

# Programación con Shell Scripting: Sesión 6

Máster Universitario en Bioinformática



**Universidad**  
Internacional  
de Valencia

Dra. Paula Soler Vila  
[paula.solerv@professor.universidadviu.com](mailto:paula.solerv@professor.universidadviu.com)

De:  
 Planeta Formación y Universidades

## Aspectos a tratar

volver a mirar los 25 primeros mins

1 Ejercicio **GENCODE**: procesamiento básico de datos

2 Presentación de la Actividad 1

3 Introducción al *Shell* scripting.

- Usos y aplicaciones
- Editores de texto
- Tipos de *Shell*
  - Comando *chsh*

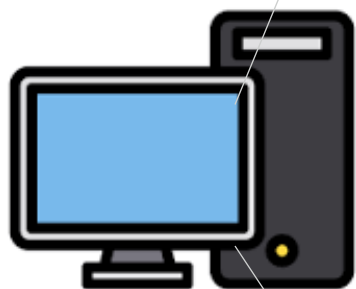
Comando **imp** de examen: `tr "\t" "\n"`  
`<` para REDIRIGIR LA  
**ENTRADA (EL <)** y modificar los  
 tabuladores por guiones

Si pones **history** puedes ver  
 todos los comandos que has  
 utilizado.

Si pones **!** y el n° del comando, te  
 da el resultado como si  
 estuvieras poniendo ese  
 comando

Atajo: si utilizas **control+r** sale  
 buscar: pones una palabra y te  
 las busca entre los comandos  
 que has usado. Después con  
**control+r** vas pasando de unas a  
 otras

# Combinando lo aprendido



[Human](#) [Mouse](#) [How to access data](#) [FAQ](#) [Documentation](#) [About us](#)



Human

## Release 45 (GRCh38.p14)

- [Statistics of this release](#)
- [More information about this assembly](#) (including patches, scaffolds and haplotypes)
- [Go to GRCh37 version of this release](#)

### GTF / GFF3 files

Content	Regions	Description	Download
Comprehensive gene annotation	CHR	• It contains the comprehensive gene annotation on the reference chromosomes only	<a href="#">GTF</a> <a href="#">GFF3</a>
Comprehensive gene annotation	ALL	• It contains the comprehensive gene annotation on the reference chromosomes, scaffolds, assembly patches and alternate loci (haplotypes)	<a href="#">GTF</a> <a href="#">GFF3</a>
Comprehensive gene annotation	PRI	• It contains the comprehensive gene annotation on the primary assembly (chromosomes and scaffolds) sequence regions	<a href="#">GTF</a> <a href="#">GFF3</a>
Basic gene annotation	CHR	• It contains the basic gene annotation on the reference chromosomes only • This is a <b>subset</b> of the corresponding comprehensive annotation, including only those transcripts tagged as 'basic' in every gene • This is the <b>main annotation file</b> for most users	<a href="#">GTF</a> <a href="#">GFF3</a>
Basic gene annotation	ALL	• It contains the basic gene annotation on the reference chromosomes, scaffolds, assembly patches and alternate loci (haplotypes) • This is a <b>subset</b> of the corresponding comprehensive annotation, including only those transcripts tagged as 'basic' in every gene • This is a <b>superset</b> of the main annotation file	<a href="#">GTF</a> <a href="#">GFF3</a>

<https://www.encodegenes.org/human/>

## Aspectos a tratar

- 1 Ejercicio **GENCODE**: procesamiento básico de datos
- 2 Presentación de la Actividad 1
- 3 Introducción al *Shell* scripting.
  - Usos y aplicaciones
  - Editores de texto
  - Tipos de *Shell*
    - Comando *chsh*

# Actividad 1

## Actividad 1.- Manipulación de Archivos en Formato BED

El objetivo de esta actividad es que el estudiante adquiera habilidades en la manipulación y formateo de archivos usando comandos de Linux, aprendidos en las sesiones teóricas de la asignatura. En particular, se enfocará en el formato **BED** (*Browser Extensible Data*) que se utiliza extensamente en bioinformática para almacenar regiones genómicas, como coordenadas y anotaciones asociadas. Este formato se caracteriza por presentar los datos en forma de columnas separadas por espacios o tabuladores.

