

**UNIVERSIDAD LUTERANA SALVADOREÑA  
FACULTAD DE CIENCIAS DEL HOMBRE Y LA NATURALEZA  
LICENCIATURA EN CIENCIAS DE LA COMPUTACIÓN  
CÁTEDRA DE INTRODUCCION AL SOFTWARE LIBRE**



**TEMA:** PARCIAL FINAL.

**NOMBRE DE ESTUDIANTE:** Edwin Enmanuel Rivas Ramírez rr01136339

**DOCENTE:** Lic. Eduardo Chachagua

Tutor. Carlos Alfredo Ramos Águila

**CICLO/AÑO:** I/2020

**FECHA:** 12/06/20

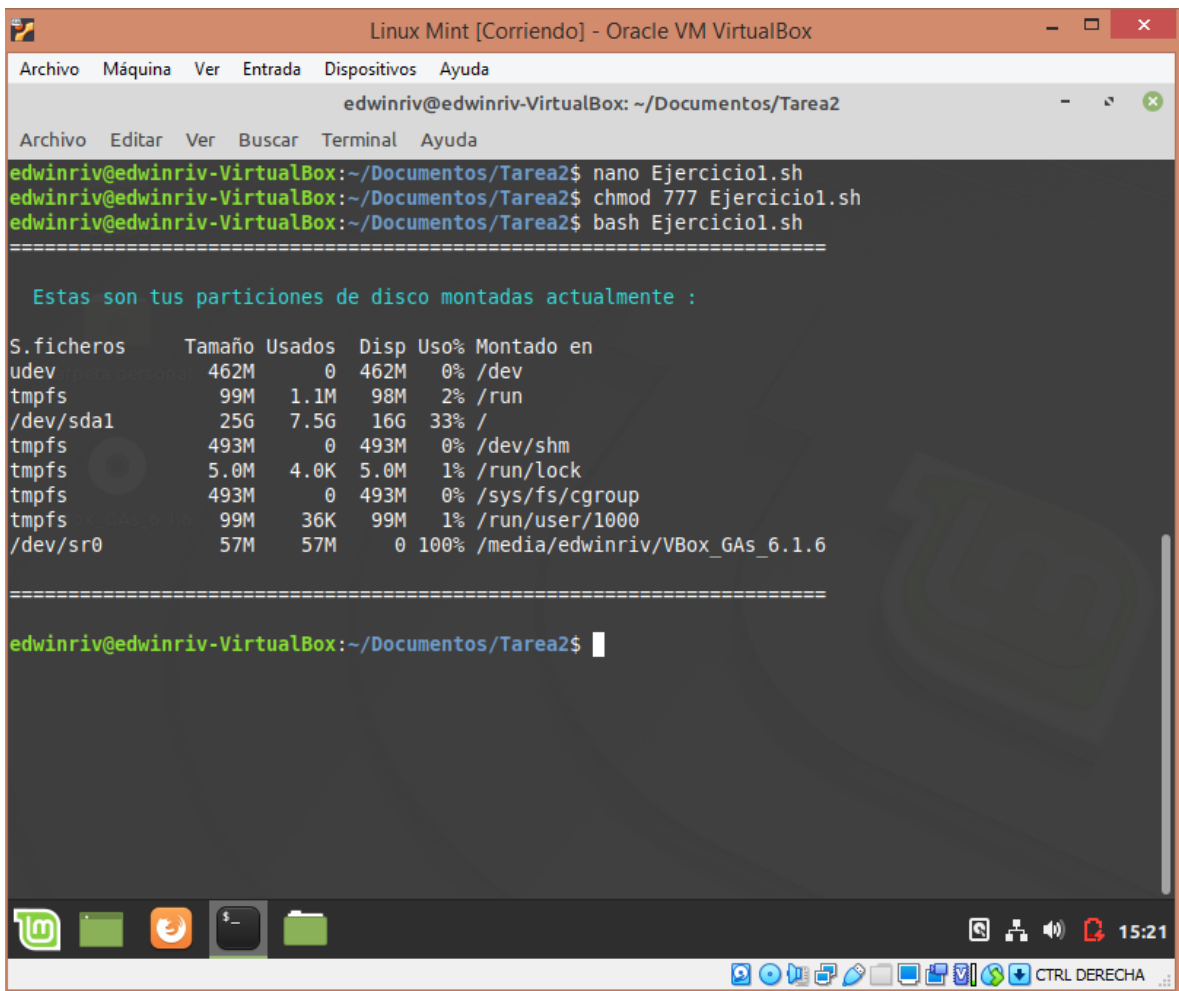
## Actividad-Práctica bash

1) Crear un script bash que busque las particiones del disco que están montadas actualmente en el sistema (las actuales) y el sistema de archivos que usan estas particiones y las muestre en formato centrado y a color

### Solución:

Comenzaríamos usando el comando nano y al final del nombre iría el formato “.sh”, cuando terminemos de digitar nuestro código le daremos permisos de ejecución con el comando “chmod” más los permisos “777” y por procederemos a ejecutar el Script con el comando “bash”.

Y como podemos ver, nos presenta las particiones que actualmente están instaladas o disponibles, con detalles de tamaño, uso, su ruta y etc.



```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2
Archivo Editar Ver Buscar Terminal Ayuda
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ nano Ejercicio1.sh
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ chmod 777 Ejercicio1.sh
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ bash Ejercicio1.sh
=====
Estas son tus particiones de disco montadas actualmente :
S.ficheros      Tamaño Usados  Disp Uso% Montado en
udev            462M      0 462M  0% /dev
tmpfs           99M      1.1M  98M   2% /run
/dev/sda1       25G      7.5G  16G  33% /
tmpfs           493M      0 493M  0% /dev/shm
tmpfs           5.0M      4.0K  5.0M  1% /run/lock
tmpfs           493M      0 493M  0% /sys/fs/cgroup
tmpfs           99M      36K   99M  1% /run/user/1000
/dev/sr0        57M      57M   0 100% /media/edwinriv/VBox_GAs_6.1.6
=====
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$
```

Y este es el código de creación, como siempre al inicio la ruta "bin/bash" para ejecutarlo de esa manera. Usamos el comando "printf" para centrar nuestro texto. El comando "sleep" para hacer pausas entre la ejecución de comandos. Las funciones, en mi caso la nombre "particiones", el comando que está dentro de la función "df -h" el cual nos ayuda para lo que necesita el Script y eso sería el código completo.

```
#!/bin/bash

#Ejercicio1

#inicio

echo "======"
printf "%50s\n \e[96m Estas son tus particiones de disco montadas actualmente : \e[0m  "
echo ""
echo ""
sleep 3

function particiones () {
    df -h
}

particiones

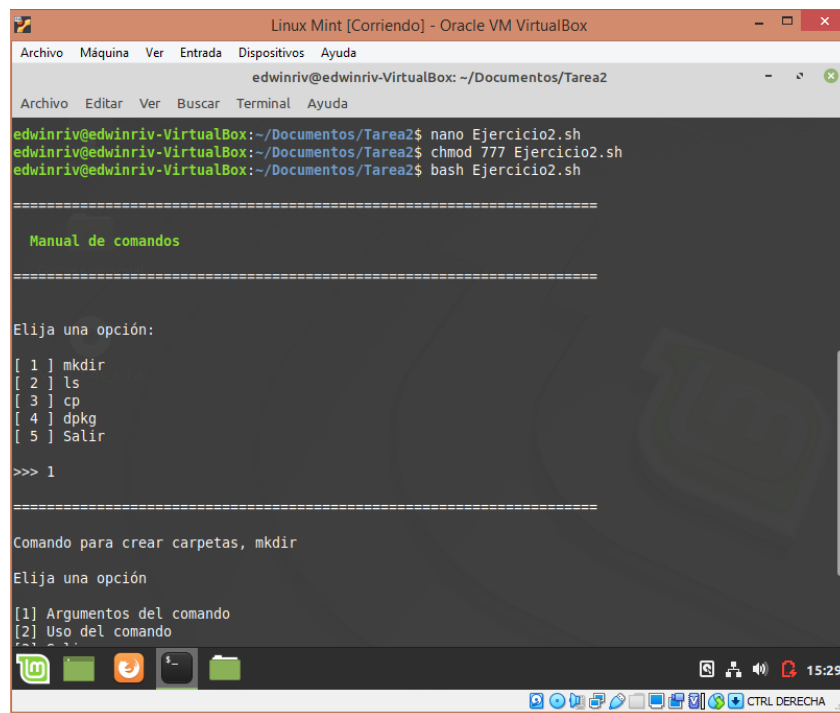
sleep 3

echo ""
echo "======"
echo ""
#fin
```

2) Crear un script bash el cual sirva como manual personalizado (con título: Manual de comandos ) de los comandos básicos en Linux (mkdir,ls,cp,dpkg) que permita elegir entre uno de éstos (en pantalla debe mostrarse los comandos disponible entre los que se puede elegir) y al seleccionar un comando que muestre dos opciones (1=argumentos del comando o 2=uso de los comandos) al elegir 1 debe ver todos los argumentos que tiene disponible este comando previamente elegido (también debe mostrarse todos los argumentos existentes para ese comando) y al seleccionar un argumento debe devolver información (en español) de cuál es el uso de ese argumento, en caso de ser 2 que muestre un título Uso del comando y se muestren 5 opciones entre las cuales al ingresar 1 me muestre la ejecución del comando (caso de uso) que se eligió en un principio con un argumento este debe ser ejecutado en el directorio /home/carlos/Descargas o /home/carlos/Downloads si se tiene en inglés el sistema, en caso ser 2 que ejecute el comando con un argumento diferente y así sucesivamente hasta 5 según la opción que se quiera (recordar que son casos de uso del comando, los que se ejecutarán). Use man mkdir, man ls, man cp, man dpkg.

### Solución:

Crearemos un archivo de texto con “nano” y en formato “.sh”, luego de su creación daremos los permisos con “chmod” y los permisos “777”, por último ejecutaremos el comando bash. Como vemos al inicio nos da el título del Script y nos abre un menú donde podremos seleccionar el comando deseado.



```
Linux Mint [Corriendo] - Oracle VM VirtualBox
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ nano Ejercicio2.sh
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ chmod 777 Ejercicio2.sh
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ bash Ejercicio2.sh

Manual de comandos

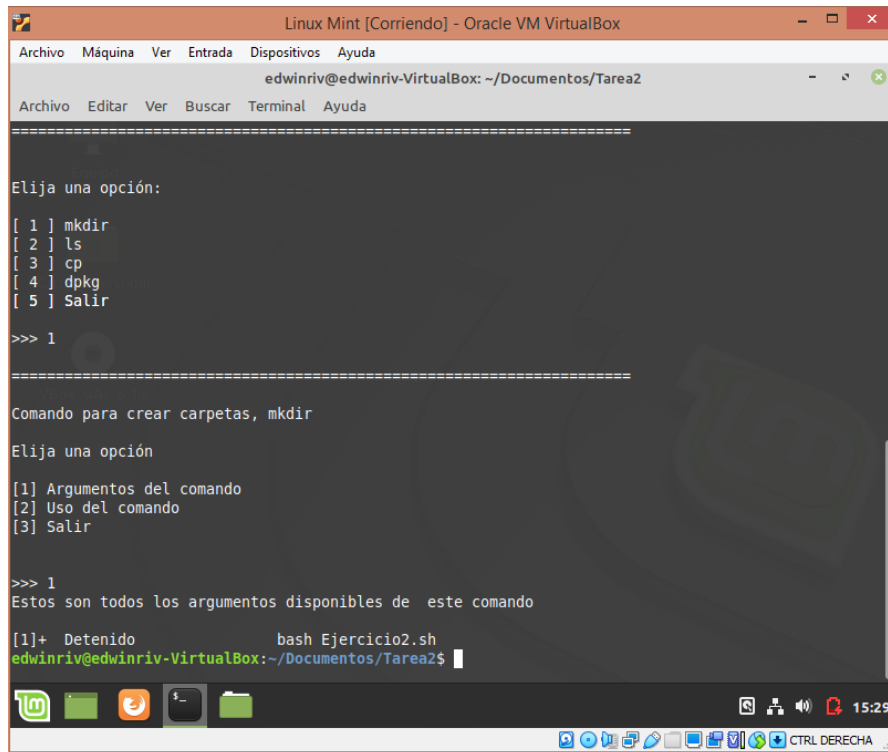
Elija una opción:
[ 1 ] mkdir
[ 2 ] ls
[ 3 ] cp
[ 4 ] dpkg
[ 5 ] Salir

>>> 1

Comando para crear carpetas, mkdir

Elija una opción
[1] Argumentos del comando
[2] Uso del comando
```

En mi caso elegí la opción 1 y me apareció un segundo menú donde nos aparecen 3 opciones, yo elegiré la opción 1.



```
Linux Mint [Corriendo] - Oracle VM VirtualBox
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2

=====
Elija una opción:
[ 1 ] mkdir
[ 2 ] ls
[ 3 ] cp
[ 4 ] dpkg
[ 5 ] Salir

>>> 1

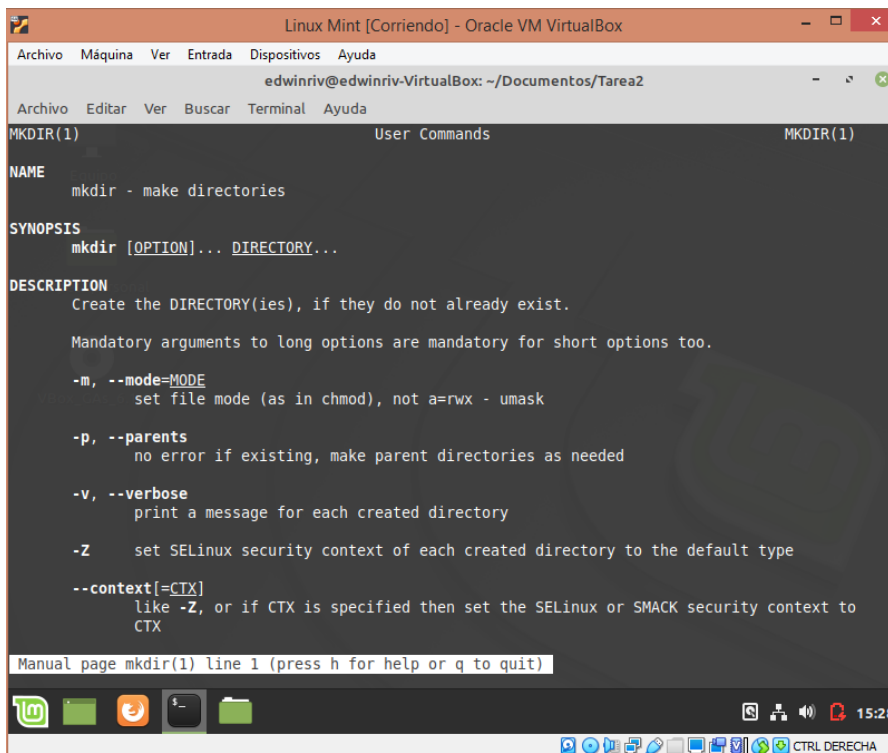
=====
Comando para crear carpetas, mkdir

Elija una opción
[1] Argumentos del comando
[2] Uso del comando
[3] Salir

>>> 1
Estos son todos los argumentos disponibles de este comando

[1]+ Detenido bash Ejercicio2.sh
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$
```

Como vemos nos apareció esta ventana la cual es donde están todos los argumentos de dicho comando.



```
Linux Mint [Corriendo] - Oracle VM VirtualBox
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2

MKDIR(1) User Commands MKDIR(1)

NAME
mkdir - make directories

SYNOPSIS
mkdir [OPTION]... DIRECTORY...

DESCRIPTION
Create the DIRECTORY(ies), if they do not already exist.

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

-m, --mode=MODE
    set file mode (as in chmod), not a=rwx - umask

-p, --parents
    no error if existing, make parent directories as needed

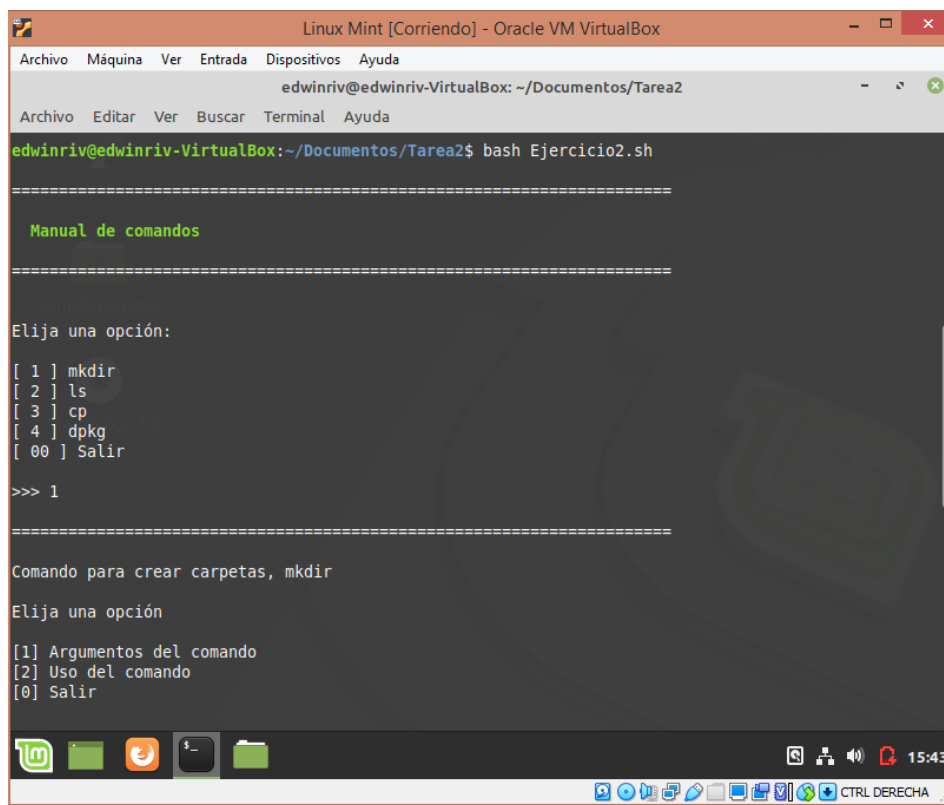
-v, --verbose
    print a message for each created directory

-Z
    set SELinux security context of each created directory to the default type

--context[=CTX]
    like -Z, or if CTX is specified then set the SELinux or SMACK security context to CTX

Manual page mkdir(1) line 1 (press h for help or q to quit)
```

Ahora seleccionare otra opción del mismo menú, la opción 1.



```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2
Archivo Editar Ver Buscar Terminal Ayuda
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ bash Ejercicio2.sh
=====
Manual de comandos
=====

Elija una opción:

[ 1 ] mkdir
[ 2 ] ls
[ 3 ] cp
[ 4 ] dpkg
[ 00 ] Salir

>>> 1

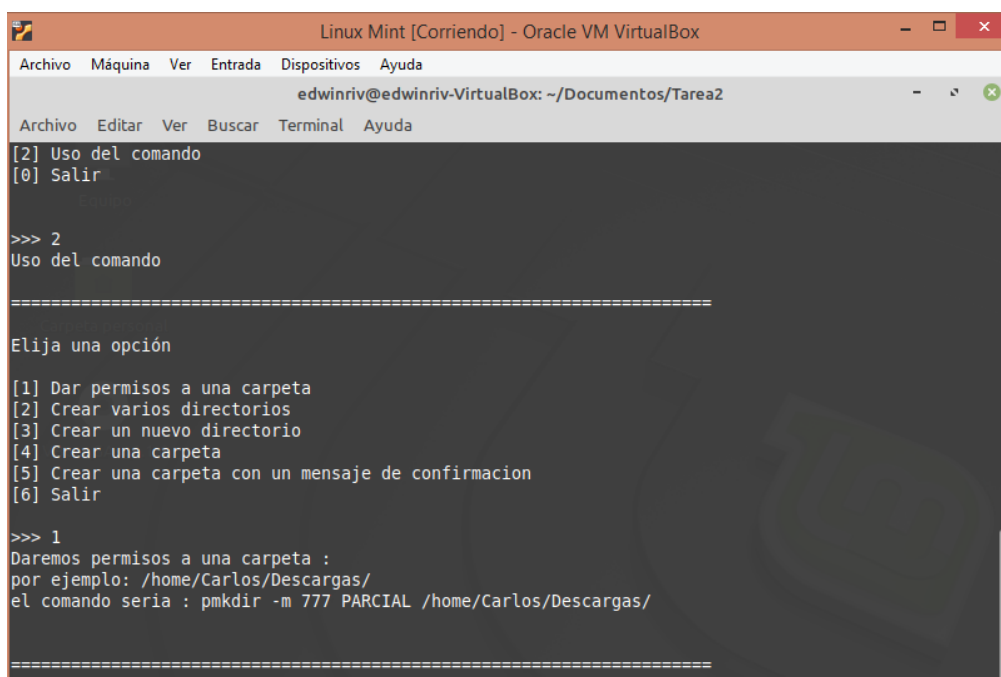
=====

Comando para crear carpetas, mkdir

Elija una opción

[1] Argumentos del comando
[2] Uso del comando
[0] Salir
```

