



Universidad Tecnológica Nacional  
Facultad Regional Buenos Aires  
Algoritmos y Estructuras de Datos  
Curso K2054  
Ing. Pablo D. Mendez  
Trabajo práctico Nro. 1

# Expresiones regulares y expresiones regulares extendidas en Bash

Fecha de entrega de enunciado: 24/08/2023  
Fecha de entrega: 5/10/2023

# Bash (Bourne – Again Shell)

El Bash es el shell, o también llamado “command language interpreter”, del sistema operativo Unix. El nombre es el acrónimo de ‘Bourne-Again SHell’, en referencia a al creador del shell anterior al bash, llamado “sh”, Stephen Bourne.

Actualmente Bash es el shell por defecto en casi todas las versiones de Linux y, actualmente, existen también versiones para Windows activando "Windows Subsystem for Linux".

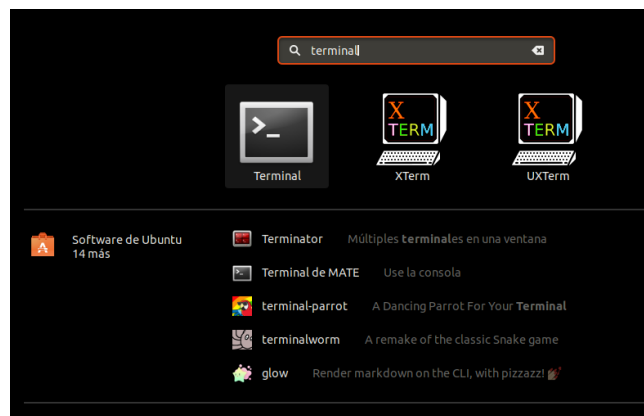
Además de Bash, los otros shells más utilizados actualmente son:

- **csch**: The C shell for programmers who like the syntax of the C language
- **ksh**: The Korn shell, written by David Korn and popular with Unix users
- **tcsh**: A version of csh with more ease-of-use features
- **zsh**: The Z shell, which combines many features of other popular shells

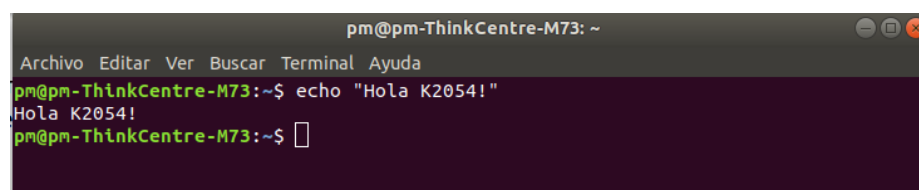
Se dice “shell” ya que es la coraza del kernel, permitiéndonos interactuar con el sistema operativo a través de él. Básicamente es un ejecutor de comandos. Sin embargo, los shell de Unix/Linux podrían verse también como lenguajes de programación, que, como hemos visto en clases anteriores, son lenguajes formales.

Como cualquier lenguaje de programación de alto nivel, shell tiene variables, sentencias algorítmicas y funciones.

Como intérprete de comandos, bash se puede utilizar invocando un comando por vez. Estos comandos pertenecen al conjunto de utilidades GNU. Para ello, por ejemplo en Ubuntu Linux, solamente se debe abrir un terminal:



Y luego ejecutar el comando deseado. En este caso se un “echo” en pantalla, asignación externa de salida como el “cout” de c++ visto en AyED:



Sin embargo, como lenguaje de programación, permite combinar los resultados obtenidos por distintas ejecuciones.

Los shells pueden utilizarse interactivamente tipeando el input de cada comando o en modo no interactivo leyendo los comandos desde un archivo.

Además se pueden ejecutar comandos GNU tanto sincrónicamente como asincrónicamente. La ejecución de comandos sincrónica espera el resultado de cada comando para continuar recibiendo datos de entrada; la ejecución asincrónica puede ejecutar en paralelo varios comandos con el shell. Puede ver un listado de los comandos GNU en:

[https://en.wikipedia.org/wiki/List\\_of\\_GNU\\_Core\\_Uutilities\\_commands](https://en.wikipedia.org/wiki/List_of_GNU_Core_Uutilities_commands)

## grep y sed

Para el procesamiento de textos en Linux/Unix/Mac OS, las tres herramientas prácticamente built in son grep, sed y awk. Aunque son herramientas diferentes, presentan similitudes en algunos escenarios. Por ejemplo, para buscar patrones en archivos de texto e imprimir las coincidencias por salida estándar.

### grep

El comando grep busca líneas que concuerden con una expresión regular e imprime las líneas coincidentes en la salida estándar. En general, las invocaciones de grep pueden verse de la siguiente manera:

```
grep [-E] 'REGEX' ARCHIVO  
COMANDO | grep [-E] 'REGEX'
```

La opción -o permite que, en vez de mostrar todas las líneas que hacen sobre las que la expresión regular hace match, muestran únicamente el match.