### Instrucciones de entrega

- La entrega se realizará a través del Campus VIU en un archivo único en formato **PDF** utilizando este documento como plantilla. Recuerde que las actividades a realizar están resaltadas en negrita.
- Incluya el código empleado, capturas de pantalla con su usuario (agregando el *prompt* completo) y resolución máxima.
- Proporcione explicaciones **claras y concisas** de los comandos utilizados. Si los comandos empleados no se explican brevemente, el valor de la pregunta será penalizado a la mitad.
- Reporte solo una opción o forma para resolver cada una de las preguntas propuestas.

Are  
You  
Ready?



# Actividad 1

## A tener en cuenta



### Prueba aplicativa 1



Acceso a la entrega de la actividad (2 intentos)

Archivos adjuntos:

- human\_coordinates\_1.bed (45,321 KB)
- human\_coordinates\_2.bed (45,39 KB)
- selected\_genes.txt (2,886 KB)
- Actividad 1.docx (229,304 KB)
- Actividad 1.pdf (534,258 KB)

### Actividad 1.- Manipulación y formateo de archivos: Formato BED

El objetivo de esta actividad es que el estudiante adquiera habilidades en la manipulación y formateo de archivos utilizando comandos de Linux, que han sido aprendidos a lo largo de las sesiones teóricas de la asignatura. En particular, se enfocará en el formato BED (*Browser Extensible Data*) que se utiliza extensamente en bioinformática para almacenar regiones genómicas, como coordenadas y anotaciones asociadas. Este formato se caracteriza por presentar los datos en forma de columnas separadas por espacios o tabuladores.

#### Instrucciones de entrega

- La entrega se realizará a través del Campus VIU en un archivo único en formato PDF utilizando este documento como plantilla. Recuerde que las actividades a realizar están resaltadas en negrita.

Incluya el código empleado, capturas de pantalla con su usuario (agregando el *prompt* completo) y resolución máxima.

- Proporcione explicaciones claras y concisas de los comandos utilizados. Si los comandos empleados no se explican brevemente, el valor de la pregunta será penalizado a la mitad.
- Reportar solo una opción/forma para resolver las distintas preguntas



Siempre emplear comandos para realizar las actividades.

- Nunca edición manual de los ficheros

# Actividad 1

# A tener en cuenta



## Prueba aplicativa 1



Acceso a la entrega de la actividad (2 intentos)

Archivos adjuntos:

- human\_coordinates\_1.bed (45,321 KB)
- human\_coordinates\_2.bed (45,39 KB)
- selected\_genes.txt (2,886 KB)
- Actividad 1.docx (229,304 KB)
- Actividad 1.pdf (534,258 KB)

## Actividad 1.- Manipulación y formateo de archivos: Formato BED

El objetivo de esta actividad es que el estudiante adquiera habilidades en la manipulación y formateo de archivos utilizando comandos de Linux, que han sido aprendidos a lo largo de las sesiones teóricas de la asignatura. En particular, se enfocará en el formato BED (*Browser Extensible Data*) que se utiliza extensamente en bioinformática para almacenar regiones genómicas, como coordenadas y anotaciones asociadas. Este formato se caracteriza por presentar los datos en forma de columnas separadas por espacios o tabuladores.

### Instrucciones de entrega

- La entrega se realizará a través del Campus VIU en un archivo único en formato PDF utilizando este documento como plantilla. Recuerde que las actividades a realizar están resaltadas en negrita.
- Incluya el código empleado, capturas de pantalla con su usuario (agregando el *prompt* completo) y resolución máxima.
- Proporcione explicaciones claras y concisas de los comandos utilizados. Si los comandos empleados no se explican brevemente, el valor de la pregunta será penalizado a la mitad.

Reporte



```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr gencode_annotation_v45]$ cat -n mini_bed.bed | head -n 3
1  chr1    11869   12227
2  chr1    11869   14409
3  chr1    11869   14409
```

El comando **cat** con la opción **-n** permite visualizar el contenido completo del archivo y numerar cada una de las líneas de este. Gracias a la tubería o pipe (**|**), el resultado de este comando es transferido al comando **head** que, mediante la opción **-n**, imprime las tres primeras líneas.