Nos aparecerá el uso de comando como título y seis opciones para ejecutar. En mi caso seleccione la opción 1 y me dio un resumen de su uso y un ejemplo para poder usarlo.



```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2
Archivo Editar Ver Buscar Terminal Ayuda
[2] Uso del comando
[0] Salir

>>> 2
Uso del comando

=====

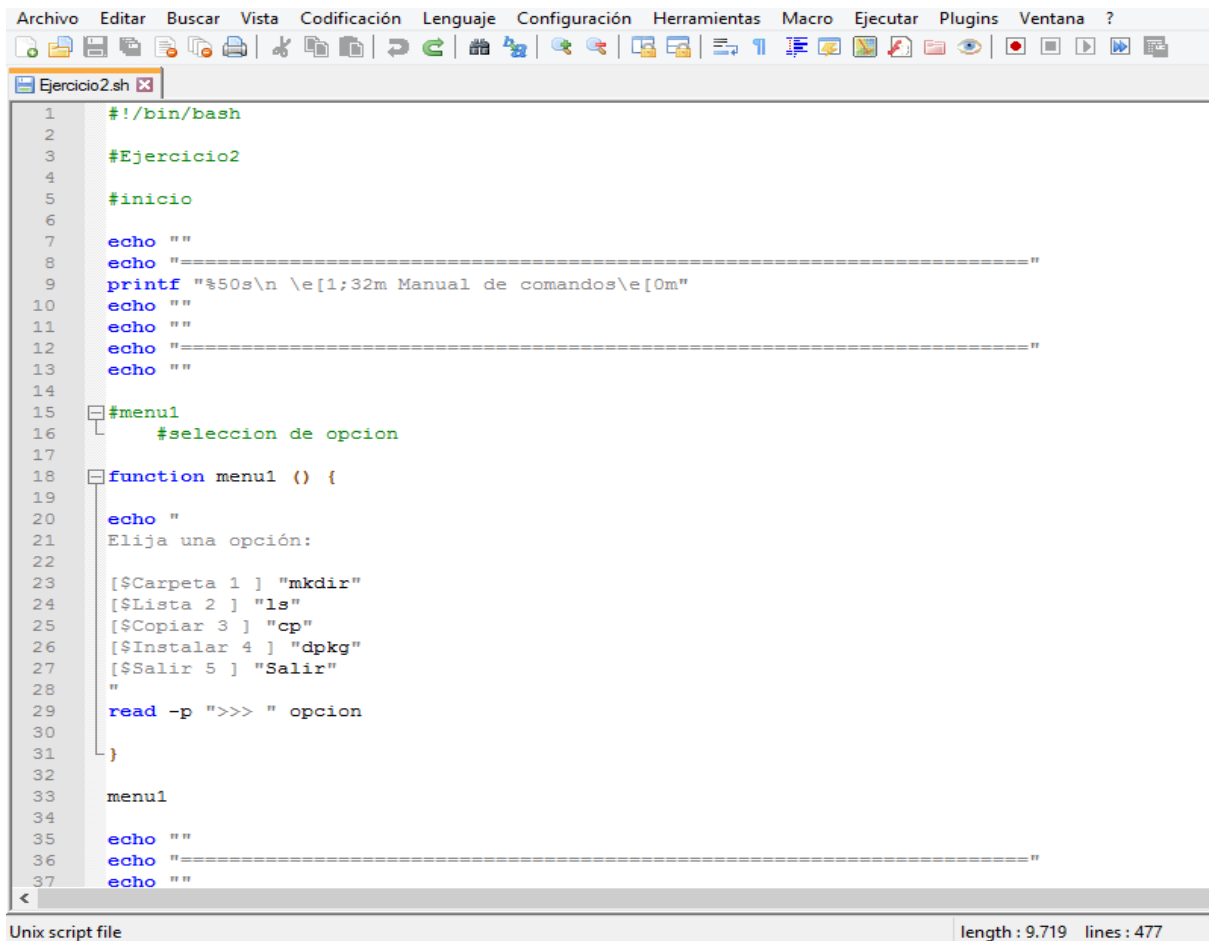
Elija una opción

[1] Dar permisos a una carpeta
[2] Crear varios directorios
[3] Crear un nuevo directorio
[4] Crear una carpeta
[5] Crear una carpeta con un mensaje de confirmacion
[6] Salir

>>> 1
Daremos permisos a una carpeta :
por ejemplo: /home/Carlos/Descargas/
el comando seria : pmkdir -m 777 PARCIAL /home/Carlos/Descargas/

=====
```

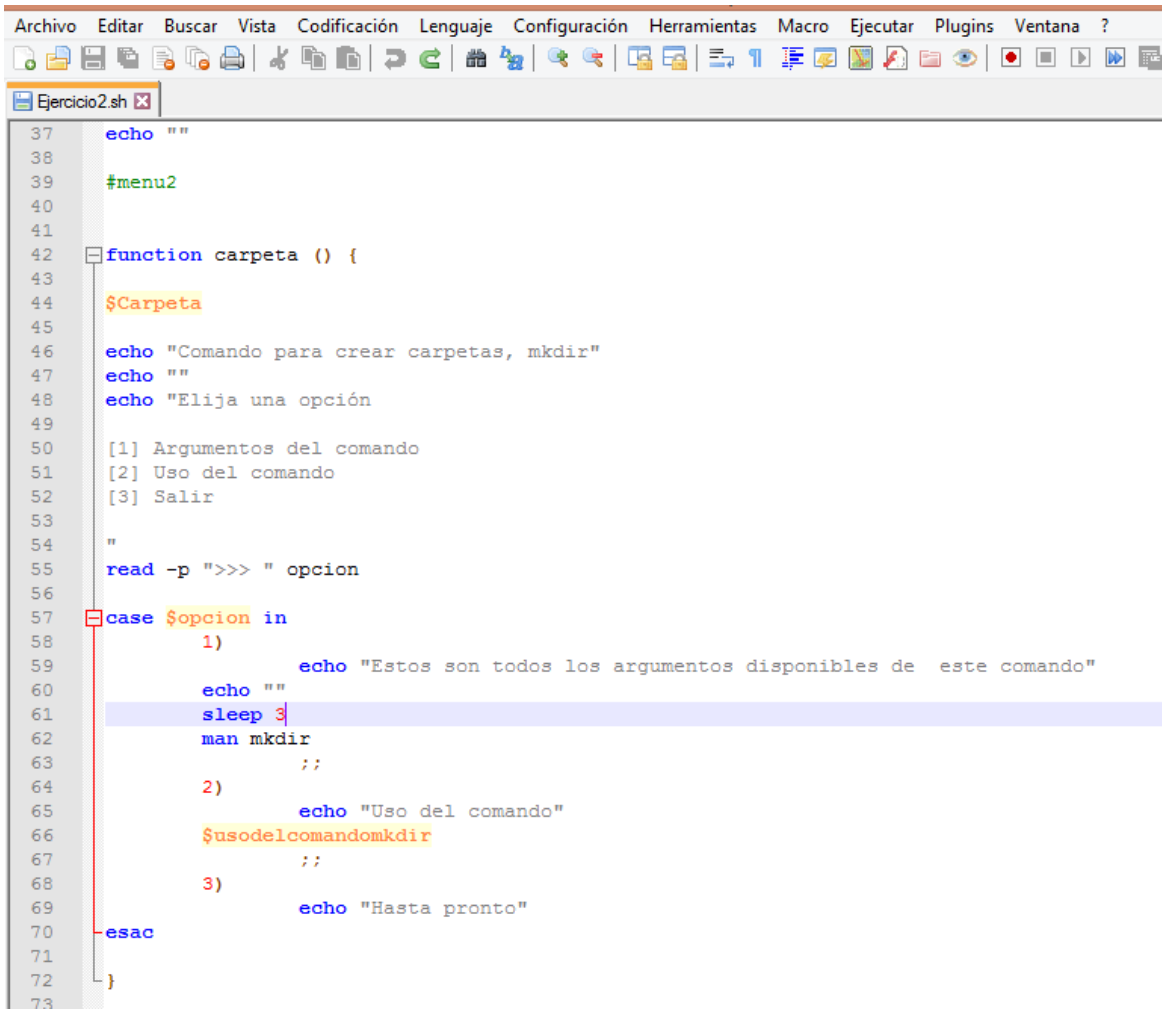
Este es su código de creación, utilizamos “echo” y “printf” para crear y centrar el mensaje del título, luego los “echo ”” para el espacio entre líneas. Y damos inicio a las opciones con las funciones, en mi caso la primera función la nombre menu1 y adentro de esa función cree el menú donde están los comandos para ser seleccionados y entre los corchetes puse las variables de cada opción. Después agregue” read” más” -p” para poder leer la opción que digite el usuario, al lado de los corchetes agregue cada comando que usaremos y la última opción es “Salir”, la cual sirve valga la redundancia para cerrar el Script y para terminar la función le agregue el nombre de nuevo.



```
1  #!/bin/bash
2
3  #Ejercicio2
4
5  #inicio
6
7  echo ""
8  echo "-----"
9  printf "%50s\n \e[1;32m Manual de comandos\e[0m"
10 echo ""
11 echo ""
12 echo "-----"
13 echo ""
14
15 #menu1
16 #seleccion de opcion
17
18 function menu1 () {
19
20     echo "
21     Elija una opción:
22
23     [${Carpeta 1 } ] "mkdir"
24     [${Lista 2 } ] "ls"
25     [${Copiar 3 } ] "cp"
26     [${Instalar 4 } ] "dpkg"
27     [${Salir 5 } ] "Salir"
28     "
29     read -p ">>> " opcion
30
31 }
32
33 menu1
34
35 echo ""
36 echo "-----"
37 echo ""
```

Unix script file | length : 9.719 lines : 477

Aquí agregue otra función, en la cual van las tres opciones disponibles, cree un case en el cual agregue las tres opciones, en la primera di uso del comando “man mkdir” el cual sirve para mostrar los argumentos de cualquier comando y agregue una variable en la opción dos, la llame “usodelcomandomkdir”. Y al final de la función agregue nuevamente su nombre.



```
37 echo ""
38
39 #menu2
40
41
42 function carpeta () {
43
44     $Carpeta
45
46     echo "Comando para crear carpetas, mkdir"
47     echo ""
48     echo "Elija una opción"
49
50     [1] Argumentos del comando
51     [2] Uso del comando
52     [3] Salir
53
54     "
55     read -p ">>> " opcion
56
57     case $opcion in
58         1)
59             echo "Estos son todos los argumentos disponibles de este comando"
60             echo ""
61             sleep 3
62             man mkdir
63             ;;
64         2)
65             echo "Uso del comando"
66             $usodelcomandomkdir
67             ;;
68         3)
69             echo "Hasta pronto"
70     esac
71 }
72
73
```

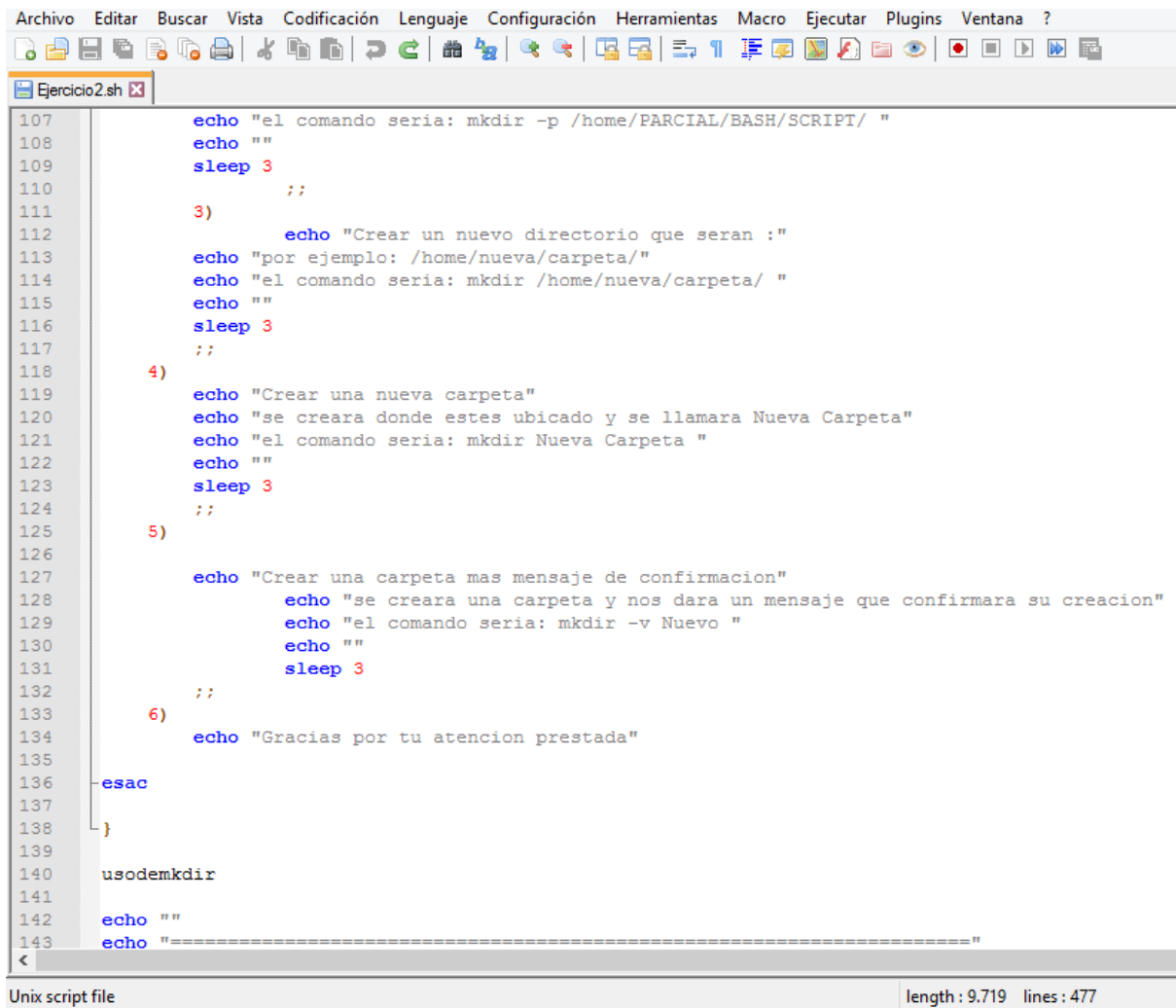


Cree una nueva función en la cual agregue la variable que mencione anteriormente. Y cree otro menú en el cual habrán seis opciones y lo acompañe de un nuevo case con las 6 opciones.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
Ejercicio2.sh x
72  }
73
74  carpeta
75
76  echo ""
77  echo "-----"
78  echo ""
79
80
81  function usodemkdir () {
82
83  $usodelcomandomkdir
84
85  echo "Elija una opción
86
87  [1] Dar permisos a una carpeta
88  [2] Crear varios directorios
89  [3] Crear un nuevo directorio
90  [4] Crear una carpeta
91  [5] Crear una carpeta con un mensaje de confirmacion
92  [6] Salir
93  "
94  read -p ">>> " opcion
95
96  case $opcion in
97  1)
98      echo "Daremos permisos a una carpeta :"
99      echo "por ejemplo: /home/Carlos/Descargas/"
100     echo "el comando seria : pmkdir -m 777 PARCIAL /home/Carlos/Descargas/ "
101     echo ""
102     sleep 3
103     ;;
104  2)
105     echo "Crearemos varios directorios que seran :"
106     echo "por ejemplo: /home/PARCIAL/BASH/SCRIPT/"
107     echo "el comando seria: mkdir -p /home/PARCIAL/BASH/SCRIPT/ "
108     echo ""
<
```

Unix script file length : 9.719 lines : 477

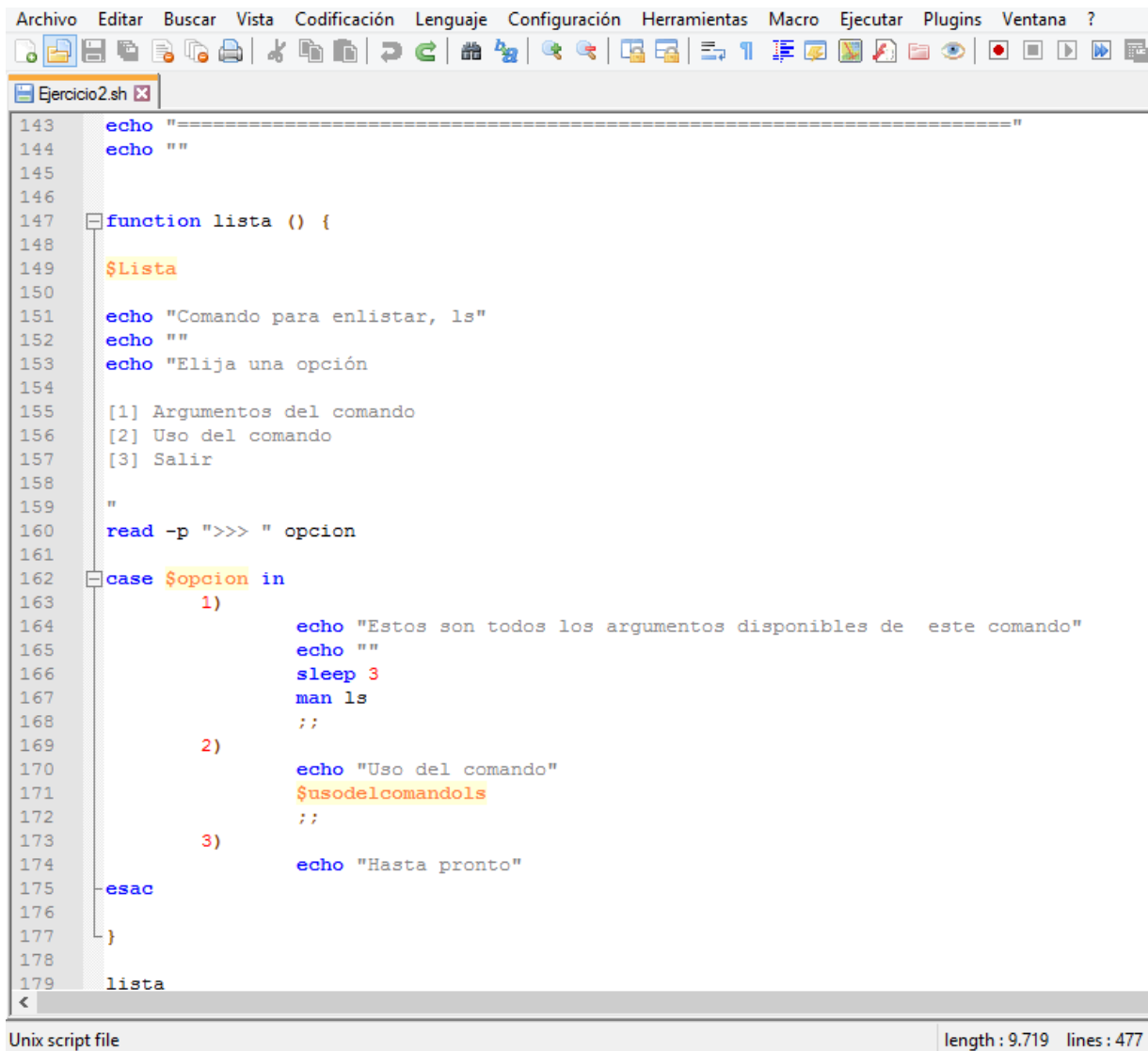
Termine de digitar las seis opciones y por ultimo cerrar "case" con el comando "esac".



```
107     echo "el comando seria: mkdir -p /home/PARCIAL/BASH/SCRIPT/ "
108     echo ""
109     sleep 3
110         ;;
111     3)
112         echo "Crear un nuevo directorio que seran :"
113         echo "por ejemplo: /home/nueva/carpeta/"
114         echo "el comando seria: mkdir /home/nueva/carpeta/ "
115         echo ""
116         sleep 3
117         ;;
118     4)
119         echo "Crear una nueva carpeta"
120         echo "se creara donde estes ubicado y se llamara Nueva Carpeta"
121         echo "el comando seria: mkdir Nueva Carpeta "
122         echo ""
123         sleep 3
124         ;;
125     5)
126
127         echo "Crear una carpeta mas mensaje de confirmacion"
128         echo "se creara una carpeta y nos dara un mensaje que confirmara su creacion"
129         echo "el comando seria: mkdir -v Nuevo "
130         echo ""
131         sleep 3
132         ;;
133     6)
134         echo "Gracias por tu atencion prestada"
135
136 esac
137
138 }
139
140 usodemkdir
141
142 echo ""
143 echo "====="
```

Unix script file length : 9.719 lines : 477

Creación de menú de opciones para la opción dos del menú principal.