La opción -i (ignore case) ignora las mayúsculas y minúsculas, haciendo match sin importar el caso.

### sed

Otra forma de usar expresiones regulares es mediante el comando sed. Este es más adecuado para reemplazar o borrar texto, pero también puede usarse para hacer búsquedas. La sintaxis para ello sería así: sed -n[r] '/REGEX/p' ARCHIVO.

Sed puede buscar, reemplazar y borrar partes que coincidan con un patrón de búsqueda dentro de un archivo. Por ejemplo:

```
$sed 's/palabra1/palabra2/' un_Archivo.txt
```

El comando reemplaza todas las apariciones de palabra1 por palabra2 en un\_Archivo.txt. La letra 's' al comienzo indica que se aplicará una sustitución.

El comando sed también usa expresiones regulares básicas por defecto, se pueden usar expresiones regulares extendidas con la opción -r.

Para borrar un patrón de un archivo se indica d al final:

```
$ sed '/REGEX/d' un_Archivo.txt
```

La sintaxis de sed no es tan intuitiva como la de grep. Eso radica en la complejidad de sus operaciones ya que, si bien las expresiones regulares siguen siendo válidas, se debe expresar también las operaciones a realizar sobre el archivo, que pueden ser mostrar, sustituir, añadir, o eliminar (a diferencia de grep con el que se puede sólo mostrar).

Para profundiar el grep se recomienda:

<https://likegeeks.com/es/sed-de-linux/>

## Símbolos para expresiones regulares y expresiones regulares extendidas (Por orden de precedencia):

`c` cualquier carácter no especial `c` concuerda a sí mismo

`\c` cancela cualquier significado especial del carácter `c`

`^` inicio de línea

`$` final de línea

`\b` marca inicio o final de palabra

`.` cualquier carácter individual

`[...]` cualquiera de los caracteres en ...; los rangos tipo a-z son legales

`[^...]` cualquier carácter individual que no se encuentre en ...; los rangos valen `\n` lo que concordó con el n-ésimo `\(...\)` (grep solamente)

`r*` cero o más ocurrencias de `r`

`r+` una o más ocurrencias de `r` (egrep solamente)

`r?` cero o una ocurrencia de `r` (egrep solamente)

`r{n}` `n` ocurrencias de `r` (egrep solamente)

`r{n,}` `n` o más ocurrencias de `r` (egrep solamente)

`r{,m}` de 0 a `m` ocurrencias de `r` (egrep solamente)

`r{n,m}` de `n` a `m` ocurrencias de `r` (egrep solamente)

`r1r2` `r1` seguida de `r2`

`r1|r2` `r1` o `r2` (egrep solamente)