## Prueba aplicativa 1



Acceso a la entrega de la actividad (2 intentos)

Archivos adjuntos:

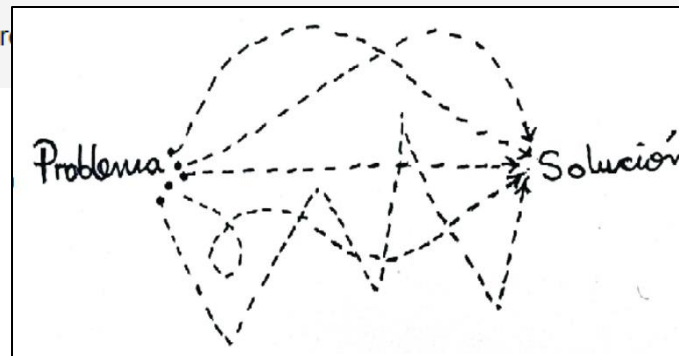
- human\_coordinates\_1.bed (45,321 KB)
- human\_coordinates\_2.bed (45,39 KB)
- selected\_genes.txt (2,886 KB)
- Actividad 1.docx (229,304 KB)
- Actividad 1.pdf (534,258 KB)

## Actividad 1.- Manipulación y formateo de archivos: Formato BED

El objetivo de esta actividad es que el estudiante adquiera habilidades en la manipulación y formateo de archivos utilizando comandos de Linux, que han sido aprendidos a lo largo de las sesiones teóricas de la asignatura. En particular, se enfocará en el formato BED (*Browser Extensible Data*) que se utiliza extensamente en bioinformática para almacenar regiones genómicas, como coordenadas y anotaciones asociadas. Este formato se caracteriza por presentar los datos en forma de columnas separadas por espacios o tabuladores.

### Instrucciones de entrega

- La entrega se realizará a través del Campus VIU en un archivo único en formato PDF utilizando este documento como plantilla. Recuerde que las actividades a realizar están resaltadas en negrita.
- Incluya el código empleado, capturas de pantalla con su usuario (agregando el *prompt* completo) y resolución máxima.
- **Proporcione explicaciones claras y concisas de los comandos utilizados.** Si los comandos empleados no se explican brevemente, el valor de la pregunta será penalizado a la mitad.
- **Reportar solo una opción/forma** para r



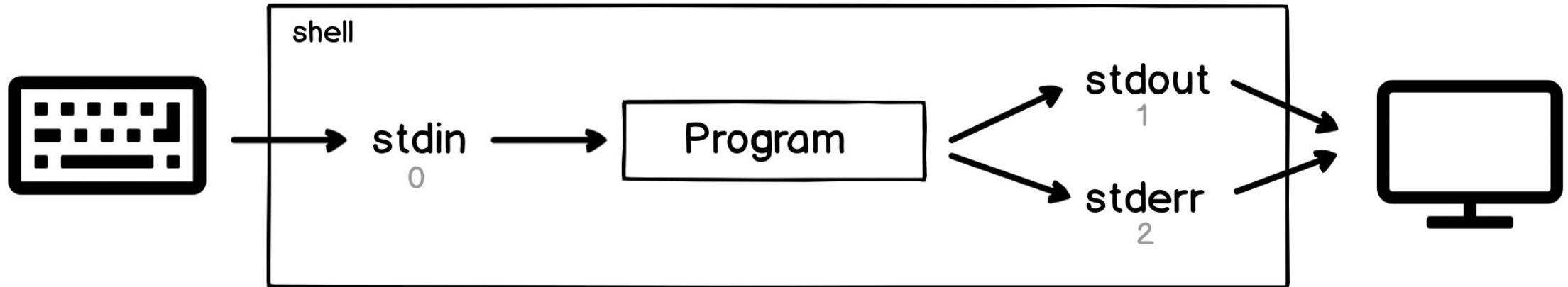


## Aspectos a tratar

- 1 Ejercicio **GENCODE**: procesamiento básico de datos
- 2 Presentación de la Actividad 1
- 3 Introducción al *Shell* scripting.
  - Usos y aplicaciones
  - Editores de texto
  - Tipos de *Shell*
    - Comando *chsh*

# Shell Scripting

Es un **intérprete** de comandos que provee una interfaz entre el usuario y el *kernel* del S.O



```
$ echo "Hello World!"  
Hello World!
```