```
143 echo "=====
144 echo ""
145
146
147 function lista () {
148
149     $Lista
150
151     echo "Comando para enlistar, ls"
152     echo ""
153     echo "Elija una opción
154
155     [1] Argumentos del comando
156     [2] Uso del comando
157     [3] Salir
158
159     "
160     read -p ">>> " opcion
161
162     case $opcion in
163         1)
164             echo "Estos son todos los argumentos disponibles de este comando"
165             echo ""
166             sleep 3
167             man ls
168             ;;
169         2)
170             echo "Uso del comando"
171             $usodelcomandols
172             ;;
173         3)
174             echo "Hasta pronto"
175     esac
176
177 }
178
179 lista
```

Unix script file length: 9.719 lines: 477

## Creación de opciones para el comando ls.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
Ejercicio2.sh x3
179 lista
180
181 echo ""
182 echo "-----"
183 echo ""
184
185 function usodels () {
186
187     $usodelcomandols
188
189     echo "Elija una opción
190
191     [1] Listar el directorio actual, el directorio Descargas y el directorio /home/Carlos/Descargas/
192     [2] Mostrar archivos y directorios ocultos
193     [3] Mostrar casi todos los archivos
194     [4] Listado largo o Mostrar los detalles de los archivos
195     [5] Mostrar/Listar archivos o directorios en orden inverso
196     [6] Salir
197     "
198     read -p ">>> " opcion
199
200     case $opcion in
201     1)
202         echo " Enlistaremos el directorio actual, el directorio de Descargas y el directorio:"
203         echo "por ejemplo: /home/Carlos/Descargas/"
204         echo "el comando seria: ls . Descargas/ /home/Carlos/Descargas/ "
205         echo ""
206         sleep 3
207         ;;
208     2)
209         echo " Mostraremos archivos y directorios ocultos:"
210         echo "por ejemplo: /usr/local/ es recomendable ser root"
211         echo "el comando seria: ls -a . Descargas/ /usr/local/ "
212         echo ""
213         sleep 3
214         ;;
215     3)
216     <
```

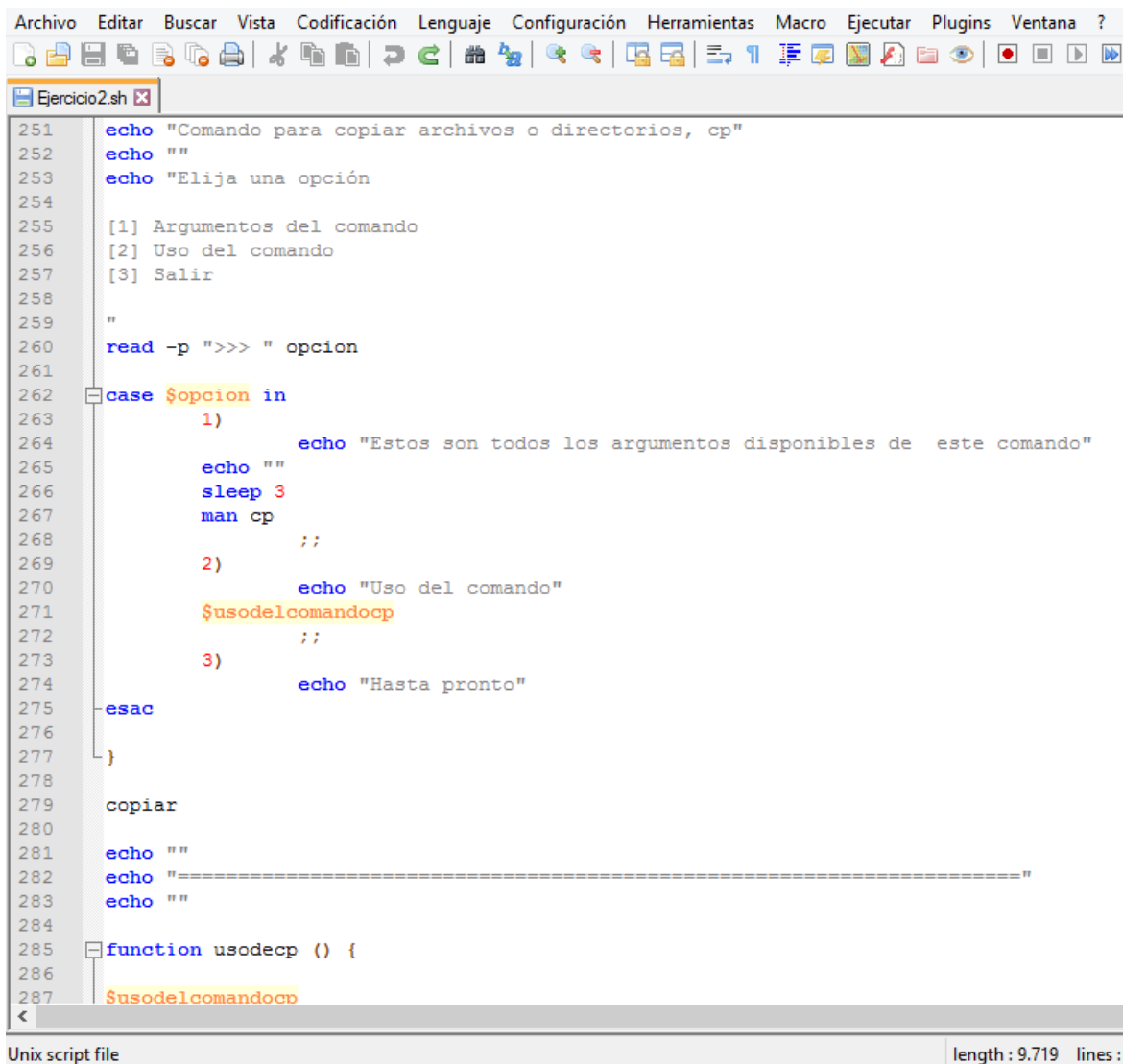
Unix script file length: 9.719 lines: 477 Ln: 213 C

Concluyendo las opciones y creando una nueva función para la siguiente función.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
Ejercicio2.sh x
215         3)
216             echo "Mostraremos casi todos los archivos :"
217         echo "el comando seria: ls -A "
218         echo ""
219         sleep 3
220         ;;
221     4)
222         echo "Listado largo o Mostrar los detalles de los archivos"
223         echo "el comando seria : ls -l "
224         echo ""
225         sleep 3
226         ;;
227     5)
228         echo "Mostrar/Listar archivos o directorios en orden inverso"
229             echo "el comando seria : ls -r "
230             echo ""
231             sleep 3
232         ;;
233     6)
234         echo "Gracias por tu atencion prestada"
235
236     esac
237
238 }
239
240
241 usodels
242
243 echo ""
244 echo "-----"
245 echo ""
246
247 function copiar () {
248     $Copiar
249
250
251     echo "Comando para copiar archivos o directorios. cp"
252 }
<
```

Unix script file length : 9.719 lines : 477

Creación de un nuevo case y una nueva función.



```
251 echo "Comando para copiar archivos o directorios, cp"
252 echo ""
253 echo "Elija una opción"
254
255 [1] Argumentos del comando
256 [2] Uso del comando
257 [3] Salir
258
259 "
260 read -p ">>> " opcion
261
262 case $opcion in
263 1)
264     echo "Estos son todos los argumentos disponibles de este comando"
265     echo ""
266     sleep 3
267     man cp
268     ;;
269 2)
270     echo "Uso del comando"
271     $usodelcomandocp
272     ;;
273 3)
274     echo "Hasta pronto"
275 esac
276 }
277
278 copiar
279
280 echo ""
281 echo "=====
282 echo ""
283
284
285 function usodecp () {
286     $usodelcomandocp
287 }
```

Unix script file length : 9.719 lines :

Nuevo case más opciones disponibles.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
Ejercicio2.sh
287  $usodelcomandocp
288
289  echo "Elija una opción
290
291  [1] Copiar un archivo a un directorio
292  [2] Copiar 2 o más archivos a un directorio
293  [3] Copiar todos los archivos a un directorio
294  [4] Copiar un directorio completo
295  [5] Forzar la copia
296  [6] Salir
297  "
298  read -p ">>> " opcion
299
300  case $opcion in
301  1)
302      echo " Copiar un archivo a un directorio:"
303      echo "Este es el uso más sencillo de del comando cp que es copiar un archivo dentro de un directorio."
304      echo "Ejemplo: cp archivo-01.txt /home/Carlos/Descargas/"
305      sleep 3
306      echo "se copiaría ese archivo a esa dirección"
307      echo ""
308      sleep 3
309      ;;
310  2)
311      echo " Copiar 2 (o más) archivos a un directorio:"
312      echo "El comando linux cp permite indicar la copia de uno o más archivo que serán copiados a un directorio."
313      echo "Ejemplo: cp archivo.txt programa.py ~/bin/programa/"
314      echo ""
315      echo "se copiarían ambos archivos a esa dirección"
316      echo ""
317      sleep 3
318      ;;
319  3)
320      echo "Copiar todos los archivos a un directorio :"
321      echo "Es posible utilizar los caracteres comodines * y para indicar que deseamos copiar cualquier nombr
322      echo ".txt sería: cp *.txt respaldo"
323      echo "Si se deseara realizar la copia de todos los archivos a un directorio se puede hacer con este comando:"
324  <
```

Unix script file | length: 9.719 | lines: 477 | Ln: 213 | Col: 16 | Sel: 0 | 0

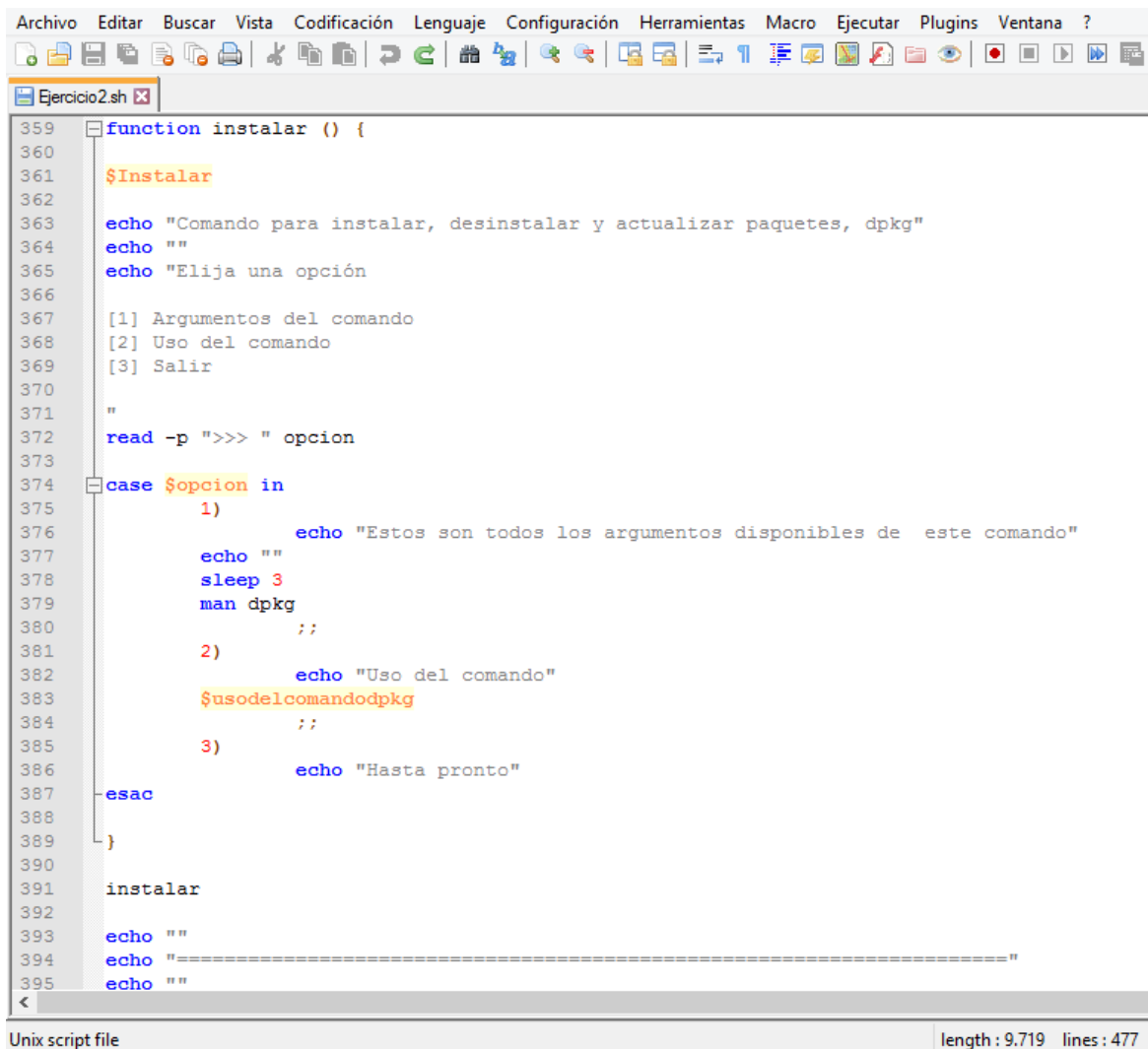
## Conclusión de opciones con case.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
Ejercicio2.sh
323     echo "Si se deseara realizar la copia de todos los archivos a un directorio se puede hacer con este comando:"
324     echo "cp * directorio"
325     echo "Ejemplo: cp * /home/Carlos/Descargas/"
326     echo ""
327     sleep 3
328     ;;
329 4)
330     echo "Copiar un directorio completo"
331     echo "Seria con el comando"
332     echo "cp -r directorio ./respaldo/"
333     echo ""
334     sleep 3
335     ;;
336 5)
337     echo " "
338
339
340     echo "Forzar la copia"
341         echo "Seria con el comando"
342         echo "cp -f archivo.txt ./respaldo/"
343         echo ""
344         sleep 3
345     ;;
346 6)
347     echo "Gracias por tu atencion prestada"
348
349 esac
350
351 }
352
353 usodecp
354
355 echo ""
356 echo "=====
357 echo ""
358
359 function instalar () {
<
```

Unix script file | length : 9.719 | lines : 477 | Ln : 213 | Col : 16 | Sel : 0 | 0



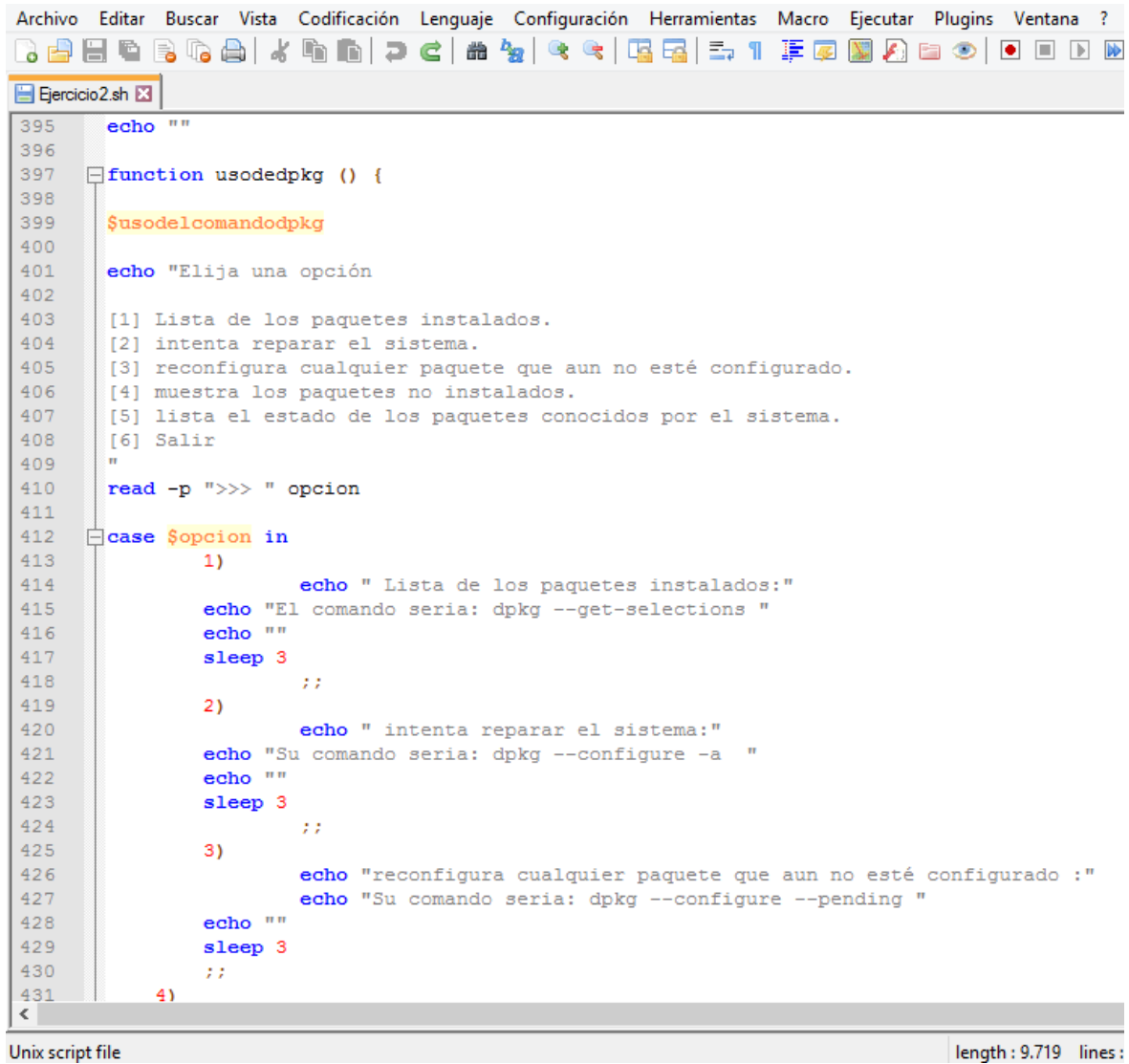
Nueva función y menú.



```
359 function instalar () {
360
361     $Instalar
362
363     echo "Comando para instalar, desinstalar y actualizar paquetes, dpkg"
364     echo ""
365     echo "Elija una opción
366
367     [1] Argumentos del comando
368     [2] Uso del comando
369     [3] Salir
370
371     "
372     read -p ">>> " opcion
373
374     case $opcion in
375         1)
376             echo "Estos son todos los argumentos disponibles de este comando"
377             echo ""
378             sleep 3
379             man dpkg
380             ;;
381         2)
382             echo "Uso del comando"
383             $usodelcomandodpkg
384             ;;
385         3)
386             echo "Hasta pronto"
387     esac
388 }
389
390
391 instalar
392
393 echo ""
394 echo "=====
395 echo ""
```

Unix script file | length: 9.719 | lines: 477

Nueva función de opciones.



```
395 echo ""
396
397 function usodedpkg () {
398
399     $usodelcomandodpkg
400
401     echo "Elija una opción
402
403     [1] Lista de los paquetes instalados.
404     [2] intenta reparar el sistema.
405     [3] reconfigura cualquier paquete que aun no esté configurado.
406     [4] muestra los paquetes no instalados.
407     [5] lista el estado de los paquetes conocidos por el sistema.
408     [6] Salir
409     "
410     read -p ">>> " opcion
411
412     case $opcion in
413         1)
414             echo " Lista de los paquetes instalados:"
415             echo "El comando seria: dpkg --get-selections "
416             echo ""
417             sleep 3
418             ;;
419         2)
420             echo " intenta reparar el sistema:"
421             echo "Su comando seria: dpkg --configure -a "
422             echo ""
423             sleep 3
424             ;;
425         3)
426             echo "reconfigura cualquier paquete que aun no esté configurado : "
427             echo "Su comando seria: dpkg --configure --pending "
428             echo ""
429             sleep 3
430             ;;
431         4)
432     esac
433 }
```

Unix script file length: 9,719 lines:

Conclusión de case, y finalmente la función para salir.

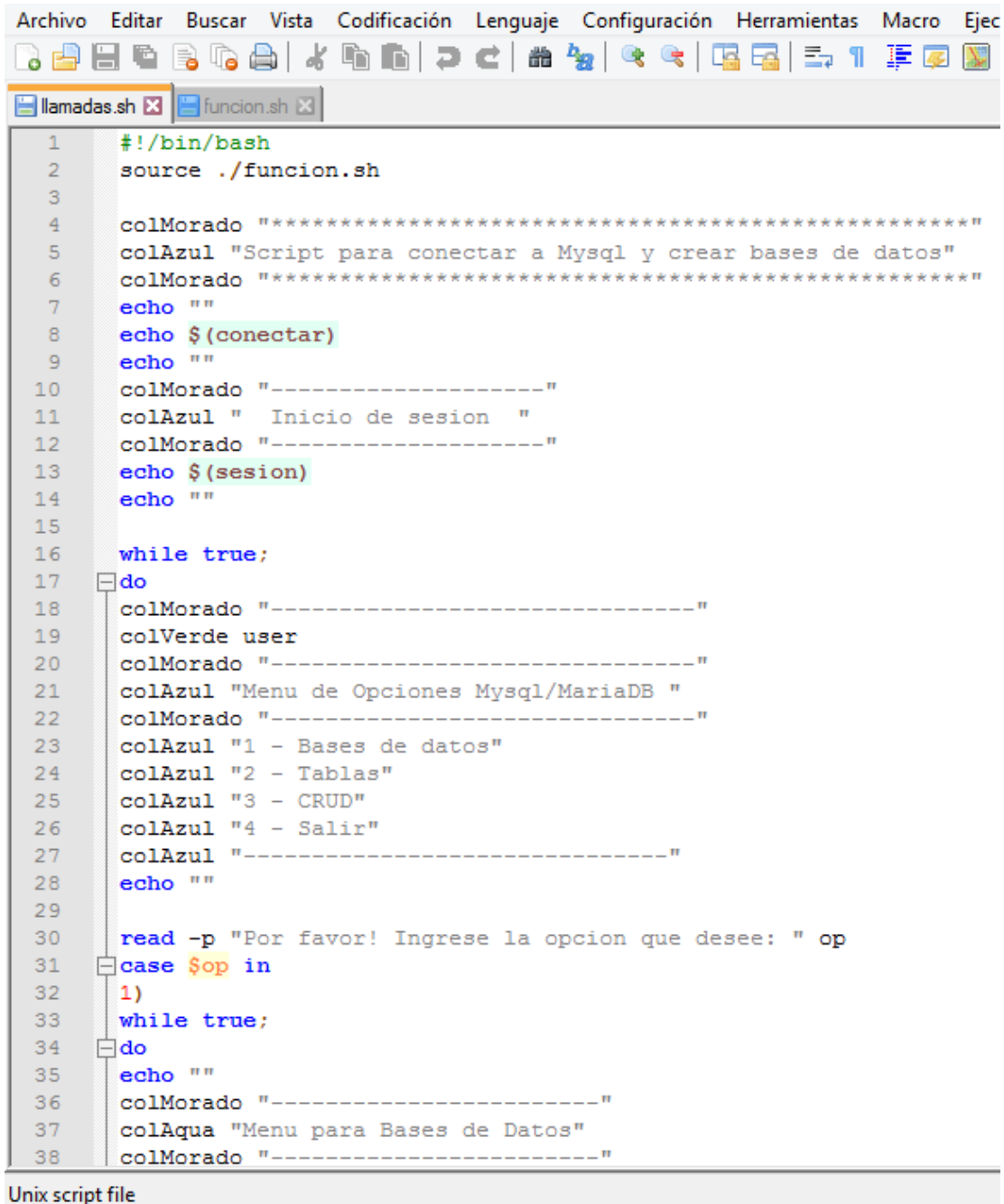
```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Venta
Ejercicio2.sh x
431     4)
432     echo "muestra los paquetes no instalados."
433     echo "Seria con el comando: dpkg -l | grep ^[a-z]n"
434     echo ""
435     sleep 3
436     ;;
437     5)
438     echo " "
439     echo "lista el estado de los paquetes conocidos por el sistema."
440         echo "Seria con el comando: dpkg -l"
441         echo ""
442         sleep 3
443     ;;
444     6)
445     echo "Gracias por tu atencion prestada"
446
447 esac
448 }
449
450
451 usodedpkg
452
453 echo ""
454 echo "-----"
455 echo ""
456
457 function salir () {
458
459     $Salir
460
461     printf "%50s\n \e[1;32m Gracias por su atencion prestada\e[0m"
462     echo ""
463
464 }
465
466 salir
467
<
```

3) Crear un script que se conecte al servidor mysql/mariadb sin ingresar al prompt y poder ejecutar sentencias sql, el script debe poder crear bases de datos, tablas y luego se debe poder insertar datos, consultar datos, actualizar datos, eliminar datos, también debe poder buscar un registro específico en una tabla específica para cada tabla deben hacerse estas funciones mencionadas, también debe poder mostrarse el usuario con el que se está conectando al servidor con una sentencia sql, los títulos deben llevar colores sobre lo que hace el script, además, el script debe permitir poder hacer copias seguridad de una base de datos, de varias bases de datos y de todas las bases de datos según sea el caso que se elija, además al conectarse al servidor mysql/mariadb debe permitir ejecutar el comando `dpkg -l | grep mysql` (el comando debe ser ejecutado como sentencias sql, no directamente como comando bash), también debe permitir restaurar las bases de datos, el script como entrada debe permitir el host (localhost), usuario con el que se conectarán al servidor, la base de datos cuando ya se haya creado, haga uso de funciones, cree todas las funciones que sean necesarias, esto debe hacerse en dos scripts, uno donde se definen todas las funciones (funciones.sh) y otra donde se llaman todas las funciones (llamadas.sh) este último debe tener una buena presentación donde se indique cuáles opciones seleccionar para hacer alguna operación que se quiera ejecutar.

Nota: para hacer la práctica de este ejercicio haga uso de las siguientes tablas: Tabla Alumno (codigo, nombre, apellido, nacionalidad, dirección), Profesor (id\_profesor, nombre, apellido, dirección, dui), Materia (id\_materia, nombre, unidades), turno (id\_turno, turno), carrera (id\_carrera, carrera, duración), calificación (codigo, calificación1, calificación2, calificación3) a cada tabla se le debe ingresar mínimo 10 registros.

## Solución:

Lastimosamente no logre terminar este Script debido a la falta de tiempo, sin embargo el código lo avance hasta lo siguiente: con source ejecutaríamos el segundo script donde van la mayoría de comandos.