`\(r\)` expresión regular marcada `r` (grep solamente); puede ser una expresión

# Ejemplos

Dado el archivo llamado EJEMPLO:

```
109660 AEROSOL PENETRIT 316 10 FUNC 163 GR: $ 384,04
109670 AEROSOL PENETRIT 318 10 FUNC 290 GR: $ 586,16
ACEITES LUBRICANTES MULTIUSO PENETRIT
109640 ACEITE PENETRIT 113 100 CM3: $ 159,86
ACEITES TRADICIONAL PENETRIT
048370 ACEITE PENETRIT 101 - 100 CM3: $ 220,50
048390 ACEITE PENETRIT 106 - 500 CM3: $ 898,54
048400 ACEITE PENETRIT 103 - 1000 CM3: $ 1798,91
048470 AEROSOL PENETRIT 302 - 163 GR: $ 512,66
048480 AEROSOL PENETRIT 303 - 290 GR: $ 859,95
ACOPLES COMPRESION PROFESIONAL DUKE
036450 ACOPLE DUKE COMPRESION PROF 1/2: $ 259,36
036460 ACOPLE DUKE COMPRESION PROF 3/4: $ 320,62
PALAS ACERO CABO CORTO HIERRO
050820 PALA ACERO ANCHA C/CAÑO: $ 2571,82
050830 PALA ACERO CORAZON C/CAÑO: $ 2571,82
050840 PALA ACERO POCERA C/CAÑO: $ 2571,82
050850 PALA ACERO PUNTA C/CAÑO: $ 2571,82
PALAS CARBONERAS GHERARDI
033019 PALA GHERARDI ALUMINIO CARBONERA: $ 9007,07
033020 PALA GHERARDI ESTAMPADA CARBONERA: $ 4279,69
PALAS CHAPA ESTAMPADA C/CORTO GHERARDI
152290 PALA GHERARDI ESTAMP ANCHA: $ 3930,77
152310 PALA GHERARDI ESTAMP CORAZON: $ 3930,77
152330 PALA GHERARDI ESTAMP POCERA: $ 3930,77
PALAS CHAPA ESTAMPADA C/LARGO GHERARDI
152300 PALA GHERARDI ESTAMP ANCHA C/L: $ 3930,77
152320 PALA GHERARDI ESTAMP CORAZON C/L: $ 3930,77
152340 PALA GHERARDI ESTAMP POCERA C/L: $ 3930,77
PALAS ESTAMPADAS CABO CORTO MAGER
050520 PALA MAGER ANCHA C/CORTO: $ 1434,58
050530 PALA MAGER PUNTA C/CORTO: $ 1434,58
050540 PALA MAGER POCERA C/CORTO: $ 1434,58
050550 PALA MAGER CORAZON C/CORTO: $ 1434,58
050560 PALA MAGER CARBONERA ALUMINIO C/CORTO: $ 5004,21
050800 PALA MAGER CARBONERA PLASTICA C/CORTO: $ 3402,74
PALAS ESTAMPADAS CABO LARGO MAGER
022340 PALA MAGER ANCHA C/LARGO: $ 1863,28
022570 PALA MAGER PUNTA C/LARGO: $ 1863,28
022580 PALA MAGER POCERA C/LARGO: $ 1863,28
035670 PALA MAGER CORAZON C/LARGO: $ 1863,28
050810 PALA MAGER POCERA C/HIERRO MACIZO: $ 4466,05
PALAS FORJADA CABO CORTO GHERARDI
032900 PALA GHERARDI FORJ ANCHA: $ 6106,69
032910 PALA GHERARDI FORJ CORAZON: $ 5654,16
032940 PALA GHERARDI FORJ PUNTA: $ 6106,69
032970 PALA GHERARDI FORJ POCERA: $ 6106,69
032980 PALA GHERARDI FORJ POCERA ESCOCESA: $ 6883,27
PALAS FORJADA CABO HIERRO GHERARDI
031090 PALA GHERARDI POCERA C/HIERRO: $ 7756,39
032890 PALA GHERARDI POCERA ESCOCESA C/HIE: $ 8400,47
PALAS FORJADA CABO LARGO GHERARDI
032920 PALA GHER FORJ ANCHA C/L: $ 6106,69
033000 PALA GHER FORJ CORAZON C/L: $ 5654,16
033010 PALA GHER FORJ PUNTA C/L: $ 6106,69
```

Si se quisiera filtrar el archivo para mostrar únicamente las línea que contengan la expresión regular PALA se puede ejecutar:

grep 'PALA' EJEMPLO

```
pm@pm-ThinkCentre-M73:~$ cd 'Escritorio/TP 1'
pm@pm-ThinkCentre-M73:~/Escritorio/TP 1$ ls
EJEMPLO
pm@pm-ThinkCentre-M73:~/Escritorio/TP 1$ grep 'PALA' EJEMPLO
PALAS ACERO CABO CORTO HIERRO
050820 PALA ACERO ANCHA C/CAÑO: $ 2571,82
050830 PALA ACERO CORAZON C/CAÑO: $ 2571,82
050840 PALA ACERO POCERA C/CAÑO: $ 2571,82
050850 PALA ACERO PUNTA C/CAÑO: $ 2571,82
PALAS CARBONERAS GHERARDI
033019 PALA GHERARDI ALUMINIO CARBONERA: $ 9007,07
033020 PALA GHERARDI ESTAMPADA CARBONERA: $ 4279,69
PALAS CHAPA ESTAMPADA C/CORTO GHERARDI
152290 PALA GHERARDI ESTAMP ANCHA: $ 3930,77
152310 PALA GHERARDI ESTAMP CORAZON: $ 3930,77
152330 PALA GHERARDI ESTAMP POCERA: $ 3930,77
PALAS CHAPA ESTAMPADA C/LARGO GHERARDI
152300 PALA GHERARDI ESTAMP ANCHA C/L: $ 3930,77
152320 PALA GHERARDI ESTAMP CORAZON C/L: $ 3930,77
152340 PALA GHERARDI ESTAMP POCERA C/L: $ 3930,77
PALAS ESTAMPADAS CABO CORTO MAGER
050520 PALA MAGER ANCHA C/CORTO: $ 1434,58
050530 PALA MAGER PUNTA C/CORTO: $ 1434,58
050540 PALA MAGER POCERA C/CORTO: $ 1434,58
050550 PALA MAGER CORAZON C/CORTO: $ 1434,58
050560 PALA MAGER CARBONERA ALUMINIO C/CORTO: $ 5004,21
050800 PALA MAGER CARBONERA PLASTICA C/CORTO: $ 3402,74
PALAS ESTAMPADAS CABO LARGO MAGER
022340 PALA MAGER ANCHA C/LARGO: $ 1863,28
022570 PALA MAGER PUNTA C/LARGO: $ 1863,28
022580 PALA MAGER POCERA C/LARGO: $ 1863,28
035670 PALA MAGER CORAZON C/LARGO: $ 1863,28
050810 PALA MAGER POCERA C/HIERRO MACIZO: $ 4466,05
PALAS FORJADA CABO CORTO GHERARDI
032900 PALA GHERARDI FORJ ANCHA: $ 6106,69
032910 PALA GHERARDI FORJ CORAZON: $ 5654,16
032940 PALA GHERARDI FORJ PUNTA: $ 6106,69
032970 PALA GHERARDI FORJ POCERA: $ 6106,69
032980 PALA GHERARDI FORJ POCERA ESCOCESA: $ 6883,27
PALAS FORJADA CABO HIERRO GHERARDI
031090 PALA GHERARDI POCERA C/HIERRO: $ 7756,39
032890 PALA GHERARDI POCERA ESCOCESA C/HIE: $ 8400,47
PALAS FORJADA CABO LARGO GHERARDI
032920 PALA GHER FORJ ANCHA C/L: $ 6106,69
033000 PALA GHER FORJ CORAZON C/L: $ 5654,16
033010 PALA GHER FORJ PUNTA C/L: $ 6106,69
```