# Shell Scripting

## Script example

```
clear
echo "Today's date is `date`"
echo
~stepp/cowsay `whoami`

echo "These users are currently connected:"
w -h | sort
echo

echo "This is `uname -s` on a `uname -m` processor."
echo

echo "This is the uptime information:"
uptime
echo
echo "That's all folks!"
```

7

```
$ echo "Hello World!"
Hello World!
```

Sucesión de instrucciones almacenadas en un archivo de texto plano que la *shell* irá ejecutando en un determinado orden y nos reportará el resultado final enviándolos a la salida estándar .



```
$ echo 'echo "Hello World!"' > new_script
$ cat new_script
echo "Hello World!"
```

# ¿Por qué usar Shell scripting?

1) Permite **automatizar** tareas repetitivas y así **ahorrar** tiempo.



Realizar una copia de seguridad del sistema diariamente / Aplicar un flujo de trabajo bioinformático

2) Proporcionan una secuencia de actividades bien **estructurada**, **modular** y formateada.

## FUNCIONES

```
function_name ()  
{  
  Command #1  
  Command #2  
}
```

```
trim_fastq ()  
{  
  input_file="$1"  
  output_file="$2"  
  adapter_sequence="AGATCGGAAGAGCACACGT"  
  cutadapt -a $adapter_sequence -o $output_file  
  $input_file  
}
```

```
trim_fastq input.fastq trimmed.fastq
```

Escribir las funciones sólo las define y **no** ejecuta los comandos contenidos. Para ejecutar una función, hay que **llamarla**

# ¿Por qué usar Shell scripting?

1) Permite **automatizar** tareas repetitivas y así **ahorrar** tiempo.



Realizar una copia de seguridad del sistema diariamente / Aplicar un flujo de trabajo bioinformático

2) Proporcionan una secuencia de actividades bien **estructurada**, **modular** y formateada.

## FUNCIONES

```
function_name ()  
{  
  Command #1  
  Command #2  
}
```

```
trim_fastq ()  
{  
  input_file="$1"  
  output_file="$2"  
  adapter_sequence="AGATCGGAAGAGCACACGT"  
  cutadapt -a $adapter_sequence -o $output_file  
  $input_file  
}
```

```
trim_fastq input.fastq trimmed.fastq
```

3) Permite introducir **valores dinámicos** mediante el uso de argumentos por línea de comandos.

Parámetros: valores dinámicos



# Comando bc



```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr raw]$ bc
bc 1.06.95
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006 Free Software Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
1+1
2
3 + 3
6
9 / 2
4
```

## PRACTIQUEMOS



# Comando bc (history)

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr raw]$ bc
bc 1.06.95
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006 Free Software Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
1+1
2
3 + 3
6
9 / 2
4
```

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr raw]$ bc -l
bc 1.06.95
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006 Free Software Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
9 / 2
4.5000000000000000
```



# ¿Por qué usar Shell scripting?

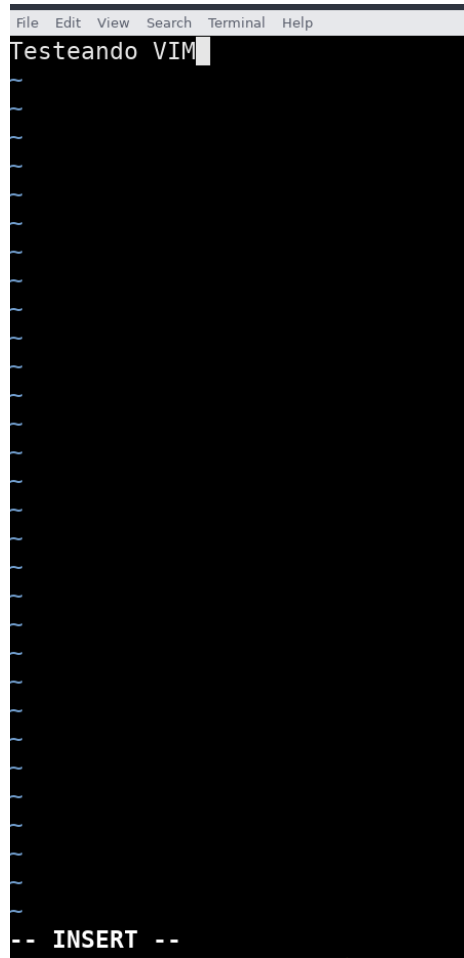
- 1) Permite **automatizar** tareas repetitivas y así **ahorrar** tiempo.
- 2) Proporcionan una secuencia de actividades bien **estructurada**, modular y formateada.
- 3) Permite introducir **valores dinámicos** mediante el uso de argumentos por línea de comandos.
- 4) Puede **simplificar** comandos complejos en una sola unidad en ejecución.
- 5) Una vez creado, se puede ejecutar cualquier cantidad de veces por **cualquier persona**.

