if [ -d $i ]; then
         echo found directory: $i
        let counter=counter+1
    fi
done


# 3) evaluamos el contenido de la variable contador e imprimimos resumen
correspondiente
if [ $counter = 0 ]; then
    echo "There are no directories in $workdir"
    exit 0
elif [ $counter > 0 ]; then
      echo "There are There are $counter directories in $workdir"
fi
```

Introducción al biocómputo en sistemas UNIX/Linux

- Comandos y conceptos básicos – Introducción a la programación en bash
  - Uso de bucles y condicionales desde la línea de comandos

```
for i in $(ls); do if [ -f $i ]; then echo file $i; elif [ -d $i ]; then
echo dir $i; fi; done
```

```
# SALIDA
file find_directories.sh
file lista_login_accounts.tab
file lista_login_accounts.txt
file parse_seleccionados1_html.1liners
file samble_bashrc.txt
file seleccionados1.html
file simple_conditionals.sh
dir tmp
```

- El uso de bucles y condicionales desde la línea de comandos puede ser muy útil. Este ejemplo alinea todos los archivos fasta con terminación fna presentes en el directorio actual, usando muscle

```
for file in *.fna; do muscle < $file > ${file%.fna}_muscle_alignment.fna; done
```

```
for file in *.fna
do
     muscle < $file > ${file%.fna}_muscle_alignment.fna
done
```

Introducción al biocómputo en sistemas UNIX/Linux

• Referencias sobre Shell y Bash libremente disponibles en la web

una lista de comandos y ejemplos de uso los encuentras aquí:

- http://en.wikipedia.org/wiki/List_of_Unix_programs

Estos son unos tutoriales que si los estudias te harán un experto programador de Bash. Comienza por el primero de ellos, que es muy corto. Los últimos 2 son tutoriales avanzados sobre Bash scripting

- http://tldp.org/HOWTO/Bash-Prog-Intro-HOWTO.html
- http://www.ee.surrey.ac.uk/Teaching/Unix/
-http://tldp.org/LDP/Bash-Beginners-Guide/html/index.html
- http://www.museum.state.il.us/ismdepts/library/linuxguides/abs-guide/index.html

Si buscas libros de referencia, mira aquí

-http://en.wikibooks.org/wiki/Guide_to_Unix
-http://www.ee.surrey.ac.uk/Teaching/Unix/books-uk.html

Introducción al biocómputo en sistemas UNIX/Linux

Vean también: http://en.wikipedia.org/wiki/List_of_Unix_programs

• Comandos y conceptos básicos – una selección de comandos

1.– moviéndonos por el sistema y trabajando con archivos

- **ls** – lista información sobre archivos y directorio
- **cat** – despliega contenidos de un archivo o los concatena
- **less** – un paginador que despliegua el contenido de un archivo página a página
- **wc** – cuenta líneas, palabras y caracteres
- **cp** – copia archivos
- **mv** – renombra o mueve archivos
- **rm** – elminia un archivo o directorio
- **chmod** – cambia permisos de archivos y directorios
- **tar** – crea un "jarro" de archivos y/o directorios
- **zip** – comprime archivos
- **head** – despliega la cabecera del archivo
- **tail** – despliega la cola del archivo
- **file** – muestra la clasificación de un archivo

2.– trabajando con directorios

- **pwd** – print working directory
- **mkdir** – crea un directorio
- **cd** – cambia de directorio
- **rmdir** – elimina directorio (sólo si están vacíos)
- **find** - busca archivos y directorios en base a características definidas por el usuario

Introducción al biocómputo en sistemas UNIX/Linux

Vean también: http://en.wikipedia.org/wiki/List_of_Unix_programs

• Comandos y conceptos básicos

2.– trabajando con directorios (cont.)

- **df** – despliega información de uso de disco
- **du** – despliega información de uso de disco por archivo
- **ln** – genera una liga simbólica a un archivo o directorio

3.– trabajando con texto

- **grep** – busca patrones en archivos
- **cut** – selecciona caracteres o campos de archivos
- **sort** – ordena y/o conjunta archivos
- **uniq** – muestra líneas únicas
- **tr** – reemplaza caracteres indicados
- **sed** – edición no interactiva de archivos
- **awk** – filtrado de archivos por campos
- **vim** – editor programable estándar de Linux
- **nedit** – editor con ambiente gráfico

4.– trabajando con procesos y comandos

- **top** – despliegue dinámico de estatus de procesos
- **ps** – despliegue de estatus de procesos
- **kill** – mata procesos por PID
- **nice** – cambia la prioridad de un comando
- **which** – muestra dónde se ubica un comando en el PATH
- **history** – muestra historial de comandos

Introducción al biocómputo en sistemas UNIX/Linux

Vean también: http://en.wikipedia.org/wiki/List_of_Unix_programs

• Comandos y conceptos básicos

5.– trabajando en la red con directorios y archivos remotos

- **ssh** – ejecuta comandos de manera segura en un sistema remoto
- **scp** – copia de manera segura uno o más archivos desde o hacia un sistema remoto
- **sftp** – copia de manera segura archivos desde un sistema remoto hacia una máquina local
- **wget** – descarga archivos desde una URL

6.– comandos para compilación de programas

- **configure** – cofigura código fuente de manera automática
- **gcc** – compila programas escritos en C y C++
- **make** – utilidad para construir binarios y librerías a partir de código fuente mediante la lectura de instrucciones contenidas en archivos llamdos makefiles que especifican cómo derivar el programa diana.