```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejec
llamadas.sh x  funcion.sh x
1  #!/bin/bash
2  source ./funcion.sh
3
4  colMorado "*****"
5  colAzul "Script para conectar a Mysql y crear bases de datos"
6  colMorado "*****"
7  echo ""
8  echo $(conectar)
9  echo ""
10 colMorado "-----"
11 colAzul "  Inicio de sesion  "
12 colMorado "-----"
13 echo $(sesion)
14 echo ""
15
16 while true;
17 do
18   colMorado "-----"
19   colVerde user
20   colMorado "-----"
21   colAzul "Menu de Opciones Mysql/MariaDB "
22   colMorado "-----"
23   colAzul "1 - Bases de datos"
24   colAzul "2 - Tablas"
25   colAzul "3 - CRUD"
26   colAzul "4 - Salir"
27   colAzul "-----"
28   echo ""
29
30   read -p "Por favor! Ingrese la opcion que desee: " op
31   case $op in
32   1)
33     while true;
34     do
35       echo ""
36       colMorado "-----"
37       colAqua "Menu para Bases de Datos"
38       colMorado "-----"
```

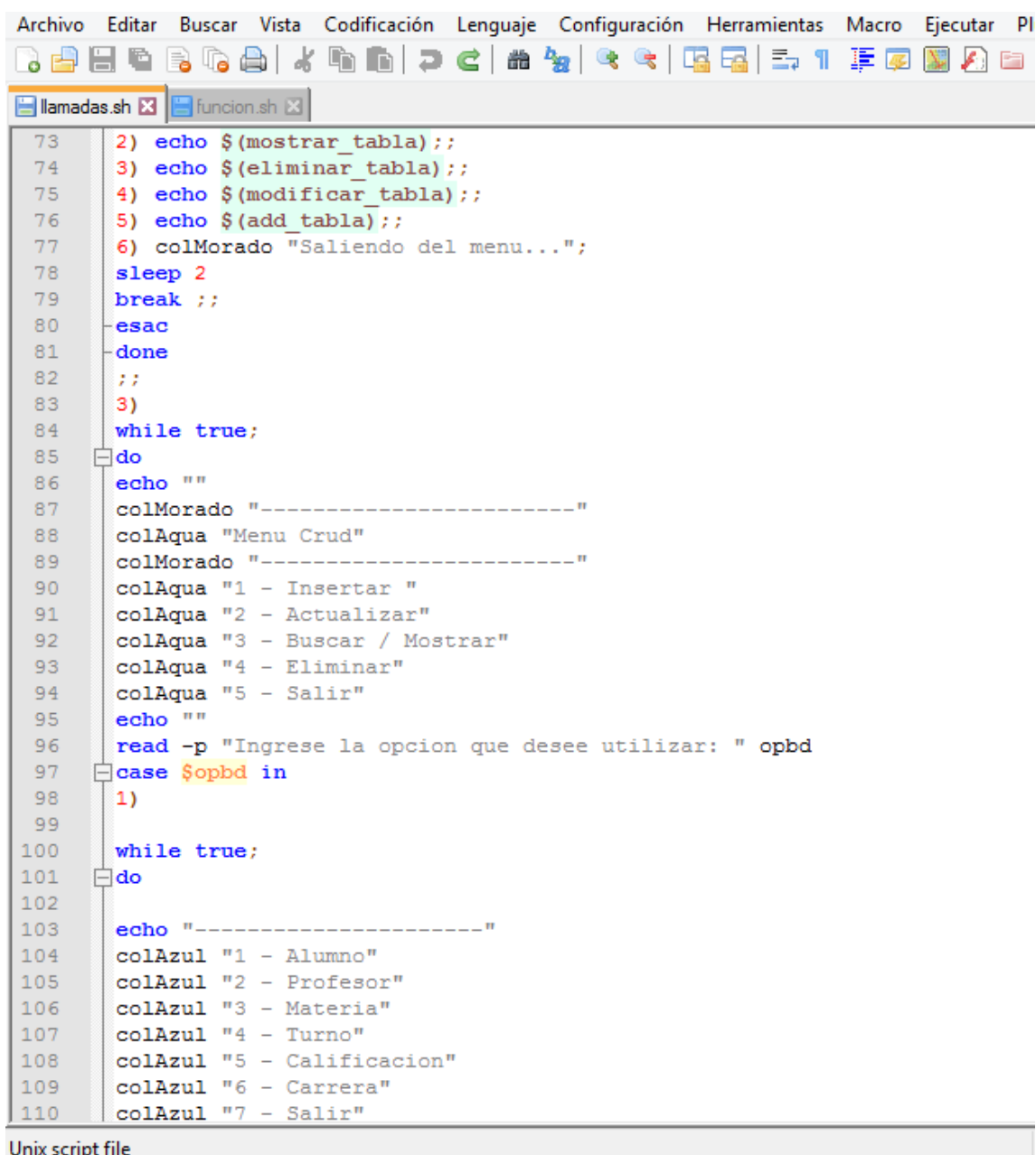
Unix script file

Hicimos uso de case y de while en múltiples ocasiones.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana
llamadas.sh  funcion.sh
36  colMorado "-----"
37  colAqua "Menu para Bases de Datos"
38  colMorado "-----"
39  colAqua "1 - Crear Bases de Datos "
40  colAqua "2 - Mostrar Bases de datos"
41  colAqua "3 - Eliminar Bases de Dtos"
42  colAqua "4 - Salir"
43  echo ""
44
45  read -p "Ingrese la opcion que desee utilizar: " opbd
46  case $opbd in
47  1) echo $(crear_db);;
48  2) echo $(mostrar_db);;
49  3) echo $(eliminar_db);;
50  4) colMorado "Saliendo del menu...";
51  sleep 2
52  break ;;
53  esac
54  done
55  ;;
56  2)
57  while true;
58  do
59  echo ""
60  colMorado "-----"
61  colAqua "Menu para Tablas"
62  colMorado "-----"
63  colAqua "1 - Crear tabla "
64  colAqua "2 - Mostrar tabla"
65  colAqua "3 - Eliminar tabla"
66  colAqua "4 - Modificar tabla"
67  colAqua "5 - Añadir a tabla"
68  colAqua "6 - Salir"
69  echo ""
70  read -p "Ingrese la opcion que desee utilizar: " opbd
71  case $opbd in
72  1) echo $(crear_tabla);;
73  2) echo $(mostrar_tabla);;
```

Unix script file | length: 3.488 |

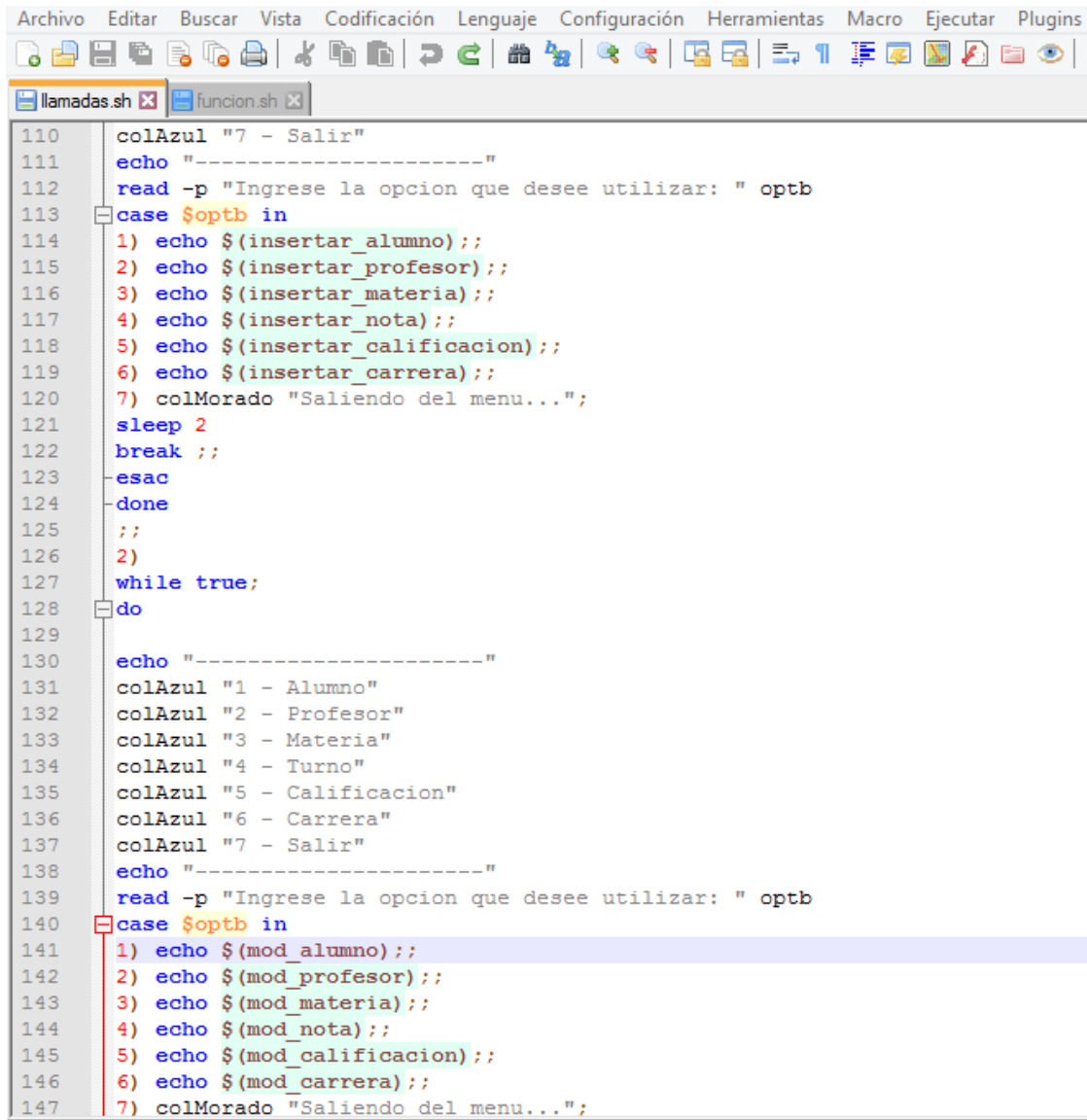
Creación de case para menús de opciones.



```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Pl
llamadas.sh  funcion.sh
73  2) echo $(mostrar_tabla);;
74  3) echo $(eliminar_tabla);;
75  4) echo $(modificar_tabla);;
76  5) echo $(add_tabla);;
77  6) colMorado "Saliendo del menu...";
78  sleep 2
79  break ;;
80  esac
81  done
82  ;;
83  3)
84  while true;
85  do
86  echo ""
87  colMorado "-----"
88  colAqua "Menu Crud"
89  colMorado "-----"
90  colAqua "1 - Insertar "
91  colAqua "2 - Actualizar"
92  colAqua "3 - Buscar / Mostrar"
93  colAqua "4 - Eliminar"
94  colAqua "5 - Salir"
95  echo ""
96  read -p "Ingrese la opcion que desee utilizar: " opbd
97  case $opbd in
98  1)
99
100  while true;
101  do
102
103  echo "-----"
104  colAzul "1 - Alumno"
105  colAzul "2 - Profesor"
106  colAzul "3 - Materia"
107  colAzul "4 - Turno"
108  colAzul "5 - Calificacion"
109  colAzul "6 - Carrera"
110  colAzul "7 - Salir"
```

Unix script file

Menú de opciones.



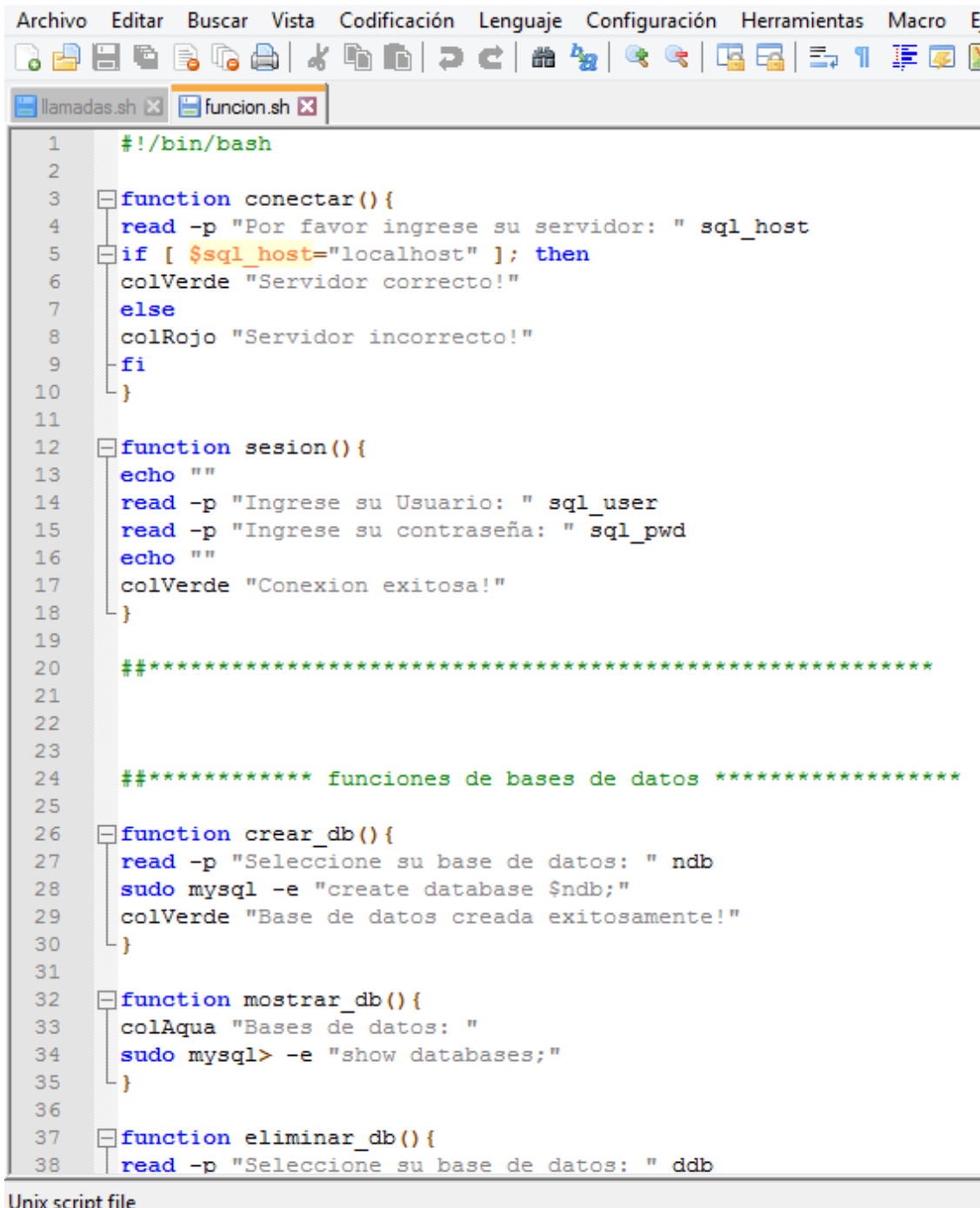
```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins
llamadas.sh x  funcion.sh x
110  colAzul "7 - Salir"
111  echo "-----"
112  read -p "Ingrese la opcion que desee utilizar: " optb
113  case $optb in
114  1) echo $(insertar_alumno);;
115  2) echo $(insertar_profesor);;
116  3) echo $(insertar_materia);;
117  4) echo $(insertar_notas);;
118  5) echo $(insertar_calificacion);;
119  6) echo $(insertar_carrera);;
120  7) colMorado "Saliendo del menu...";
121  sleep 2
122  break ;;
123  esac
124  done
125  ;;
126  2)
127  while true;
128  do
129
130  echo "-----"
131  colAzul "1 - Alumno"
132  colAzul "2 - Profesor"
133  colAzul "3 - Materia"
134  colAzul "4 - Turno"
135  colAzul "5 - Calificacion"
136  colAzul "6 - Carrera"
137  colAzul "7 - Salir"
138  echo "-----"
139  read -p "Ingrese la opcion que desee utilizar: " optb
140  case $optb in
141  1) echo $(mod_alumno);;
142  2) echo $(mod_profesor);;
143  3) echo $(mod_materia);;
144  4) echo $(mod_notas);;
145  5) echo $(mod_calificacion);;
146  6) echo $(mod_carrera);;
147  7) colMorado "Saliendo del menu...";
```



Varios case para generar opciones.

```
Archivo  Editar  Buscar  Vista  Comunicación  Lenguaje  Configuración  Herramientas  n
llamadas.sh x  funcion.sh x
136 colAzul "6 - Carrera"
137 colAzul "7 - Salir"
138 echo "-----"
139 read -p "Ingrese la opcion que desee utilizar: " optb
140 case $optb in
141 1) echo $(mod_alumno) ;;
142 2) echo $(mod_profesor) ;;
143 3) echo $(mod_materia) ;;
144 4) echo $(mod_nota) ;;
145 5) echo $(mod_calificacion) ;;
146 6) echo $(mod_carrera) ;;
147 7) colMorado "Saliendo del menu...";
148 sleep 2
149 break ;;
150 esac
151 done
152 ;;
153 3) echo $(buscar_datos) ;;
154 4) echo $(eliminar_datos) ;;
155 5) colMorado "Saliendo del menu...";
156 sleep 2
157 break ;;
158 esac
159 done
160 ;;
161
162
163 4) colRojo "Saliendo del Script"
164 colMorado "Bye!";
165 sleep 3
166 exit ;;
167 *)
168 colRojo "Debe ingresar una opcion valida:" ;;
169 esac
170 done
171
172
Unix script file
```

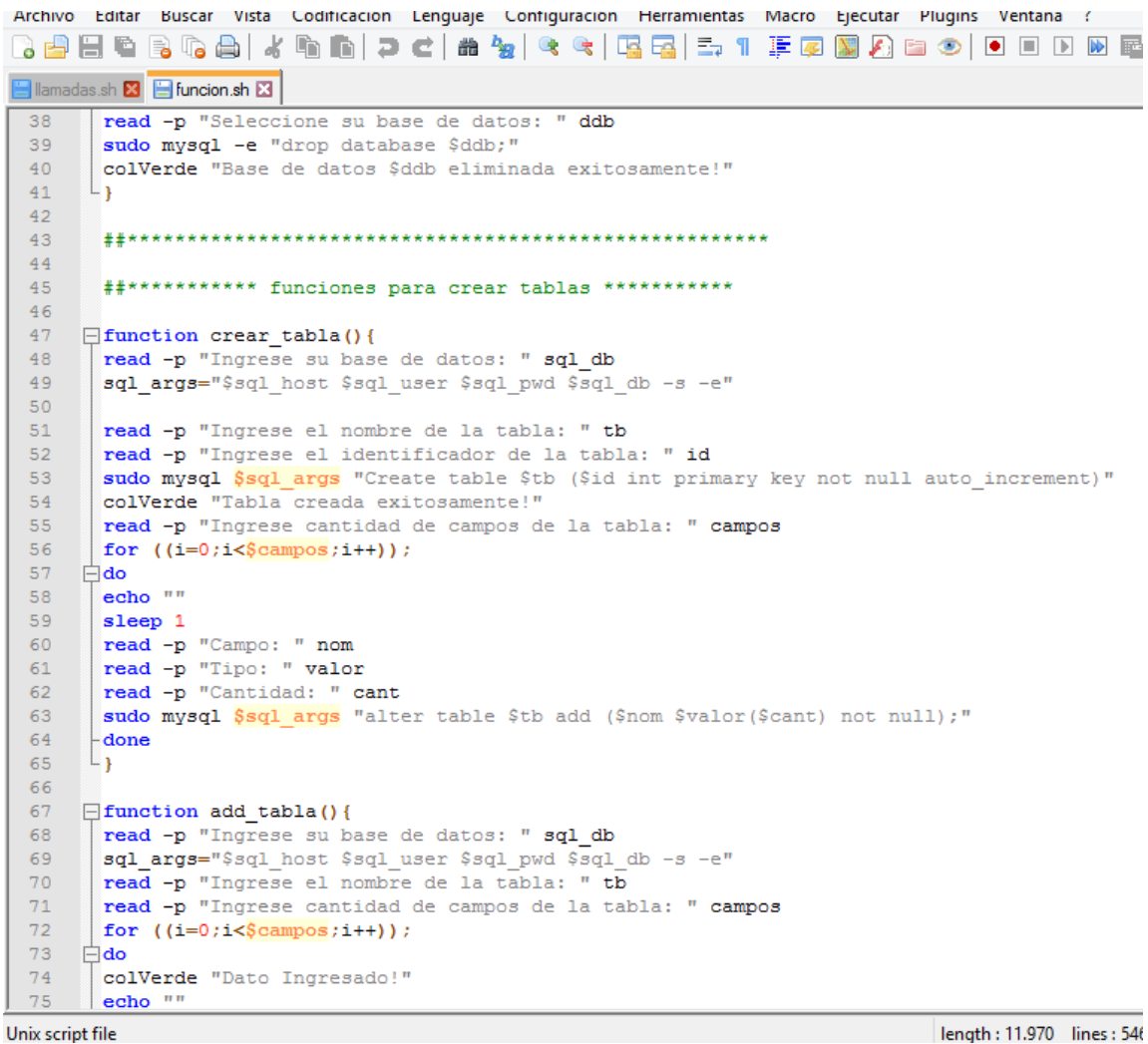
Este sería el segundo archivo “.sh”, nombrado “función.sh”, en el cual van todas las opciones a ejecutar.



```
1  #!/bin/bash
2
3  function conectar() {
4  read -p "Por favor ingrese su servidor: " sql_host
5  if [ $sql_host="localhost" ]; then
6  colVerde "Servidor correcto!"
7  else
8  colRojo "Servidor incorrecto!"
9  fi
10 }
11
12 function sesion() {
13 echo ""
14 read -p "Ingrese su Usuario: " sql_user
15 read -p "Ingrese su contraseña: " sql_pwd
16 echo ""
17 colVerde "Conexion exitosa!"
18 }
19
20 #####
21
22
23
24 ##### funciones de bases de datos #####
25
26 function crear_db() {
27 read -p "Seleccione su base de datos: " ndb
28 sudo mysql -e "create database $ndb;"
29 colVerde "Base de datos creada exitosamente!"
30 }
31
32 function mostrar_db() {
33 colAqua "Bases de datos: "
34 sudo mysql> -e "show databases;"
35 }
36
37 function eliminar_db() {
38 read -p "Seleccione su base de datos: " ddb
```