Sin embargo, si se ejecuta grep con la expresión regular PALA.\*POCERA:

```
pm@pm-ThinkCentre-M73:~/Escritorio/TP 1$ grep 'PALA.*POCERA' EJEMPLO
050840 PALA ACERO POCERA C/CAÑO: $ 2571,82
152330 PALA GHERARDI ESTAMP POCERA: $ 3930,77
152340 PALA GHERARDI ESTAMP POCERA C/L: $ 3930,77
050540 PALA MAGER POCERA C/CORTO: $ 1434,58
022580 PALA MAGER POCERA C/LARGO: $ 1863,28
050810 PALA MAGER POCERA C/HIERRO MACIZO: $ 4466,05
032970 PALA GHERARDI FORJ POCERA: $ 6106,69
032980 PALA GHERARDI FORJ POCERA ESCOCESA: $ 6883,27
031090 PALA GHERARDI POCERA C/HIERRO: $ 7756,39
032890 PALA GHERARDI POCERA ESCOCESA C/HIE: $ 8400,47
```

Ahora bien, si se deseara listar las palas poceras o de punta una expresión regular

extendida podría ser 'PALA.\*POCERA|PUNTA'. Notar que en el primer intento el resultado es nulo, ya que no se aclara que la expresión regular es extendida. Los siguientes intentos son equivalentes:

```
pm@pm-ThinkCentre-M73:~/Escritorio/TP 1$ grep 'PALA.*POCERA|PUNTA' EJEMPLO
pm@pm-ThinkCentre-M73:~/Escritorio/TP 1$ grep -E 'PALA.*POCERA|PUNTA' EJEMPLO
050840  PALA ACERO POCERA C/CAÑO: $ 2571,82
050850  PALA ACERO PUNTA C/CAÑO: $ 2571,82
152330  PALA GHERARDI ESTAMP POCERA: $ 3930,77
152340  PALA GHERARDI ESTAMP POCERA C/L: $ 3930,77
050530  PALA MAGER PUNTA C/CORTO: $ 1434,58
050540  PALA MAGER POCERA C/CORTO: $ 1434,58
022570  PALA MAGER PUNTA C/LARGO: $ 1863,28
022580  PALA MAGER POCERA C/LARGO: $ 1863,28
050810  PALA MAGER POCERA C/HIERRO MACIZO: $ 4466,05
032940  PALA GHERARDI FORJ PUNTA: $ 6106,69
032970  PALA GHERARDI FORJ POCERA:$ 6106,69
032980  PALA GHERARDI FORJ POCERA ESCOCESA:$ 6883,27
031090  PALA GHERARDI POCERA C/HIERRO: $ 7756,39
032890  PALA GHERARDI POCERA ESCOCESA C/HIE: $ 8400,47
033010  PALA GHER FORJ PUNTA C/L: $ 6106,69
pm@pm-ThinkCentre-M73:~/Escritorio/TP 1$ egrep 'PALA.*POCERA|PUNTA' EJEMPLO
050840  PALA ACERO POCERA C/CAÑO: $ 2571,82
050850  PALA ACERO PUNTA C/CAÑO: $ 2571,82
152330  PALA GHERARDI ESTAMP POCERA: $ 3930,77
152340  PALA GHERARDI ESTAMP POCERA C/L: $ 3930,77
050530  PALA MAGER PUNTA C/CORTO: $ 1434,58
050540  PALA MAGER POCERA C/CORTO: $ 1434,58
022570  PALA MAGER PUNTA C/LARGO: $ 1863,28
022580  PALA MAGER POCERA C/LARGO: $ 1863,28
050810  PALA MAGER POCERA C/HIERRO MACIZO: $ 4466,05
032940  PALA GHERARDI FORJ PUNTA: $ 6106,69
032970  PALA GHERARDI FORJ POCERA:$ 6106,69
032980  PALA GHERARDI FORJ POCERA ESCOCESA:$ 6883,27
031090  PALA GHERARDI POCERA C/HIERRO: $ 7756,39
032890  PALA GHERARDI POCERA ESCOCESA C/HIE: $ 8400,47
033010  PALA GHER FORJ PUNTA C/L: $ 6106,69
```

## Permisos de archivos en linux (Comando chmod)

Los permisos de archivos y directorios están dados por tres dígitos XYZ, siendo X el propietario, Y el propietario del grupo y Z el resto.