**“Construye una vez y ejecuta muchas veces”.**

# ¿Cómo y dónde vamos a escribir Shell scripts?



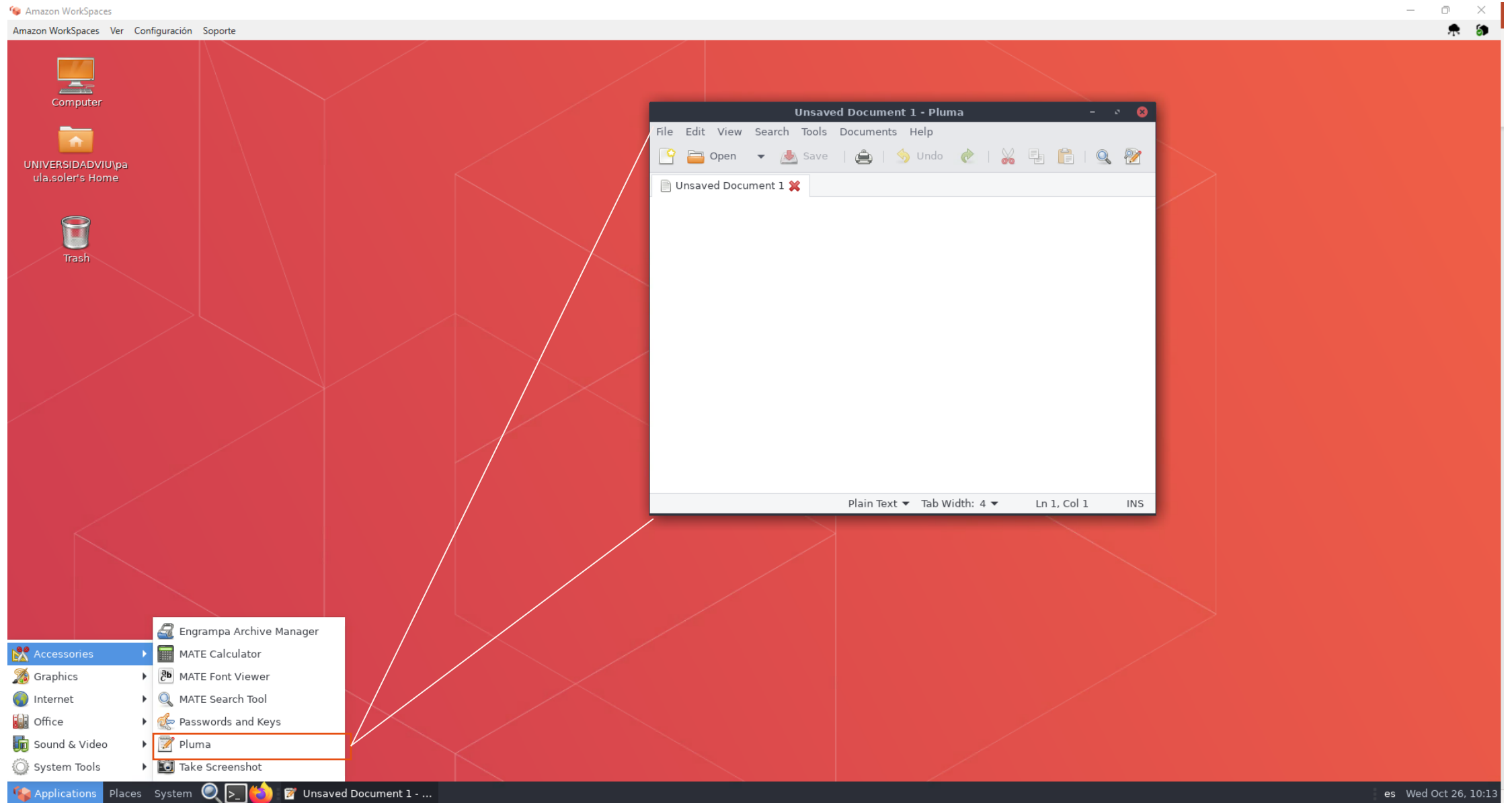
# Editores de texto: **VIM**



<https://vim-adventures.com/>

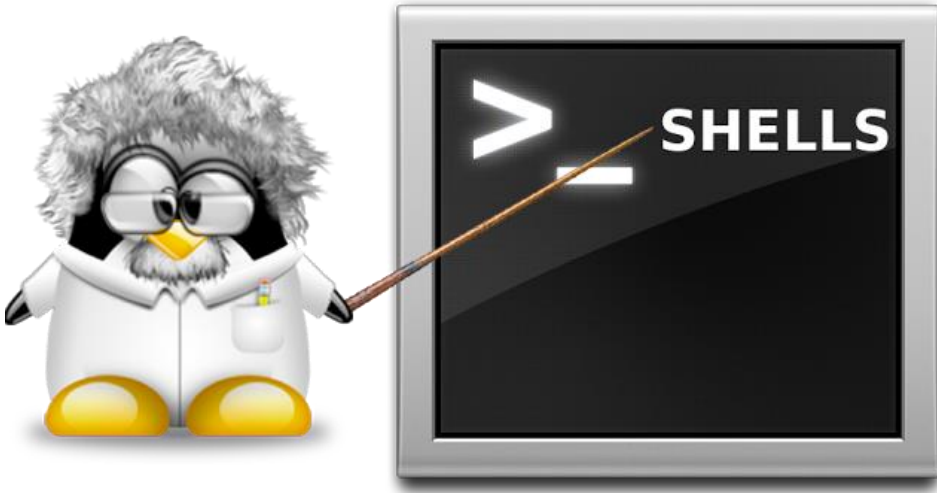


# Editores de texto: *PLUMA*



**¿Qué tipos de Shells podemos encontrar?**

# Tipos de **Shell**



- Los basados en el intérprete de Bourne:
  - sh (Bourne **Shell**)
  - ksh (**K**orn **Shell**)
  - **bash (Bourne Again Shell)**
- Los basados en el intérprete de C:
  - csh (**C Shell**)
  - tcsh (**T**enex **C Shell**)

# Shell del sistema

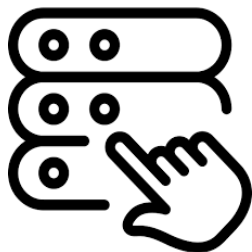


¿Qué *Shells* tenemos disponibles?

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ more /etc/shells
/bin/sh
/bin/bash
/usr/bin/sh
/usr/bin/bash
/usr/bin/zsh
/bin/zsh
```

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ sudo yum install tcsh
```

# Shell del sistema



¿Cuál es mi Shell?

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ echo $SHELL  
/bin/bash
```



¿Cuál es la versión de la Shell?

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ echo $BASH_VERSION  
4.2.46(2)-release
```