Unix script file

## Funciones para crear tablas.



```
38 read -p "Seleccione su base de datos: " ddb
39 sudo mysql -e "drop database $ddb;"
40 colVerde "Base de datos $ddb eliminada exitosamente!"
41 }
42
43 #####
44
45 ##### funciones para crear tablas #####
46
47 function crear_tabla(){
48 read -p "Ingrese su base de datos: " sql_db
49 sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
50
51 read -p "Ingrese el nombre de la tabla: " tb
52 read -p "Ingrese el identificador de la tabla: " id
53 sudo mysql $sql_args "Create table $tb ($id int primary key not null auto_increment)"
54 colVerde "Tabla creada exitosamente!"
55 read -p "Ingrese cantidad de campos de la tabla: " campos
56 for ((i=0;i<$campos;i++));
57 do
58 echo ""
59 sleep 1
60 read -p "Campo: " nom
61 read -p "Tipo: " valor
62 read -p "Cantidad: " cant
63 sudo mysql $sql_args "alter table $tb add ($nom $valor($cant) not null);"
64 done
65 }
66
67 function add_tabla(){
68 read -p "Ingrese su base de datos: " sql_db
69 sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
70 read -p "Ingrese el nombre de la tabla: " tb
71 read -p "Ingrese cantidad de campos de la tabla: " campos
72 for ((i=0;i<$campos;i++));
73 do
74 colVerde "Dato Ingresado!"
75 echo ""
```

Unix script file length : 11.970 lines : 54

## Funciones referidas a tablas.

```
Archivo  Editar  Buscar  Vista  Codificacion  Lenguaje  Configuracion  Herramientas  Macro  Ejecutar  Plugins  Ventas
llamadas.sh  funcion.sh
75  echo ""
76  sleep 1
77  read -p "Campo: " nom
78  read -p "Tipo: " valor
79  read -p "Cantidad: " cant
80  sudo mysql $sql_args "alter table $tb add ($nom $valor($cant) not null);"
81  done
82  }
83
84  function mostrar_tabla(){
85  read -p "Ingrese base de datos: " sql_db
86  sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
87  colAzul "Tabla: "
88  sudo mysql $sql_args "Show tables;"
89  }
90
91  function eliminar_tabla(){
92  read -p "Ingrese la base de datos: " sql_db
93  sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
94  read -p "Ingrese l atabla que quiere eliminar: " tb
95  sudo mysql $sql_args "drop table $tb;"
96  colVerde "Tabla $tb Eliminada exitosamente!"
97  }
98
99  function modificar_tabla(){
100 read -p "Ingrese su base de datos: " sql_db
101 sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
102 read -p "Nombre de la tabla: " tb
103 read -p "Nombre: " nom
104 read -p "Nombre nuevo: " nuevo
105 read -p "Tipo: " t
106 read -p "Cantidad: " c
107 sudo mysql $sql_args "alter table $tb change $nom $nuevo $t($c) not null;"
108 colVerde "Modificado exitosamente!"
109 }
110 ##*****
111
112 ## Funciones para Insertar datos en las tablas
```

## Funciones para insertar datos en las tablas.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
lanadas.sh  funcion.sh
112  ## Funciones para Insertar datos en las tablas
113  function insertar_alumno(){
114  read -p "Ingrese el nombre de la base de datos: " sql_db
115  sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
116  read -p "Ingrese el nombre de la tabla: " tb
117  read -p "Ingrese la cantidad de Registros: " n
118  for ((i=0;i<$n;i++));
119  do
120  sleep 2
121  read -p "Nombre del alumno: " nom
122  read -p "Apellido del alumno: " ape
123  read -p "Nacionalidad: " nac
124  read -p "Dirección: " dir
125  sudo mysql $sql_args "Insert into $tb(nombre, apellido, nacionalidad, direccion) values ('$nom','$ape','$nac','$dir')"
126  done
127  colVerde "Exelente! Datos ingresados correctamente"
128  }
129
130  function insertar_profesor(){
131  read -p "Ingrese el nombre de la base de datos: " sql_db
132  sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
133  read -p "Ingrese el nombre de la tabla: " tb
134  read -p "Ingrese la cantidad de Registros: " n
135  for ((i=0;i<$n;i++));
136  do
137  sleep 2
138  read -p "Nombre del profesor: " nom
139  read -p "Apellido del profesor: " ape
140  read -p "Direccion: " dir
141  read -p "Dui: " dui
142  sudo mysql $sql_args "Insert into $tb(nombre, apellido, direccion, dui) values ('$nom','$ape','$dir','$dui')"
143  done
144  colVerde "Exelente! Datos ingresados correctamente"
145  }
146
147
148  ## Tabla Materia
149  function insertar_materia(){
```

Tabla turno.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
llamadas.sh  funcion.sh
149  function insertar_materia(){
150      read -p "Ingrese el nombre de la base de datos: " sql_db
151      sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
152      read -p "Ingrese el nombre de la tabla: " tb
153      read -p "Ingrese la cantidad de Registros: " n
154      for ((i=0;i<$n;i++));
155          do
156              sleep 2
157              read -p "Nombre de la materia: " nom
158              read -p "Unidades: " uni
159              sudo mysql $sql_args "Insert into $tb(nombre, unidades) values ('$nom','$uni')"
160          done
161      colVerde "Exelente! Datos ingresados correctamente"
162  }
163
164
165
166
167
168  ## Tabla turno
169
170  function insertar_turno(){
171      read -p "Ingrese el nombre de la base de datos: " sql_db
172      sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
173      read -p "Ingrese el nombre de la tabla: " tb
174      read -p "Ingrese la cantidad de Registros: " n
175      for ((i=0;i<$n;i++));
176          do
177              sleep 2
178              read -p "Ingrese turno (Mañana/Tarde): " turno
179              sudo mysql $sql_args "Insert into $tb(turno) values ('$turno')"
180          done
181      colVerde "Exelente! Datos ingresados correctamente"
182  }
183
184
185
186  ## Tabla carrera
  
```

## Funciones para tablas.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
tamadas.sh  funcion.sh
186  ## Tabla carrera
187
188  function insertar_carrera(){
189      read -p "Ingrese el nombre de la base de datos: " sql_db
190      sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
191      read -p "Ingrese el nombre de la tabla: " tb
192      read -p "Ingrese la cantidad de Registros: " n
193      for ((i=0;i<$n;i++));
194      do
195          sleep 2
196          read -p "Carrera: " car
197          read -p "Duración: " tiempo
198          sudo mysql $sql_args "Insert into $tb(carrera, duracion) values ('$car','$tiempo')"
199      done
200      colVerde "Excelente! Datos ingresados correctamente"
201  }
202
203
204
205  ## Tabla calificacion
206
207  function insertar_calificacion(){
208      read -p "Ingrese el nombre de la base de datos: " sql_db
209      sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
210      read -p "Ingrese el nombre de la tabla: " tb
211      read -p "Ingrese la cantidad de Registros: " n
212      for ((i=0;i<$n;i++));
213      do
214          sleep 2
215          read -p "Calificacion 1: " c1
216          read -p "Calificacion 2: " c2
217          read -p "Calificacion 3: " c3
218          sudo mysql $sql_args "Insert into $tb(calificacion1, calificacion2, calificacion3) values ('$c1','$c2','$c3')"
219      done
220      colVerde "Excelente! Datos ingresados correctamente"
221  }
222
223  function mostrar_registros(){
```

## Funciones para eliminar y actualizar registros.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
lamadas.sh  funcion.sh
223 function mostrar_registros(){
224     read -p "Ingrese su base de datos: " sql_db
225     sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
226     read -p "Ahora Ingrese el nombre de la tabla: " tb
227     colVerde "Datos: "
228     sudo mysql $sql_args "Select * from $tb"
229 }
230
231 ## Funciones para eliminar Registros
232 function eliminar_datos(){
233     read -p "Ingrese su base de datos: " sql_db
234     sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
235     read -p "Ahora Ingrese el nombre de la tabla: " tb
236     sudo mysql $sql_args "Select * from $tb"
237     read -p "Por favor! Ingrese el N° de Id para eliminar el Registro: " id
238     sudo mysql $sql_args "Delete from $tb where id=$id"
239     colVerde "Datos eliminados exitosamente."
240 }
241
242
243 ## Funciones para actualizar Registros de las tablas
244
245
246 function mod_alumno(){
247     read -p "Ingrese el nombre de la base de datos: " sql_db
248     sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
249     read -p "Ingrese el nombre de la tabla: " tb
250     read -p "Ingrese la cantidad de Registros: " n
251     for ((i=0;i<$n;i++));
252     do
253         sleep
254         read -p "Ingrese el id del registro: " iden
255         read -p "Nombre del alumno: " nom
256         read -p "Apellido del alumno: " ape
257         read -p "Nacionalidad: " nac
258         read -p "Dirección: " dir
259         sudo mysql $sql_args "update $tb(nombre, apellido, nacionalidad, direccion) set ('$nom','$ape','$nac','$dir') where id=$iden"
260     done

```

Univerzitat

length: 11 070 line: 516

lin: 528 col: 1 sel: 010

Univ (E)



## Tabla de profesor.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
funcion.sh
260     done
261     echo -e "\e[1;32mExcelente! Datos ingresados correctamente\e[0m"
262
263 }
264
265
266 ## Tabla Profesor
267
268 function mod_profesor(){
269
270     read -p "Ingrese el nombre de la base de datos: " sql_db
271
272
273
274     sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
275
276     read -p "Ingrese el nombre de la tabla: " tb
277
278
279
280     read -p "Ingrese la cantidad de Registros: " n
281
282     for ((i=0;i<$n;i++));
283
284     do
285
286         sleep
287
288         read -p "Nombre del profesor: " nom
289
290         read -p "Apellido del profesor: " ape
291
292         read -p "Direccion: " dir
293
294         read -p "Dui: " dui
295
296         sudo mysql $sql_args "update $tb(nombre, apellido, direccion, dui) set ('$nom','$ape','$dir','$dui')"
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

Unix script file | length: 11.970 | lines: 546 | Ln: 528 | Col: 1 | Sel: 010

## Tabla de materia.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
llamadas.sh  funcion.sh
297
298     done
299
300     echo -e "\e[1;32mExelente! Datos ingresados correctamente\e[0m"
301
302 }
303
304
305
306
307
308 ## Tabla Materia
309
310 function mod_materia() {
311
312     read -p "Ingrese el nombre de la base de datos: " sql_db
313
314
315
316     sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
317
318     read -p "Ingrese el nombre de la tabla: " tb
319
320
321
322     read -p "Ingrese la cantidad de Registros: " n
323
324     for ((i=0;i<$n;i++));
325
326     do
327
328         sleep
329
330         read -p "Nombre de la materia: " nom
331
332         read -p "Unidades: " uni
333
334         sudo mysql $sql_args "update $tb(nombre, unidades) set ('$nom','$uni')"
```

Tabla turno.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
llamadas.sh  funcion.sh
334         sudo mysql $sql_args "update $tb(nombre, unidades) set ('$nom','$uni')"
335
336         done
337
338     echo -e "\e[1;32mExelente! Datos ingresados correctamente\e[0m"
339
340 }
341
342
343
344
345
346 ## Tabla turno
347
348 function mod_turno(){
349
350     read -p "Ingrese el nombre de la base de datos: " sql_db
351
352
353
354     sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
355
356     read -p "Ingrese el nombre de la tabla: " tb
357
358
359
360     read -p "Ingrese la cantidad de Registros: " n
361
362     for ((i=0;i<$n;i++));
363
364     do
365
366         sleep
367
368         read -p "Ingrese turno (Mañana/Tarde): " turno
369
370         sudo mysql $sql_args "update $tb(turno) set ('$turno')"
371
```

Tabla de carrera.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
llamadas.sh  funcion.sh
371
372     done
373
374     echo -e "\e[1;32mExelente! Datos ingresados correctamente\e[0m"
375
376 }
377
378
379
380 ## Tabla carrera
381
382 function mod_carrera(){
383
384     read -p "Ingrese el nombre de la base de datos: " sql_db
385
386
387
388     sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
389
390     read -p "Ingrese el nombre de la tabla: " tb
391
392
393
394     read -p "Ingrese la cantidad de Registros: " n
395
396     for ((i=0;i<$n;i++));
397
398     do
399
400         sleep
401
402         read -p "Carrera: " car
403
404         read -p "Duración: " tiempo
405
406         sudo mysql $sql_args "update $tb(carrera, duracion) set ('$car','$tiempo')"
407
408     done
```

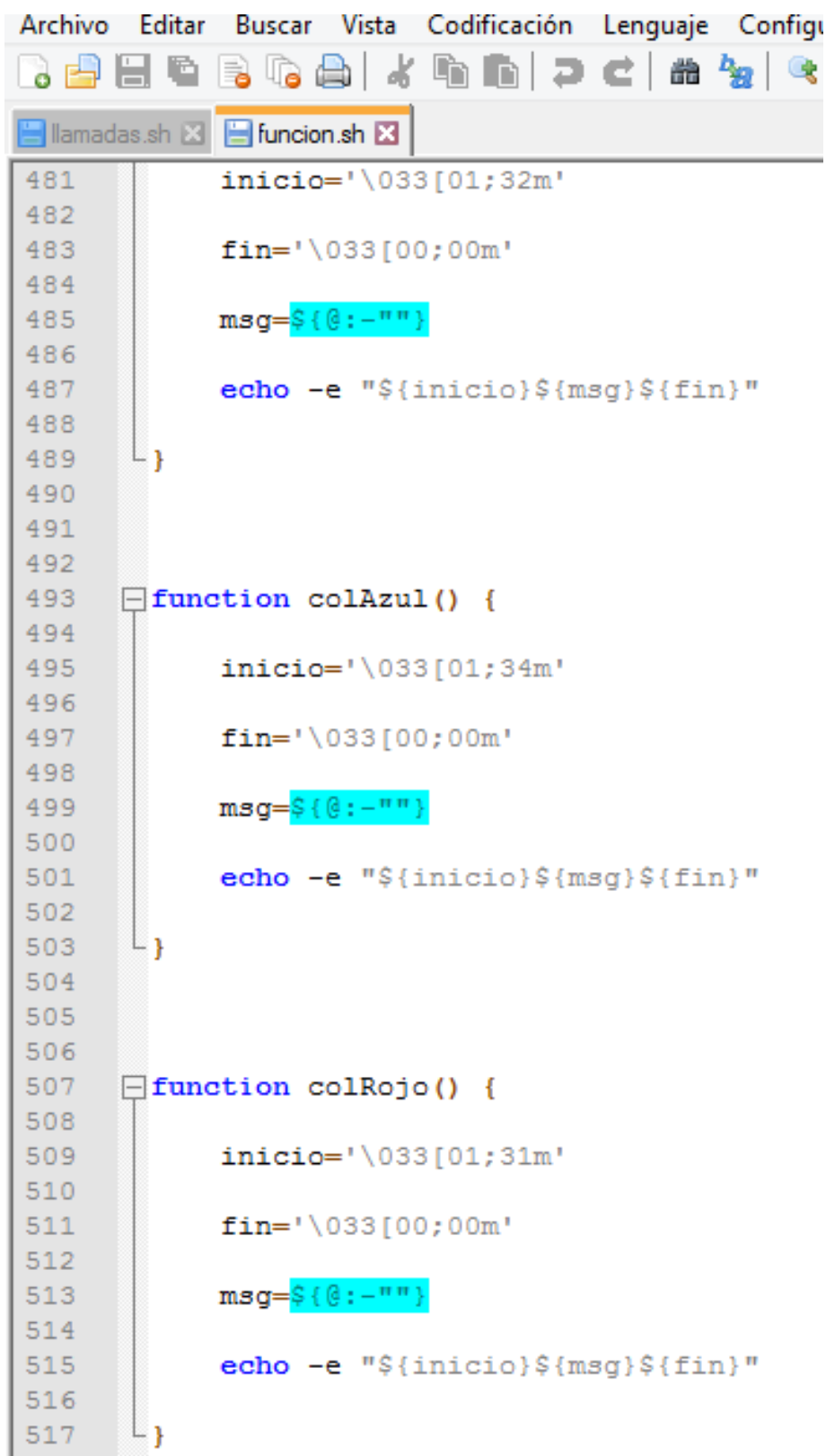
## Creación de tabla de calificación.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
funcion.sh
408     done
409
410     echo -e "\e[1;32mExelente! Datos ingresados correctamente\e[0m"
411
412 }
413
414
415
416 ## Tabla calificacion
417
418 function mod_calificacion(){
419
420     read -p "Ingrese el nombre de la base de datos: " sql_db
421
422
423
424     sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
425
426     read -p "Ingrese el nombre de la tabla: " tb
427
428
429
430     read -p "Ingrese la cantidad de Registros: " n
431
432     for ((i=0;i<$n;i++));
433
434     do
435
436         sleep
437
438         read -p "Calificacion 1: " c1
439
440         read -p "Calificacion 2: " c2
441
442         read -p "Calificacion 3: " c3
443
444         sudo mysql $sql_args "update $tb(calificacion1, calificacion2, calificacion3) set ('$c1','$c2','$c3')"
445
```

## Funciones de registros.

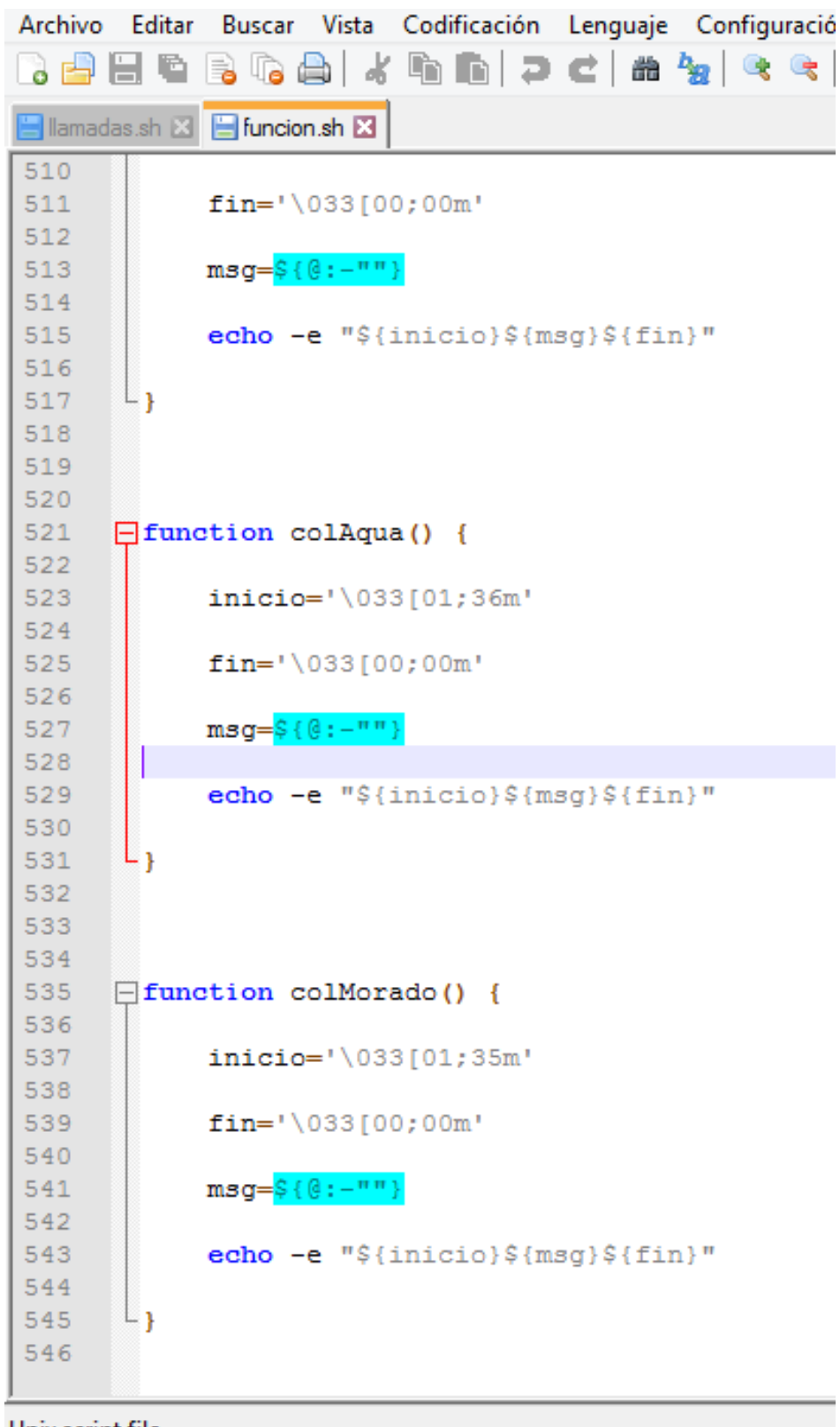
```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
tamadas.sh  funcion.sh
444         sudo mysql $sql_args "update $tb(calificacion1, calificacion2, calificacion3) set ('$c1','$c2','$c3')"
445     done
446
447     echo -e "\e[1;32mExcelente! Datos ingresados correctamente\e[0m"
448
449 }
450
451
452
453
454 ## Funciones para Buscar Registros
455 function modtar_dato(){
456     read -p "Ingrese su base de datos: " sql_db
457     sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
458     read -p "Ahora Ingrese el nombre de la tabla: " tb
459     sudo mysql $sql_args "Select * from $tb"
460 }
461
462 function buscar_dato(){
463     read -p "Ingrese su base de datos: " sql_db
464     sql_args="$sql_host $sql_user $sql_pwd $sql_db -s -e"
465     read -p "Ahora Ingrese el nombre de la tabla: " tb
466     read -p "Ingrese el dato por el cual desea buscar: " dato
467     read -p "Ingrese los datos por los cuales hara la busqueda (Nombre/Apellido/Nacionalidad/direccion): " tip
468     sudo mysql $sql_args "Select $dato from $tb where $tip like%"
469 }
470
471
472
473
474
475 ## funciones para colores
476
477
478
479 function colVerde() {
480     inicio='\033[01;32m'
481
```

Estas serían las penúltimas funciones.



```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configu
llamadas.sh x  funcion.sh x
481     inicio='\033[01;32m'
482
483     fin='\033[00;00m'
484
485     msg=${@:-""}
486
487     echo -e "${inicio}${msg}${fin}"
488
489 }
490
491
492
493 function colAzul() {
494
495     inicio='\033[01;34m'
496
497     fin='\033[00;00m'
498
499     msg=${@:-""}
500
501     echo -e "${inicio}${msg}${fin}"
502
503 }
504
505
506
507 function colRojo() {
508
509     inicio='\033[01;31m'
510
511     fin='\033[00;00m'
512
513     msg=${@:-""}
514
515     echo -e "${inicio}${msg}${fin}"
516
517 }
```

Y hasta acá sería lo que logre avanzar del comando.