Considerando que los permisos para cada acción tienen asociado un valor numérico:

- **r** (lectura) = 4
- **w** (escritura) = 2
- **x** (ejecución) = 1

La ejecución de:

`chmod 746 un_archivo`

En este ejemplo se le está dando permisos completos al propietario (4+2+1), lectura al propietario del grupo y lectura y escritura al resto (4+2).

Otro ejemplo sería este: `chmod 777 file2.txt`, este comando básicamente otorgaría todos los

permisos para cada tipo de usuario (propietario, grupo y otros) sobre el archivo file2.txt.

Valor	Valor Numérico	Descripción
<b>-rw-----</b>	<b>600</b>	Sólo el propietario puede leer y escribir
<b>-rw-r--r--</b>	<b>644</b>	El propietario puede leer y escribir, el grupo y otros pueden leer.
<b>-rw-rw-rw-</b>	<b>666</b>	El propietario, el grupo y otros pueden leer y escribir.
<b>-rwx-----</b>	<b>700</b>	El propietario puede leer, escribir y ejecutar, el grupo y otros no pueden hacer nada con el archivo.
<b>-rwx--x--x</b>	<b>711</b>	El propietario puede leer, escribir y ejecutar, el grupo y otros pueden ejecutar.
<b>-rwxr-xr-x</b>	<b>755</b>	El propietario puede leer, escribir y ejecutar, el grupo y otros pueden leer y ejecutar.
<b>-rwxrwxrwx</b>	<b>777</b>	EL propietario, el grupo y otros pueden leer, escribir y ejecutar.

## Scripting con comandos bash y redireccionamiento de la salida estándar:

En el ejemplo, vemos que hay palas marca "Gherardi" que tienen el nombre abreviado "Gher". Si queremos otro archivo con palas solamente y con el nombre de la marca completa podríamos hacer:

```
grep 'PALA' EJEMPLO > EJEMPLO2  
sed 's/GHER\b/GHERARDI/' EJEMPLO2 > OUT.TXT
```

Esta secuencia de comandos grep genera un archivo intermedio al que se le aplica luego un sed. La salida de grep se redirecciona desde la salida estándar a el archivo EJEMPLO2. Luego sed toma este último archivo, lo procesa y redirecciona la salida estándar al archivo OUT.TXT. Podemos armar un script, en este caso un archivo al que llamaremos Comandos.sh, y lo ejecutamos posicionándonos en carpeta e invocando ./Comandos.sh:

```
pm@pm-ThinkCentre-M73:~$ cd Escritorio/'TP SINTAXIS BASH'  
pm@pm-ThinkCentre-M73:~/Escritorio/TP SINTAXIS BASH$ ./Comandos.sh  
bash: ./Comandos.sh: Permiso denegado
```



Vemos que tenemos permisos denegados. Para ello podemos realizar:

```
pm@pm-ThinkCentre-M73:~/Escritorio/TP SINTAXIS BASH$ chmod 777 Comandos.sh
pm@pm-ThinkCentre-M73:~/Escritorio/TP SINTAXIS BASH$ ./Comandos.sh
pm@pm-ThinkCentre-M73:~/Escritorio/TP SINTAXIS BASH$
```

Esta vez el script se ejecutó exitosamente, generando dos archivos, EJEMPLO2 y OUT.TXT, teniendo este último, el resultado final. Es importante aclarar que asignar permisos 777 no es para nada seguro y se utiliza en este simple ejemplo didáctico.

## Pipelines

Una forma de simplificar el script anterior y evitar la generación de un archivo intermedio es mediante la utilización de pipelines. Se puede obtener el mismo resultado ejecutando simplemente:

```
pm@pm-ThinkCentre-M73:~/Escritorio/TP SINTAXIS BASH$ grep 'PALA' EJEMPLO | sed 's/GHER\b/GHERARDI/' > Out2
pm@pm-ThinkCentre-M73:~/Escritorio/TP SINTAXIS BASH$
```

Esto hace que simplemente la salida de grep se direcciona como input al siguiente comando, en este caso sed.