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ bash --version  
GNU bash, version 4.2.46(2)-release (x86_64-koji-linux-gnu)  
Copyright (C) 2011 Free Software Foundation, Inc.  
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
```

This is free software; you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.



# Shell del sistema

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ man bash
```

```
BASH(1)                                                    General Commands Manual                                BASH(1)

NAME
    bash - GNU Bourne-Again SHell

SYNOPSIS
    bash [options] [file]

COPYRIGHT
    Bash is Copyright (C) 1989-2011 by the Free Software Foundation, Inc.

DESCRIPTION
    Bash is an sh-compatible command language interpreter that executes commands read from the standard input or from a file. Bash also incorporates useful features from the Korn and C shells (ksh and csh).

    Bash is intended to be a conformant implementation of the Shell and Utilities portion of the IEEE POSIX specification (IEEE Standard 1003.1). Bash can be configured to be POSIX-conformant by default.

OPTIONS
    All of the single-character shell options documented in the description of the set builtin command can be used as options when the shell is invoked. In addition, bash interprets the following options when it is invoked:

    -c string If the -c option is present, then commands are read from string. If there are arguments after the string, they are assigned to the positional parameters, starting with $0.
    -i          If the -i option is present, the shell is interactive.
    -l          Make bash act as if it had been invoked as a login shell (see INVOCATION below).
    -r          If the -r option is present, the shell becomes restricted (see RESTRICTED SHELL below).
    -s          If the -s option is present, or if no arguments remain after option processing, then commands are read from the standard input. This option allows the positional parameters to be set when invoking an interactive shell.
    -D          A list of all double-quoted strings preceded by $ is printed on the standard output. These are the strings that are subject to language translation when the current locale is not C or POSIX. This implies the -n option; no commands will be executed.
    [-+]0 [shopt_option]
        shopt_option is one of the shell options accepted by the shopt builtin (see SHELL BUILTIN COMMANDS below). If shopt_option is present, -0 sets the value of that option; +0 unsets it. If shopt_option is not supplied, the names and values of the shell options accepted by shopt are printed on the standard output. If the invocation option is +0, the output is displayed in a format that may be reused as input.
    --          A -- signals the end of options and disables further option processing. Any arguments after the -- are treated as filenames and arguments. An argument of - is equivalent to --.
```

**¿Se puede cambiar el Shell de inicio de sesión?**

# Comando *chsh* (*change shell*)

Sintaxis básica

```
chsh <opciones> <usuario>
```

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ chsh -l
```

```
/bin/sh
```

```
/bin/bash
```

```
/usr/bin/sh
```

```
/usr/bin/bash
```

```
/usr/bin/zsh
```

```
/bin/zsh
```

```
/etc/shells
```

Modo interactivo

```
$ chsh
```

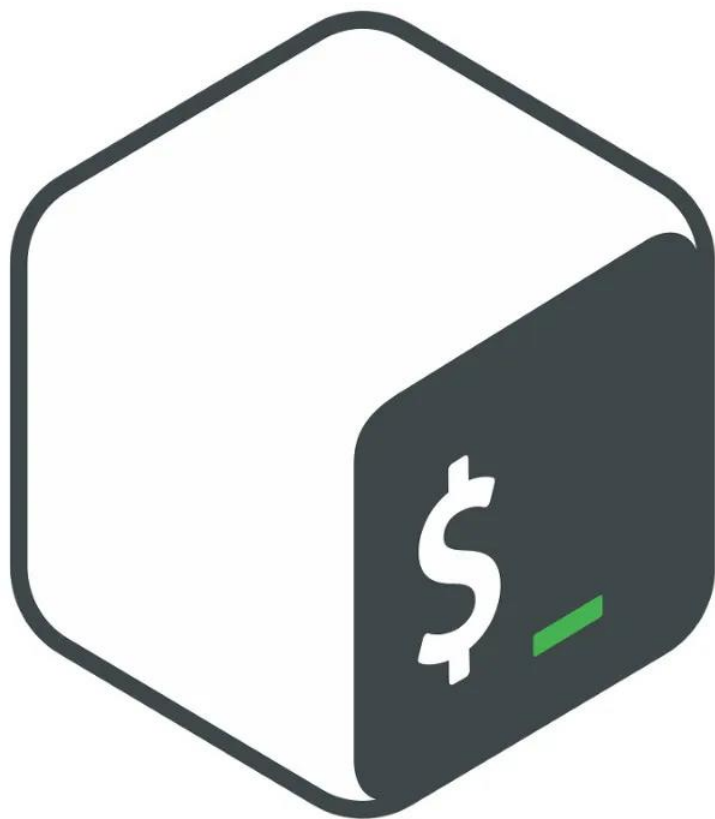
```
Changing the login shell for emma
```

```
Enter the new value, or press ENTER for the default
```

```
Login Shell [/bin/bash]: /usr/bin/zsh
```

Modo no interactivo (-s)

```
$ chsh -s /usr/bin/zsh
```



# BASH

THE BOURNE-AGAIN SHELL



viu

**Universidad**  
Internacional  
de Valencia

[universidadviu.com](http://universidadviu.com)

De:  
 Planeta Formación y Universidades