The image shows a screenshot of a text editor window with two tabs: 'llamadas.sh' and 'funcion.sh'. The editor displays shell script code for three functions: 'colAqua()', 'colMorado()', and 'colVerde()'. Each function sets an 'inicio' (start) and 'fin' (end) ANSI escape sequence, and uses 'echo -e' to print a message with those colors. The 'msg' variable is assigned the value '@:-' in each function. The code is as follows:

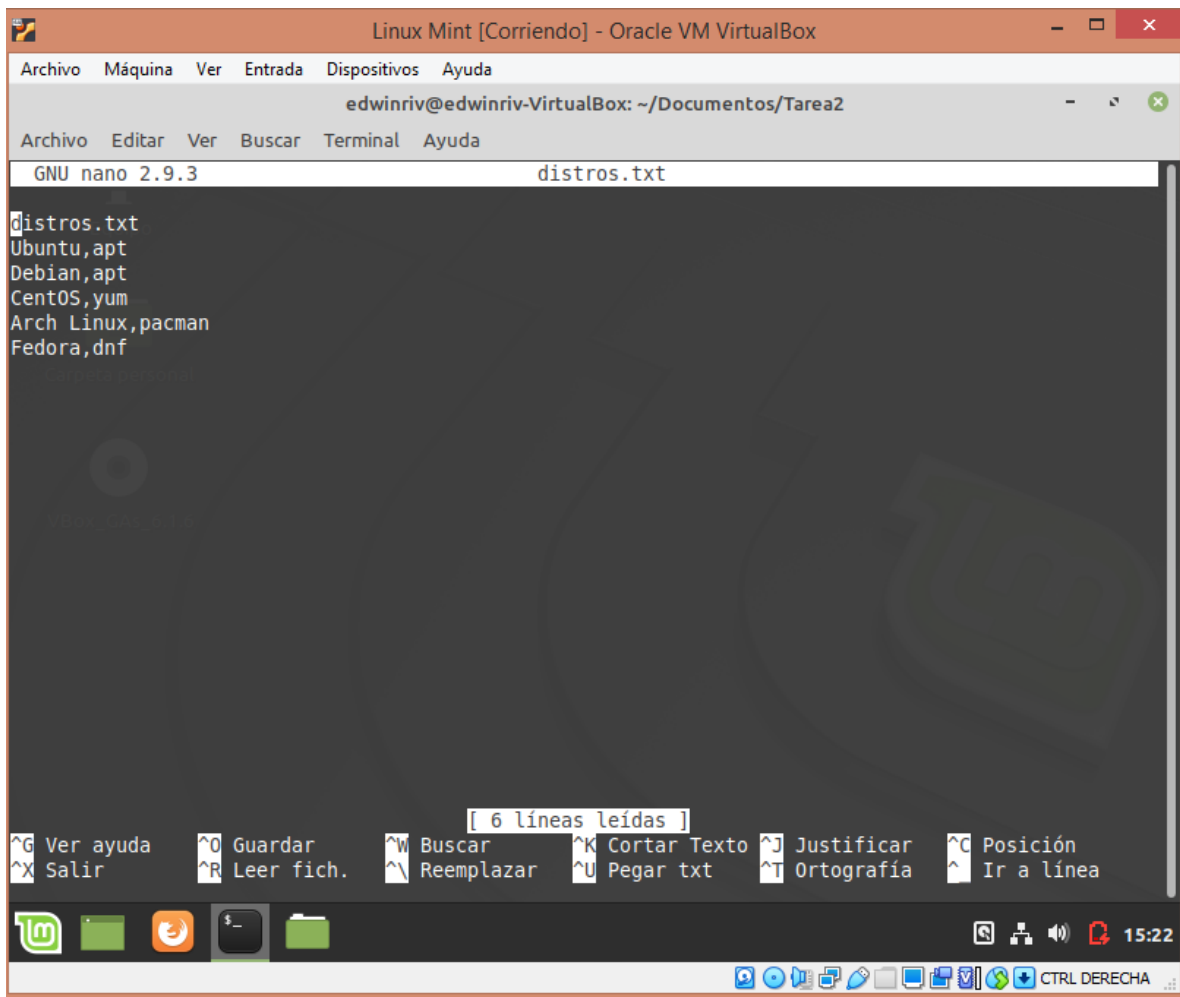
```
510
511     fin='\033[00;00m'
512
513     msg=@:-
514
515     echo -e "${inicio}${msg}${fin}"
516
517 }
518
519
520
521 function colAqua() {
522
523     inicio='\033[01;36m'
524
525     fin='\033[00;00m'
526
527     msg=@:-
528
529     echo -e "${inicio}${msg}${fin}"
530
531 }
532
533
534
535 function colMorado() {
536
537     inicio='\033[01;35m'
538
539     fin='\033[00;00m'
540
541     msg=@:-
542
543     echo -e "${inicio}${msg}${fin}"
544
545 }
```



4) Crear un script que recorra todas las líneas de un archivo y las muestre y cada una le anteponga "Línea:" el nombre del archivo debe pasarse como argumento al script.

### Solución:

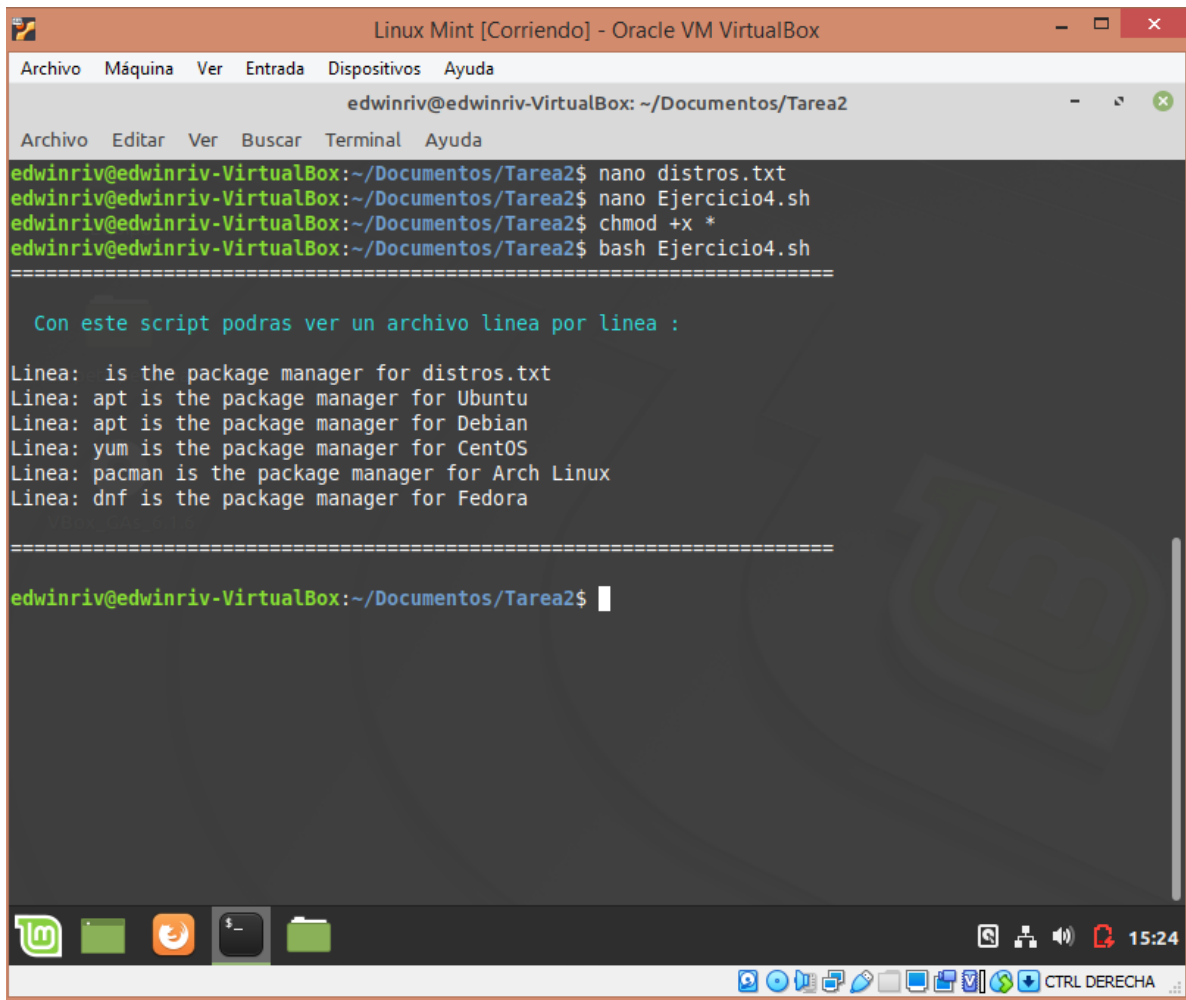
Para comenzar será necesario crear este archivo de texto y tenerlo donde creemos el Script.



```
Linux Mint [Corriendo] - Oracle VM VirtualBox
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2
GNU nano 2.9.3 distros.txt
distros.txt
Ubuntu,apt
Debian,apt
CentOS,yum
Arch Linux,pacman
Fedora,dnf
[ 6 líneas leídas ]
^G Ver ayuda  ^O Guardar  ^W Buscar  ^K Cortar Texto  ^J Justificar  ^C Posición
^X Salir      ^R Leer fich.  ^_ Reemplazar  ^U Pegar txt  ^T Ortografía  ^_ Ir a línea
```

Después de crear el archivo de texto, crearemos el script, daremos permisos y lo ejecutaremos.

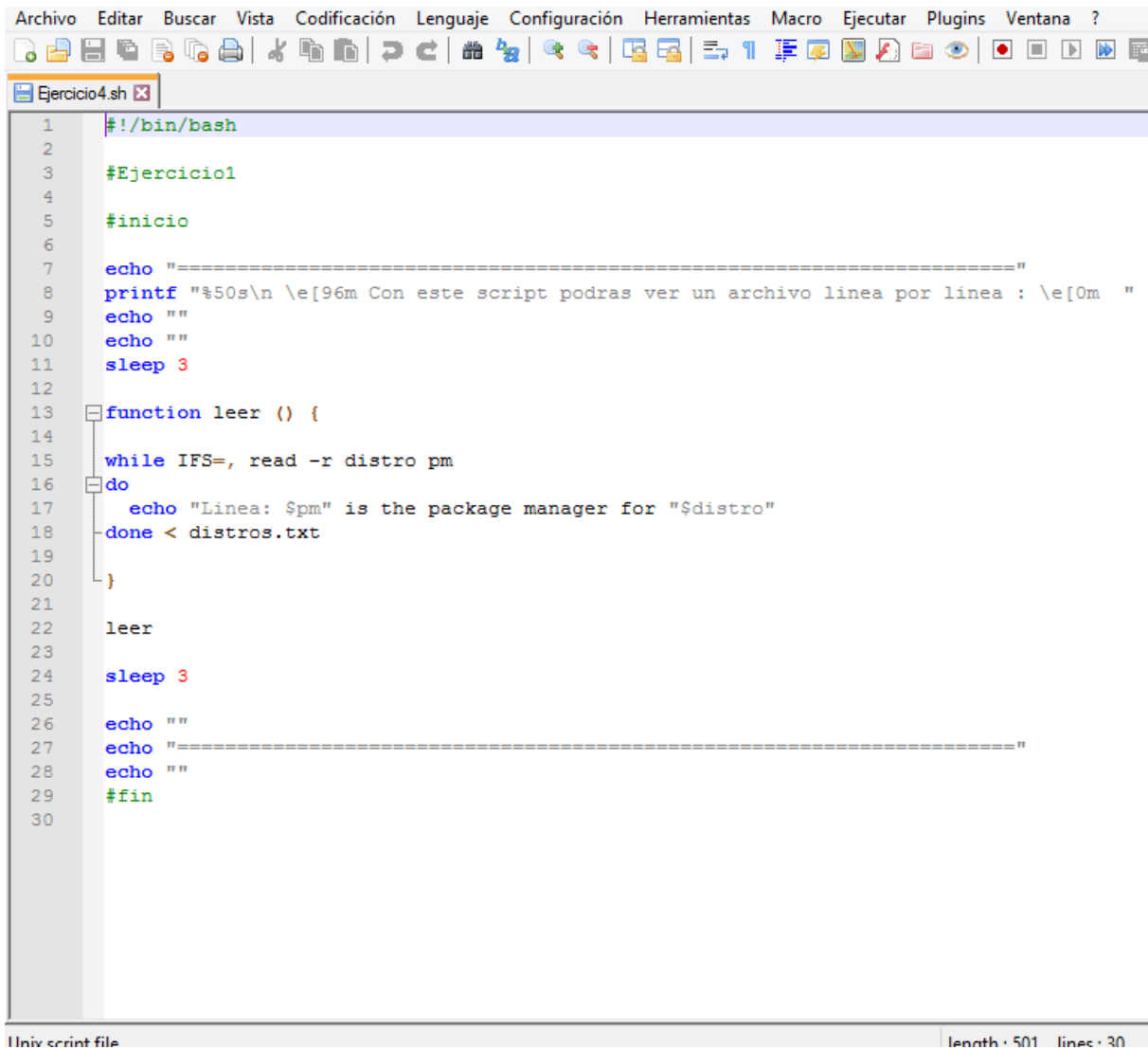
Al ejecutar el Script se mostrara el mensaje de inicio y después se ejecutara el comando para enlistar lo que creamos en el archivo de texto.



```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ nano distros.txt
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ nano Ejercicio4.sh
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ chmod +x *
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ bash Ejercicio4.sh
=====
Con este script podras ver un archivo linea por linea :

Linea: is the package manager for distros.txt
Linea: apt is the package manager for Ubuntu
Linea: apt is the package manager for Debian
Linea: yum is the package manager for CentOS
Linea: pacman is the package manager for Arch Linux
Linea: dnf is the package manager for Fedora
=====
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$
```

El código sería el siguiente. Printf para mostrar el mensaje de inicio centrado. Cree la función con el nombre leer y adentro de la función agregue el ciclo while do y done. Y un echo para mostrar cada línea escrita del txt y en done agregue el nombre del archivo txt.



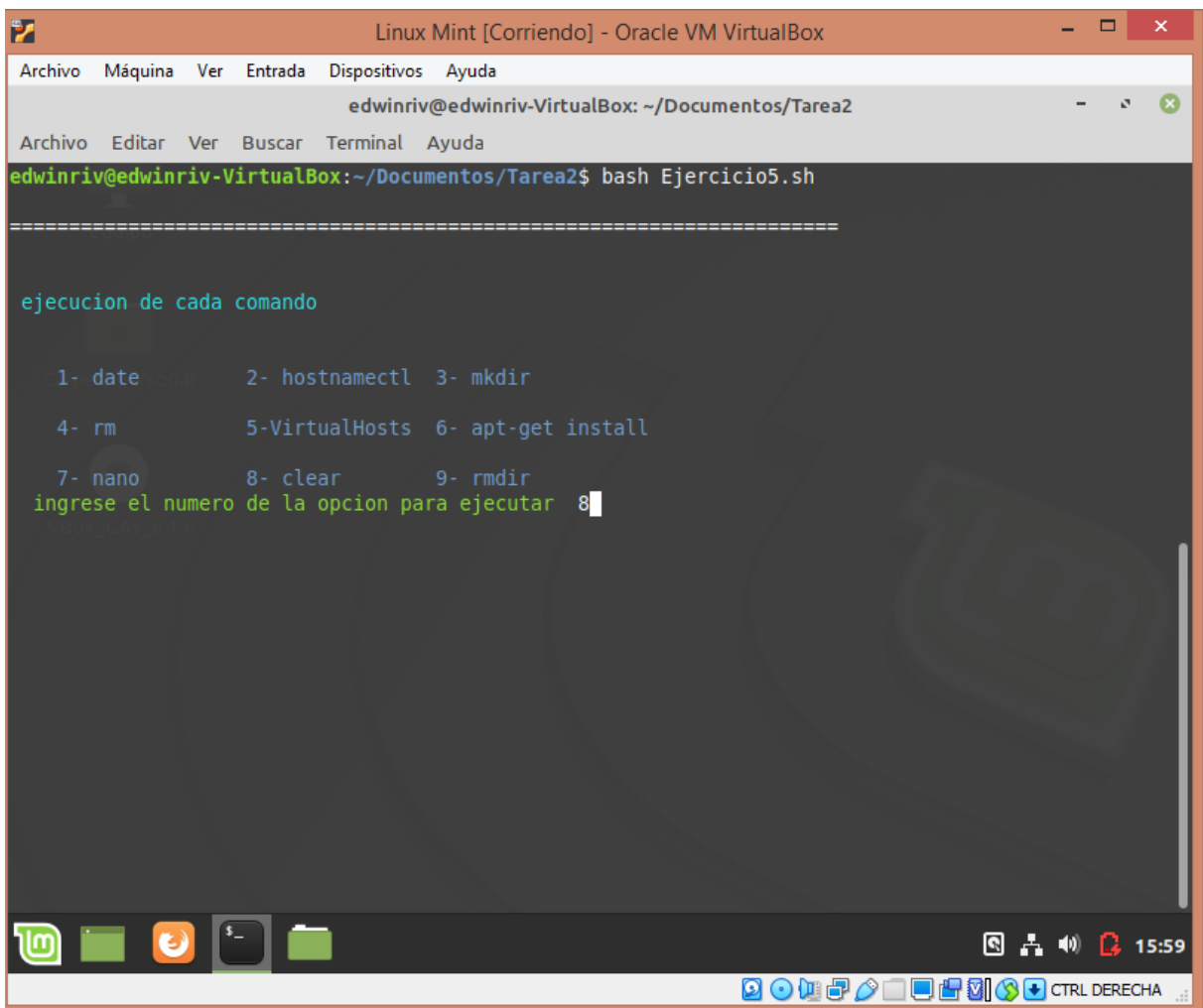
```
1  #!/bin/bash
2
3  #Ejercicio1
4
5  #inicio
6
7  echo "=====
8  printf "%50s\n \e[96m Con este script podras ver un archivo linea por linea : \e[0m "
9  echo ""
10 echo ""
11 sleep 3
12
13 function leer () {
14
15     while IFS=, read -r distro pm
16     do
17         echo "Linea: $pm" is the package manager for "$distro"
18     done < distros.txt
19
20 }
21
22 leer
23
24 sleep 3
25
26 echo ""
27 echo "=====
28 echo ""
29 #fin
30
```

Unix script file | length: 501 | lines: 30

5) Crear un script que al ejecutarse muestre un título “Ejecución de cada comando” una lista de opciones y que según el número se le ingrese muestre la ejecución de cada comando visto durante todo el ciclo (uno a uno), además entre esas opciones debe existir una que al seleccionarla debe crear tres virtualhost, pero para crear estos virtualhost desde teclado debe pedir solo el nombre de dominio y el nombre de la carpeta que será creada en /var/www/.