## Consigna

1. Crear un usuario en GitHub <https://github.com/> con el correo institucional frba. Crear un repositorio. Complete la planilla: [https://docs.google.com/spreadsheets/d/1JSY2o7uRNJzwAJmo6yR0j2\\_swm7E\\_xfOshGX-UjoTsU/edit?usp=sharing](https://docs.google.com/spreadsheets/d/1JSY2o7uRNJzwAJmo6yR0j2_swm7E_xfOshGX-UjoTsU/edit?usp=sharing) con los datos del grupo. Dentro del repositorio deberá subir todos los archivos que compongan la entrega de este trabajo dentro de una carpeta llamada "TP 1". Para desarrollar este punto, ver el apéndice de este documento.
2. Debe entregar un único script que resuelva los siguientes puntos:
  - a. Reemplace cada punto del archivo "breve\_historia.txt" por punto y salto de línea generando un nuevo archivo.
  - b. Borre todas las líneas en blanco.
  - c. Cree un nuevo archivo: "breve\_historia\_2.txt" con el resultado de las operaciones a y b (redireccionamiento de la salida estándar).
  - d. Del archivo "breve\_historia.txt", liste todas las oraciones que contengan la palabra "independencia" sin distinguir mayúsculas y minúsculas.
  - e. Muestre las líneas que empiecen con "El" y terminen con "." del archivo "breve\_historia.txt".
  - f. Sobre el mismo archivo del punto anterior, indique en cuántas oraciones aparece la palabra "peronismo". Puede usar la opción -c para contar.
  - g. Muestre la cantidad de oraciones que contienen la palabra "Sarmiento" y la palabra "Rosas".
  - h. Muestre las oraciones que tengan fechas referidas al siglo XIX.
  - i. Borre la primera palabra de cada línea. Utilice sustitución con sed. La sintaxis para sustituir la primera palabra de cada línea por "nada" sería:  
`$sed "s/^[a-zA-Z]*\b//g" nombre_archivo`  
(La "s" indica sustitución; entre los dos primeros /.../ está la expresión regular que queremos reemplazar, en este caso "/^[a-zA-Z]\*\b"; entre el segundo y el tercer "/" se indica la expresión por la cual será reemplazada, en este caso por la palabra vacía. Finalmente la "g" indica que el cambio será en todo el archivo.
  - j. Escriba un comando que enumere todos los archivos de una carpeta que contengan extensión ".txt". (tip: pipe con el comando ls).
3. Investigue y explique, dando ejemplos cómo se utilizan los siguientes elementos en bash:
  - Variables.
  - Sentencias condicionales.
  - Sentencias cíclicas.
  - Subprogramas

Dé ejemplos de cada una.

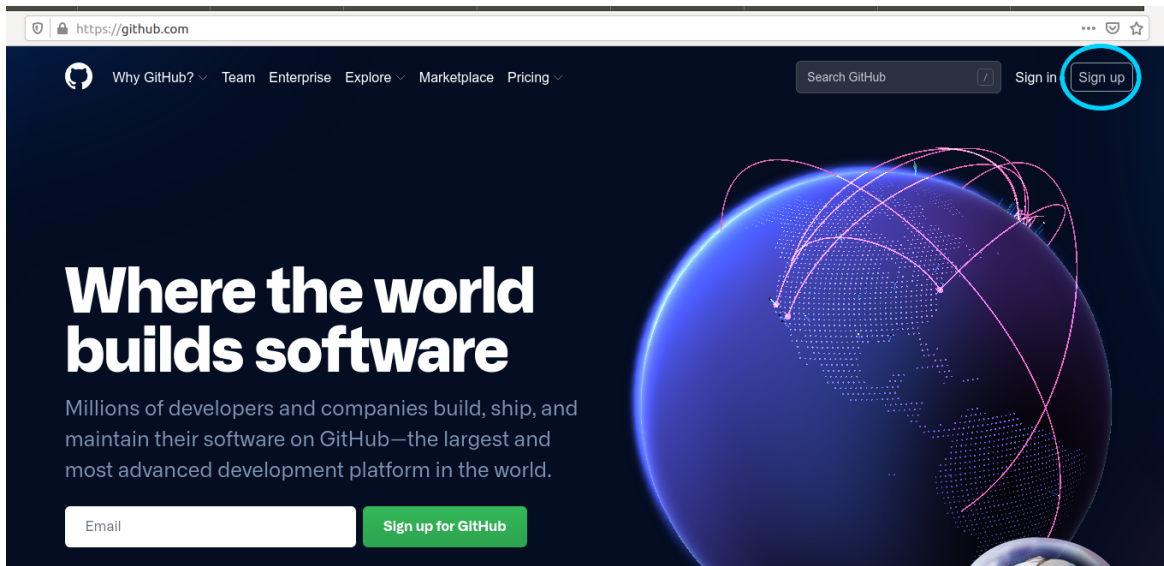
## Condiciones de entrega

La entrega del trabajo se debe realizar el 5/10 por GitHub institucional en una carpeta llamada "TP 1" dentro del repositorio del grupo. La carpeta debe contener:

- Informe en PDF con carátula indicando: legajo, nombre, apellido, correo institucional, usuario gitHub de cada integrante y link al repositorio.
- Casos de prueba (ejecuciones, pueden ser capturas de pantalla de la salida de consola luego de la ejecución.
- Soluciones en uno o más scripts sh.

## Apéndice: Creación de cuenta en GitHub

- Acceda a github.com y haga click en Sign Up:



- Ingrese el nombre de usuario deseado y la cuenta frba. También debe ingresar una clave, finalmente haga click en el botón “Verify” o “Verificar”:

Join GitHub

### Create your account

Username \*

IngPabloMendez ✓

Email address \*

pmendez@frba.utn.edu.ar ✓

Password \*

\*\*\*\*\* ✓

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)

Email preferences

☐ Send me occasional product updates, announcements, and offers.

Verify your account

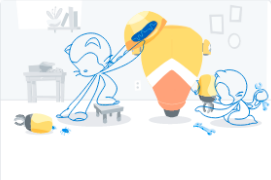
Solucione este rompecabezas para que sepamos que es una persona real

Verificar

- Seleccione la opción “Create a repository”:


## What do you want to do first?

Every developer needs to configure their environment, so let's get your GitHub experience optimized for you.




**Start a new project**  
Start a new repository or bring over an existing repository to keep contributing to it.

Create a repository



**Collaborate with your team**  
Improve the way your team works together and get access to more features with an organization.

Create an organization



**Learn how to use GitHub**  
Get started with an "Introduction to GitHub" course in our Learning Lab.

Start Learning

[Skip this for now >](#)

- Ingresar el nombre del repositorio, indicar que es privado, tildar “Add a Readme file” y “Choose a license”. Seleccionar licencia GNU en las opciones de Licenses.

### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner \*

Repository name \*

IngPabloMendez ▾

/ RepositorioPrivado1 ✓

Great repository names are short and memorable. Need inspiration? How about [literate-giggle?](#)

Description (optional)

Una descripción

☒ Public

Anyone on the internet can see this repository. You choose who can commit.

☐ Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☒ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

☐ Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

☒ Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

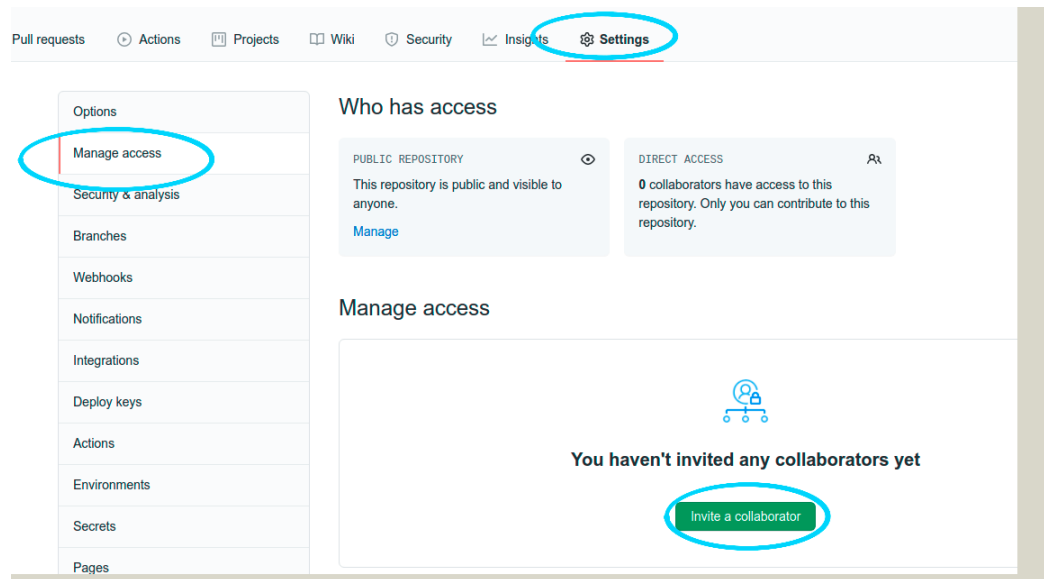
License: GNU General Public ... ▾

This will set `main` as the default branch. Change the default name in your [settings](#).