### Solución:

Después de crearlo y darle permisos lo ejecutamos y al inicio nos saldrá un menú de inicio. En mi caso elegí la opción 8

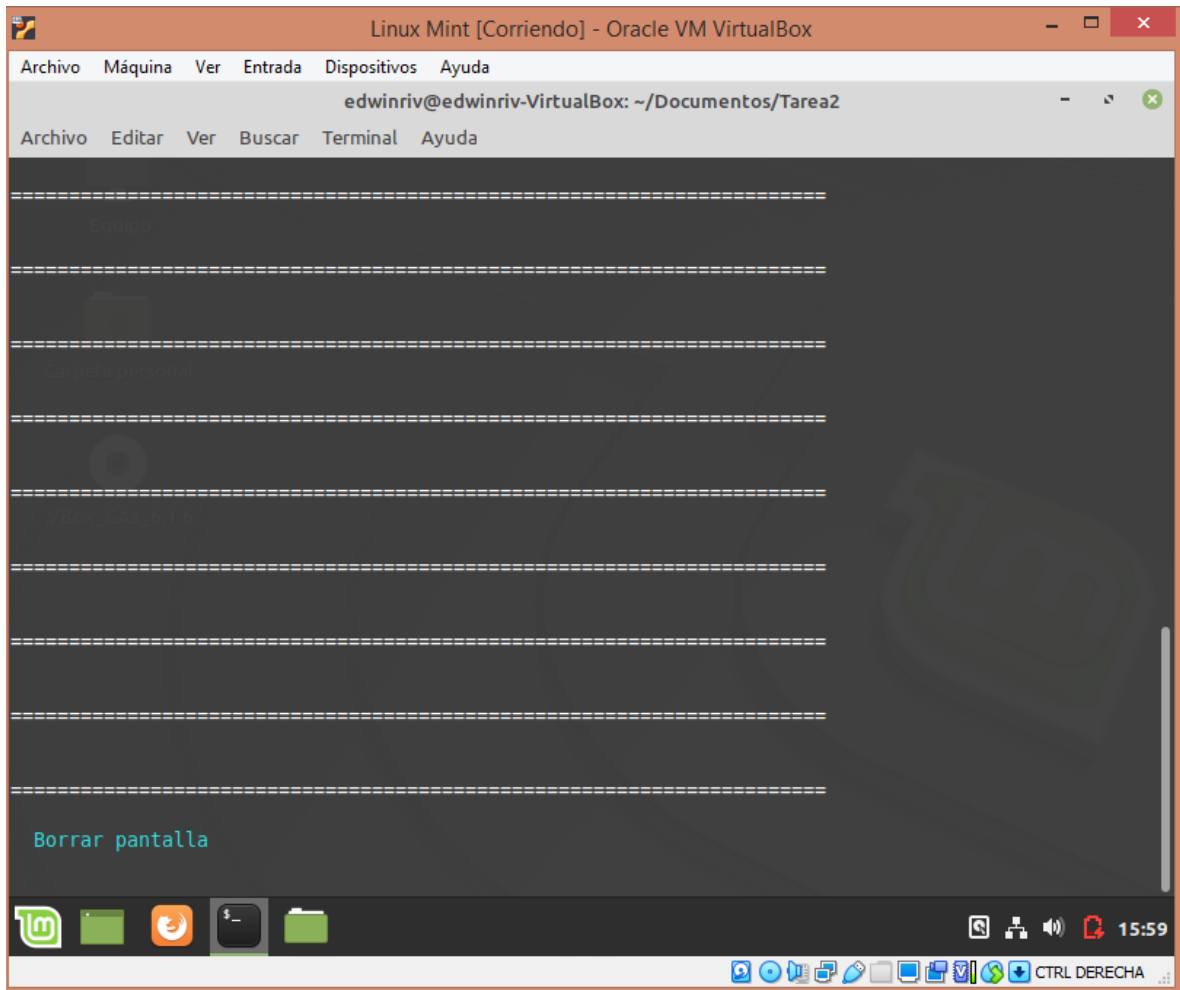


```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ bash Ejercicio5.sh
=====

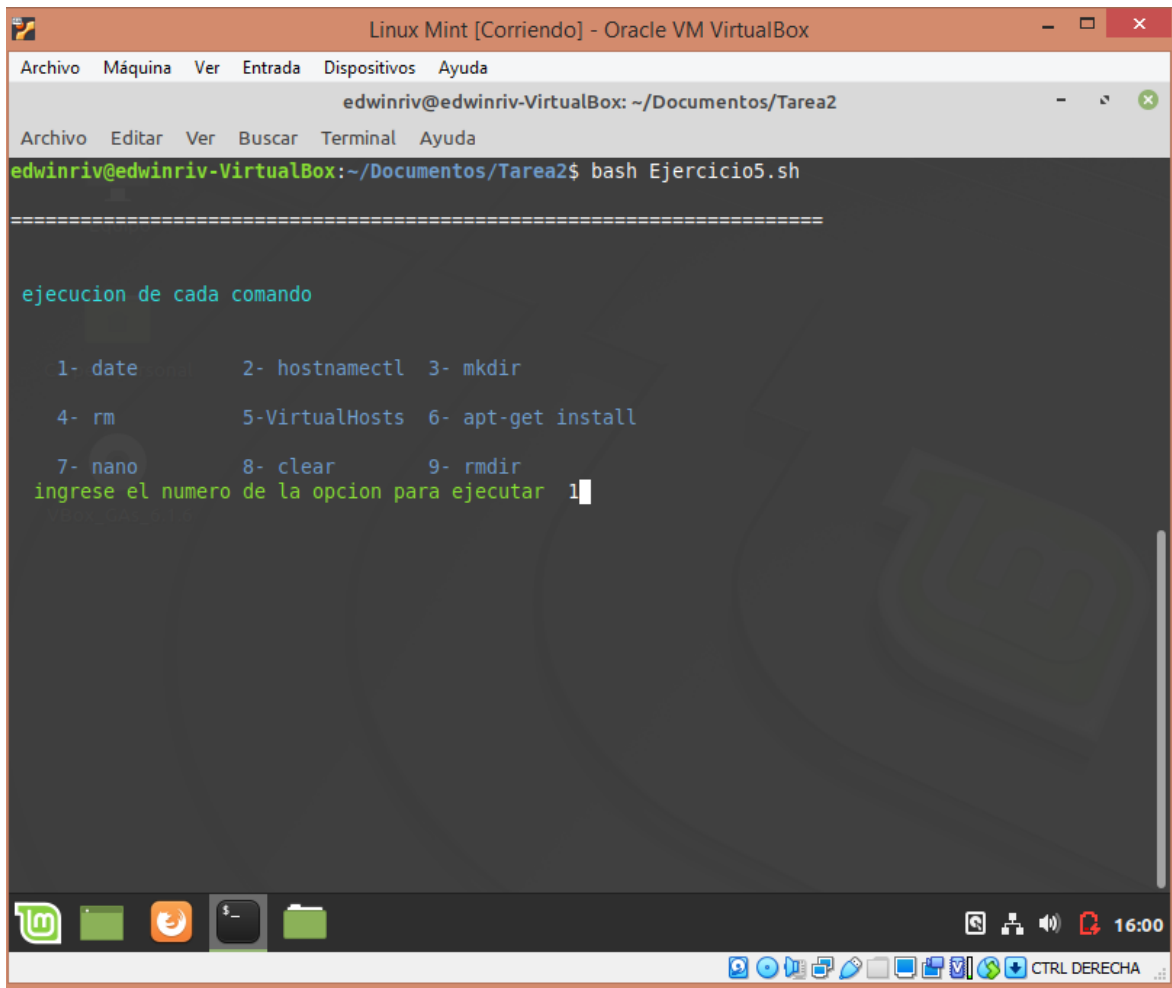
ejecucion de cada comando

1- date          2- hostnamectl  3- mkdir
4- rm            5-VirtualHosts 6- apt-get install
7- nano         8- clear       9- rmdir
ingrese el numero de la opcion para ejecutar 8
```

Y así se ve cuando ejecuta la opción ocho.



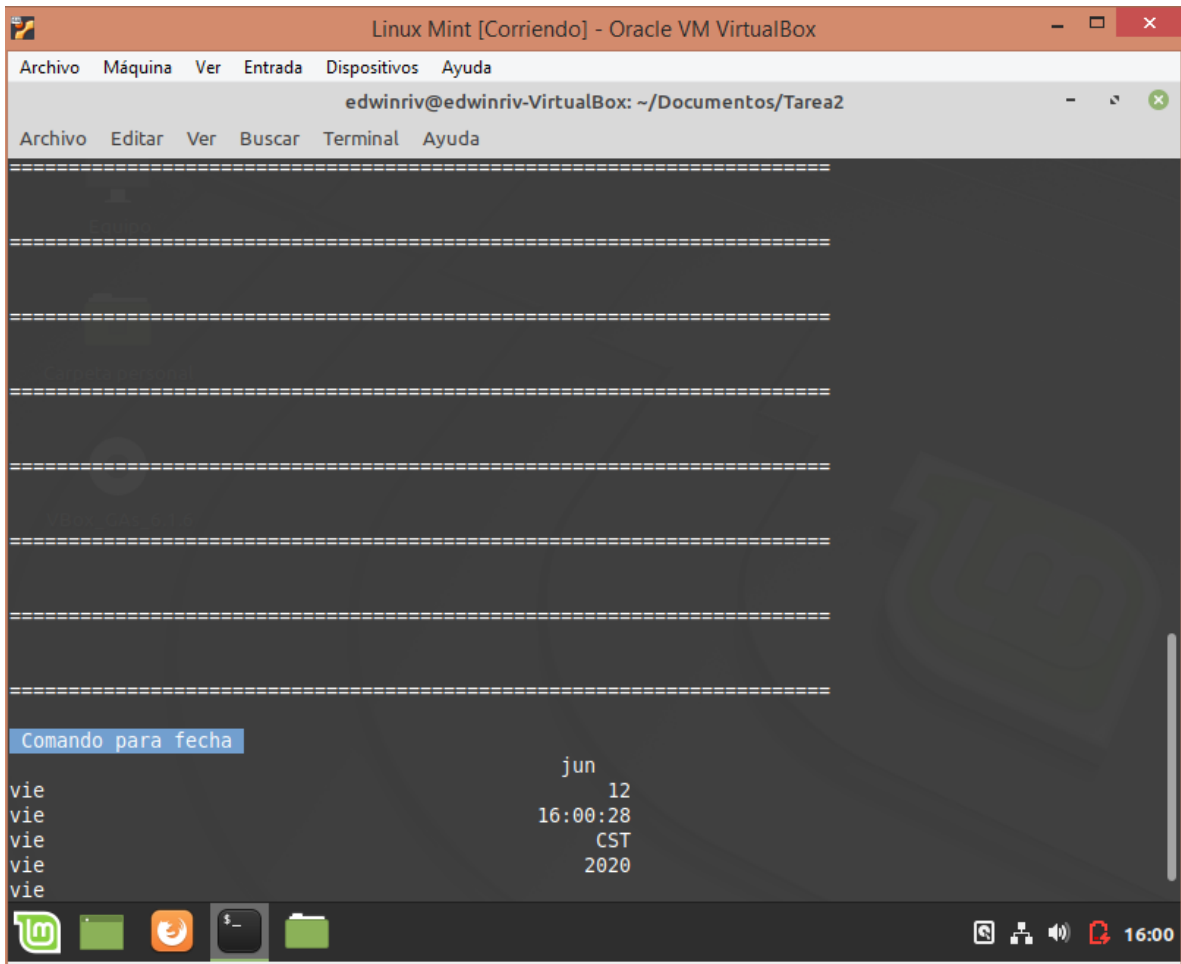
Ahora ejecutare la opción uno .



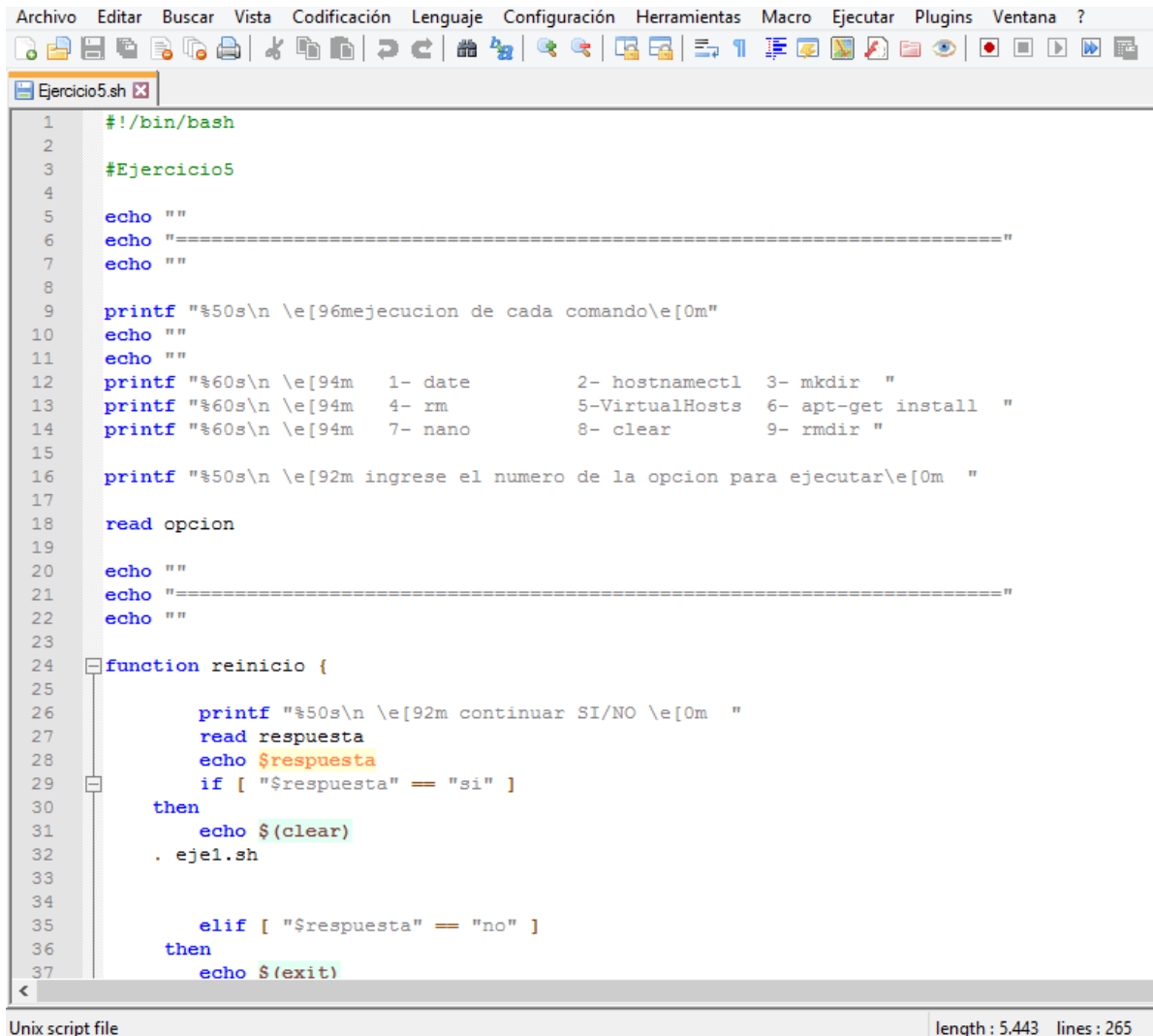
```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ bash Ejercicio5.sh
=====
ejecucion de cada comando

1- date          2- hostnamectl  3- mkdir
4- rm            5-VirtualHosts 6- apt-get install
7- nano         8- clear       9- rmdir
ingrese el numero de la opcion para ejecutar 1
```

Me mostrara la fecha de ejecucion



Usaremos printf para crear el menú, read para leer la opción, y crearemos una función llamada reinicio.



```
1  #!/bin/bash
2
3  #Ejercicio5
4
5  echo ""
6  echo "=====
7  echo ""
8
9  printf "%50s\n \e[96mejecucion de cada comando\e[0m"
10 echo ""
11 echo ""
12 printf "%60s\n \e[94m  1- date          2- hostnamectl  3- mkdir  "
13 printf "%60s\n \e[94m  4- rm          5-VirtualHosts  6- apt-get install  "
14 printf "%60s\n \e[94m  7- nano          8- clear          9- rmdir  "
15
16 printf "%50s\n \e[92m ingrese el numero de la opcion para ejecutar\e[0m  "
17
18 read opcion
19
20 echo ""
21 echo "=====
22 echo ""
23
24 function reinicio {
25
26     printf "%50s\n \e[92m continuar SI/NO \e[0m  "
27     read respuesta
28     echo $respuesta
29     if [ "$respuesta" == "si" ]
30 then
31     echo $(clear)
32     . eje1.sh
33
34
35     elif [ "$respuesta" == "no" ]
36 then
37     echo $(exit)
```

Unix script file | length : 5.443 lines : 265



Cerraremos la función eh iremos creando nuevas funciones.

```
37     echo $(exit)
38
39
40     else
41         echo "\e[91mError de opcion!!!\e[0m"
42     echo $(exit)
43
44     fi
45 }
46
47 echo ""
48 echo "===== "
49 echo ""
50
51     function uno
52 {
53     printf "%50s\n"$(date)
54 }
55
56 echo ""
57 echo "===== "
58 echo ""
59
60     function dos
61 {
62     printf "%80s/n $(hostnamectl) "
63 }
64
65 echo ""
66 echo "===== "
67 echo ""
68
69     function tres
70
71 {
72     echo -e "\e[96mingrese nombre de carpeta\e[0m  "
73     read carpeta
```

Unix script file length : 5.443

Creación de más funciones.

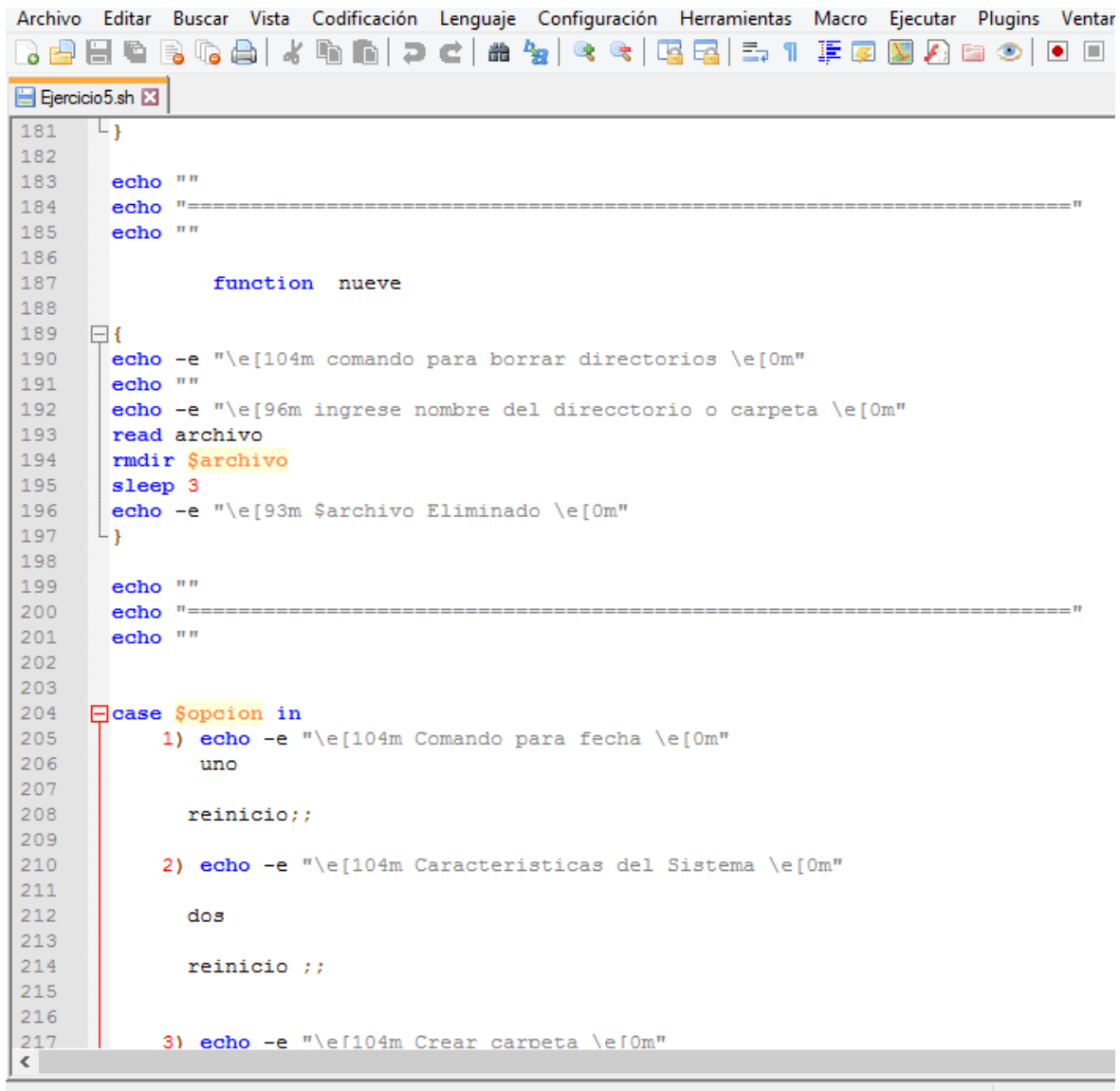
```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana
Ejercicio5.sh x
73     read carpeta
74     $(mkdir $carpeta)
75     echo -e "\e[93mCarpeta Creada\e[0m"
76 }
77
78 echo ""
79 echo "===== "
80 echo ""
81
82     function cuatro
83 {
84     echo -e "\e[96mIngrese el nombre del elemento\e[0m  "
85     read borrar
86     $(rm $borrar)
87     sleep 3
88     echo -e "\e[93m $borrar Eliminado \e[0m"
89 }
90
91 echo ""
92 echo "===== "
93 echo ""
94
95     function cinco
96 {
97     #!/bin/bash
98     #pidiendon datosde entrada
99     printf "%50s\n \e[96m ingrese el nombre de la carpeta\e[0m  "
100    read carpeta
101    printf "%50s\n \e[96m ingrese el nombre del dominio\e[0m  "
102    read dom
103    dominio=$dom".local"
104    arphp=$dom".php"
105    cd /var/www
106    mkdir $carpeta
107    cd /var/www/$carpeta
108    echo -e "\e[1m \e[93m Espere....\e[0m"
109    sleep 3
110 }
111
```

Los comandos necesarios para virtualhost.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ven
Ejercicio5.sh x
109  sleep 3
110  #creando archivo php
111
112  sudo cat $arphp
113  echo '<!DOCTYPE html> <html lang="es"> <meta charset="utf-8"> <head> <title>Bi
114
115  #ingresando dominio a etc/hosts
116
117  echo "127.0.0.1      "$dominio >> /etc/hosts
118  cd /etc/apache2/sites-available
119  #creando archivo.conf
120  sudo cat $dominio.conf
121  #ingresando configuracion al archivo .conf
122
123  echo "<VirtualHost *:80>" >> $dominio.conf
124  echo "ServerAdmin webmaster@localhost" >> $dominio.conf
125  echo "DocumentRoot /var/www/"$carpeta >> $dominio.conf
126  echo "ServerName "$dominio >> $dominio.conf
127
128  echo "ErrorLog ${APACHE_LOG_DIR}/error.log" >> $dominio.conf
129  echo "CustomLog ${APACHE_LOG_DIR}/access.log combined" >> $dominio.conf
130  echo "</VirtualHost>" >> $dominio.conf
131  #accediendo a lacarpeta sites-available para dar de alta al dominio
132
133  cd /etc/apache2/sites-available
134  sudo a2ensite $dominio.conf
135  sudo systemctl reload apache2
136  clear
137  echo -e "\e[93mServidor $dom Creado \e[0m"
138
139  }
140
141  echo ""
142  echo "=====
143  echo ""
144
145  function seis
<
Unix script file | length : 5.4
```



Más funciones y creación de case.