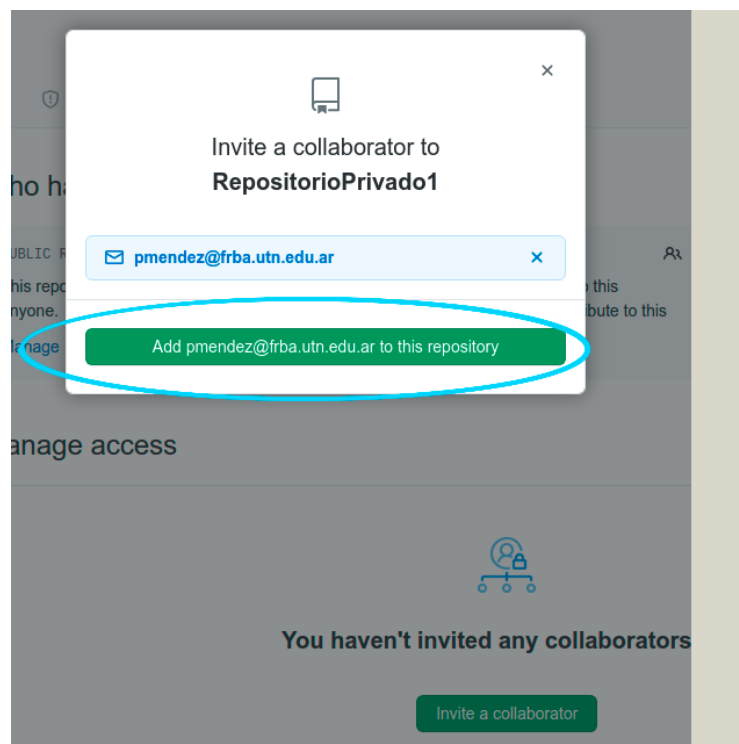
Create repository

- Compartir el repositorio con el profesor yendo a “Settings” -> “Manage

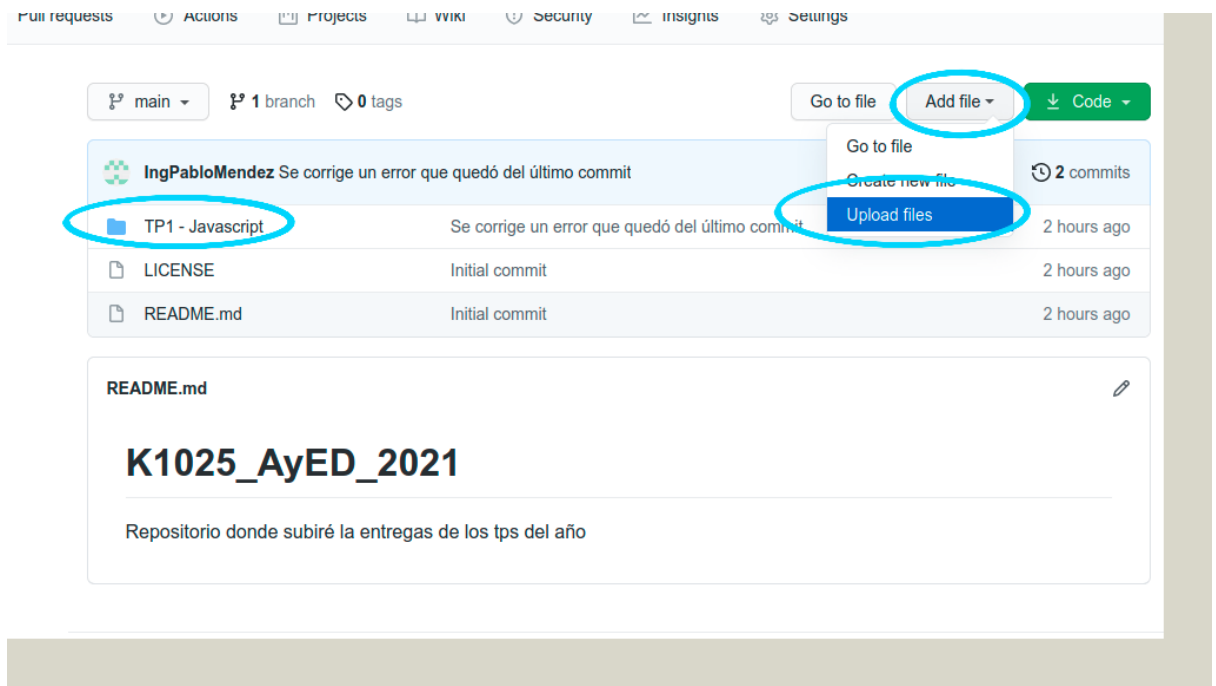
Access” y luego “Invite a Collaborator”



Finalmente buscar el usuario del profesor por mail: [pmendez@frba.utn.edu.ar](mailto:pmendez@frba.utn.edu.ar) y agregarlo:



- Finalmente agregar archivos. En este ejemplo se ilustra la subida de una carpeta completa arrastrándola y soltándola en el navegador. Se recomienda tener una carpeta local llamada TP1 y subirla arrastrándola completa al repositorio (teniendo cuidado de borrar previamente las subcarpetas bin y obj de los proyectos de codeblocks como se indica en las condiciones de entrega).



## Otras recomendaciones

Para realizar este trabajo se recomienda utilizar una versión live de Linux, puede servir cualquier distribución, una de las más comunes es Ubuntu. Se pueden crear los archivos de entrada y salida de los comandos en una unidad externa aparte de la que contenga el Sistema operativo en ejecución.

Otra solución es instalar linux en un ambiente virtualizado utilizando productos como Virtual Box o VM Ware.