```
181 }
182
183 echo ""
184 echo "-----"
185 echo ""
186
187     function nuevo
188
189 {
190     echo -e "\e[104m comando para borrar directorios \e[0m"
191     echo ""
192     echo -e "\e[96m ingrese nombre del directorio o carpeta \e[0m"
193     read archivo
194     rmdir $archivo
195     sleep 3
196     echo -e "\e[93m $archivo Eliminado \e[0m"
197 }
198
199 echo ""
200 echo "-----"
201 echo ""
202
203
204 case $opcion in
205     1) echo -e "\e[104m Comando para fecha \e[0m"
206         uno
207
208         reinicio;;
209
210     2) echo -e "\e[104m Características del Sistema \e[0m"
211
212         dos
213
214         reinicio ;;
215
216
217     3) echo -e "\e[104m Crear carpeta \e[0m"
```

Comandos necesarios.

```
217 3) echo -e "\[104m Crear carpeta \[0m"
218
219 tres
220
221 reinicio;;
222
223 4) echo -e "\[104m Comando para borrar un archivo \[0m"
224 cuatro
225
226 reinicio ;;
227
228 5) echo -e "\[104m Comando para crear Virtual Hosts \[0m"
229
230 cinco
231 echo ""
232 echo -e "\[34m para el siguiente dominio \[0m"
233 cinco
234 echo ""
235 echo -e "\[34m para el siguiente dominio \[0m"
236 cinco
237
238 reinicio
239 ;;
240
241
242 6) echo -e "\[104m Comando para instalar paquetes \[0m"
243
244 seis
245 reinicio
246 ;;
247
248 7) siete
249 reinicio
250 ;;
251
252 8) ocho
253 reinicio
<
```

Finalización de case.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro
Ejercicio5.sh x
230     cinco
231     echo ""
232     echo -e "\e[34m para el siguiente dominio \e[0m"
233     cinco
234     echo ""
235     echo -e "\e[34m para el siguiente dominio \e[0m"
236     cinco
237
238     reinicio
239     ;;
240
241
242     6) echo -e "\e[104m Comando para instalar paquetes \e[0m"
243
244     seis
245     reinicio
246     ;;
247
248     7) siete
249     reinicio
250     ;;
251
252     8) ocho
253     reinicio
254     ;;
255
256     9) nueve
257     reinicio
258
259     ;;
260 esac
261
262 echo ""
263 echo "=====
264 echo ""
265
```

6) Crear un script que permita dar permisos a un archivo o carpeta (debe evaluarse si es un archivo o carpeta), al ejecutarse debe permitir crear un archivo o carpeta y seleccionar el permiso que se le quiere dar, al final debe mostrar los archivos o carpetas que se crearon, la ruta, los permisos aplicados y la fecha de creación de estos.

### Solución:

Después de las acciones básicas, al ejecutarlo se verá así. Un mini menú de dos opciones. Yo selecciono la opción uno. Luego me pidió el nombre y lo escribí.

```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2
Archivo Editar Ver Buscar Terminal Ayuda
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ nano Ejercicio6.sh
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ chmod 777 Ejercicio6.sh
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ bash Ejercicio6.sh

=====

seleccione una opcion

1: Crear carpeta
2: Crear archivo
2

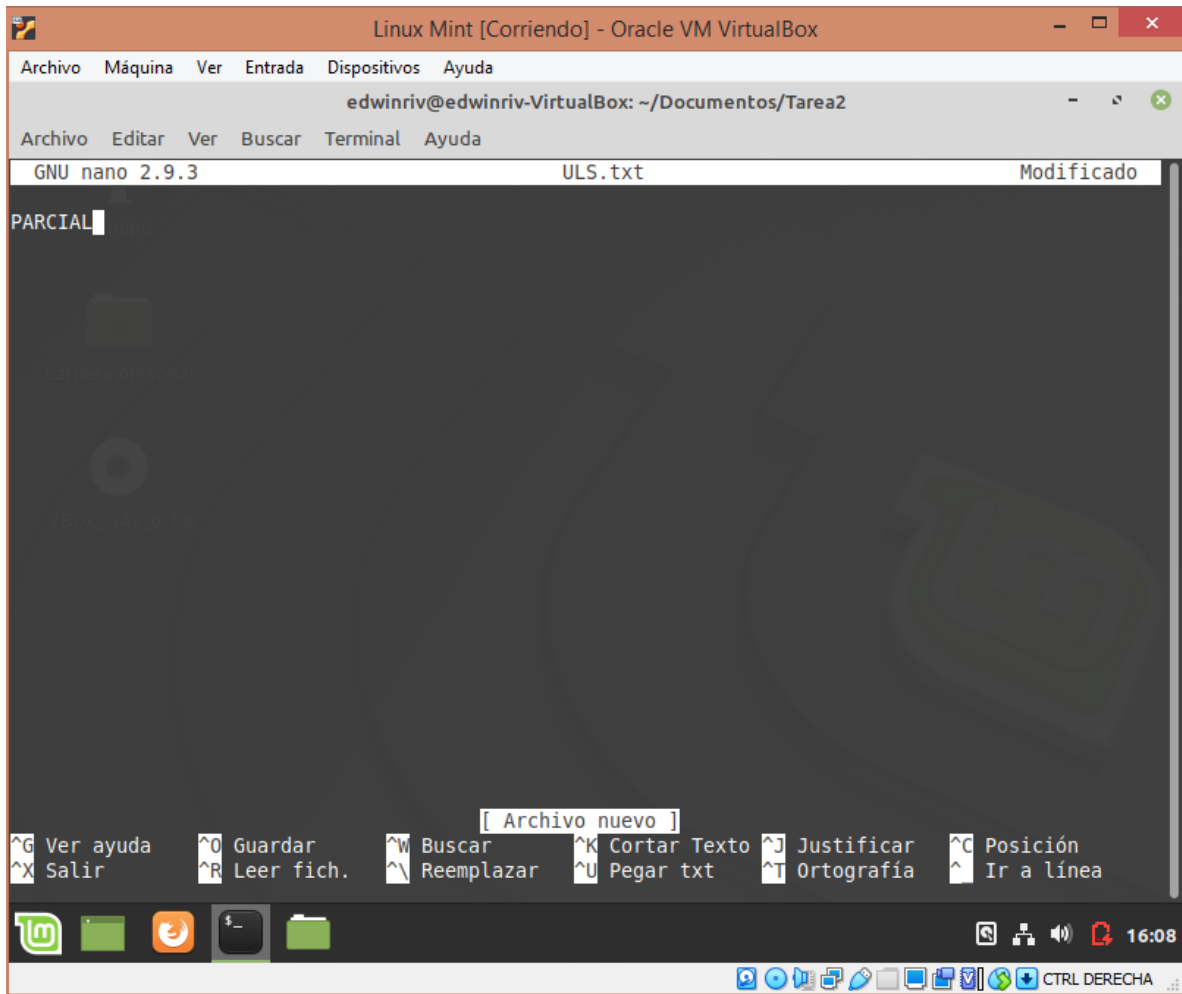
Ingrese el nombre ULS.txt

=====
=====
=====
=====
=====
=====
=====
=====
=====
=====
=====

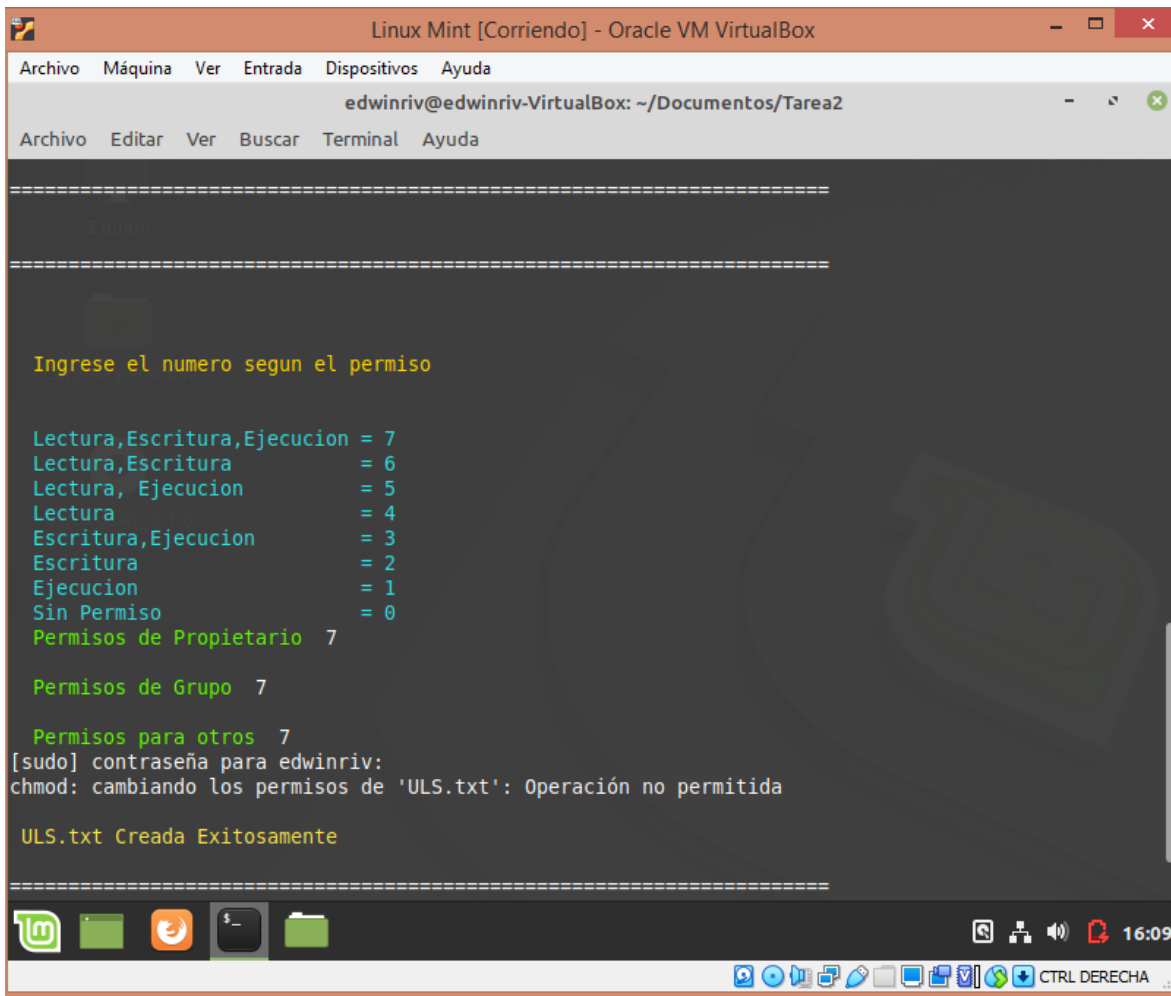
=====
```



Se abrirá nano y podremos digitar lo que sea necesario.

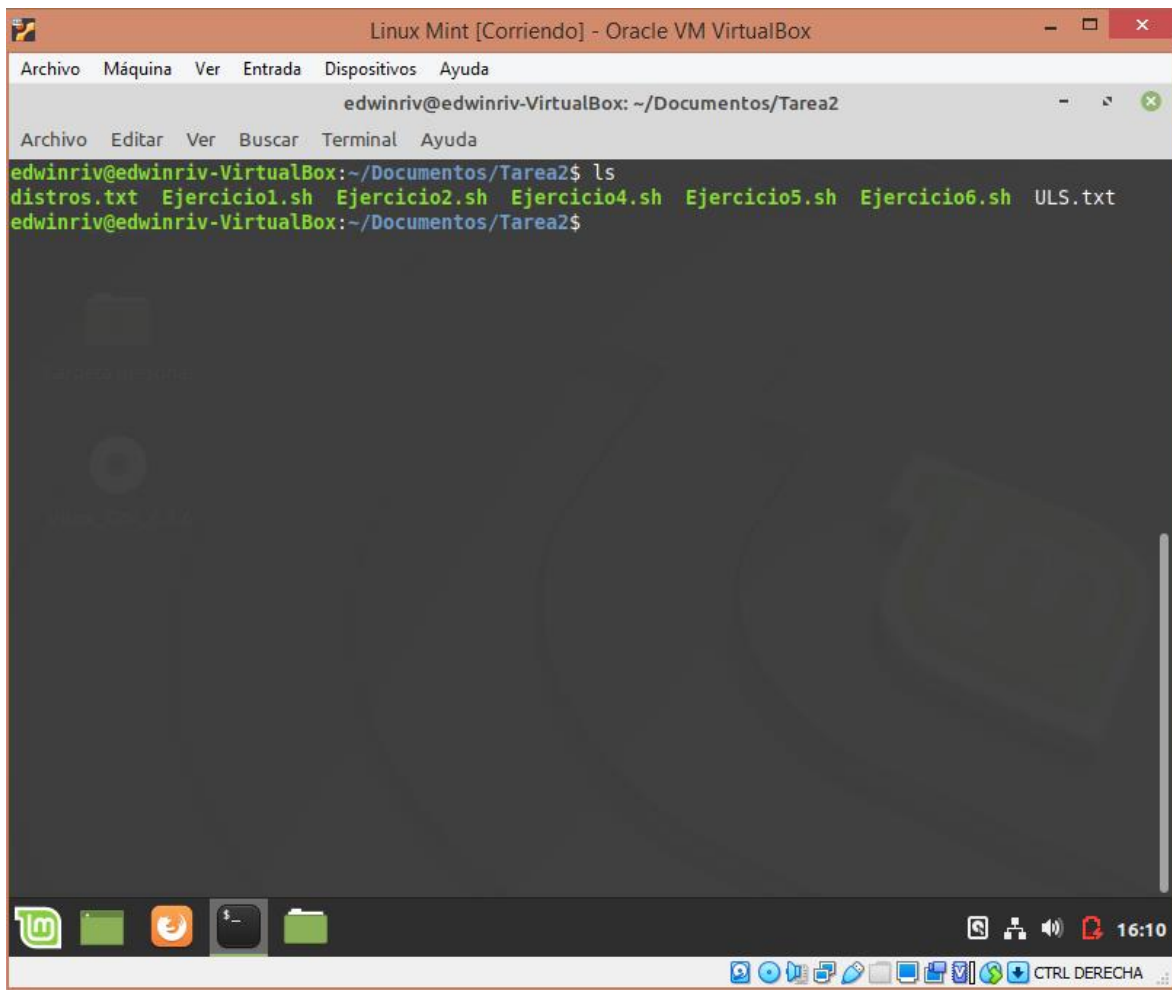


Después de guardarlo, nos pedirá lo siguiente, en mi caso elegí siete en las tres opciones. Al final nos dará el mensaje de que se creó exitosamente.



```
=====  
=====  
Ingrese el numero segun el permiso  
  
Lectura,Escritura,Ejecucion = 7  
Lectura,Escritura           = 6  
Lectura, Ejecucion          = 5  
Lectura                     = 4  
Escritura,Ejecucion         = 3  
Escritura                   = 2  
Ejecucion                   = 1  
Sin Permiso                 = 0  
Permisos de Propietario 7  
  
Permisos de Grupo 7  
  
Permisos para otros 7  
[sudo] contraseña para edwinriv:  
chmod: cambiando los permisos de 'ULS.txt': Operación no permitida  
  
ULS.txt Creada Exitosamente  
=====
```

Enlistamos y acá está el archivo que creamos.

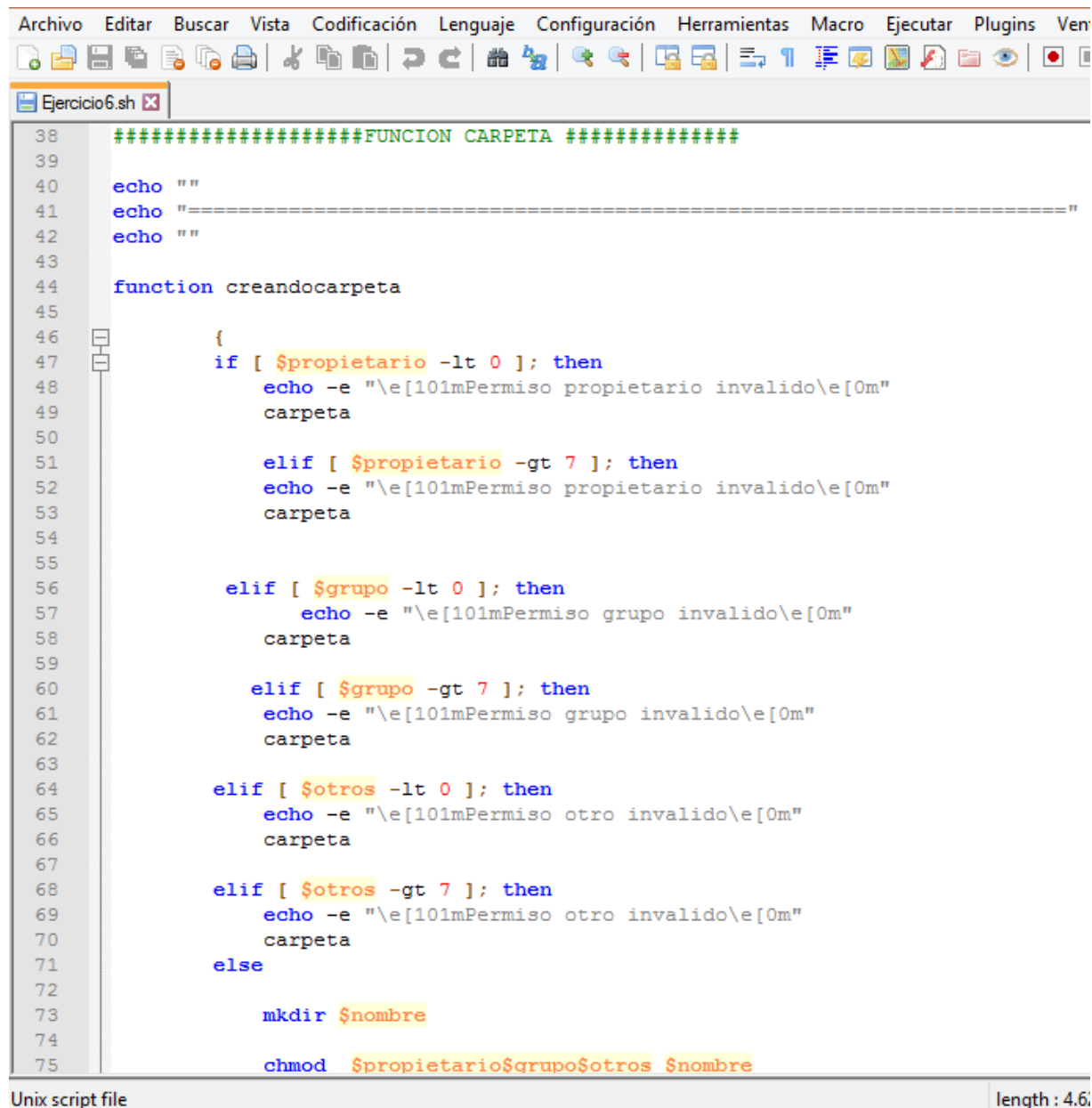


```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ ls
distros.txt  Ejercicio1.sh  Ejercicio2.sh  Ejercicio4.sh  Ejercicio5.sh  Ejercicio6.sh  ULS.txt
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$
```

Este es su código. Usamos printf para centrar todo. Creamos una función llamada permisos.

```
Archivo  Editar  Buscar  Vista  Codificacion  Lenguaje  Configuracion  Herramientas  Macro  Ejecutar  Plugins  Venta
Ejercicio6.sh x
1  #!/bin/bash
2
3  #Ejercicio6
4
5  echo ""
6  echo "-----"
7  echo ""
8
9  printf "%50s\n \e[220m seleccione una opcion\e[0m  "
10 echo ""
11 printf "%50s\n \e[96m1: Crear carpeta\e[0m  "
12 printf "%50s\n \e[96m2: Crear archivo\e[0m  "
13 echo ""
14 read op
15 echo ""
16 printf "%50s\n \e[38;5;220m Ingrese el nombre\e[0m  "
17
18 read nombre
19
20 echo ""
21 echo "-----"
22 echo ""
23
24 function permisos
25
26     { echo ""
27         printf "%50s\n \e[96m Lectura,Escritura,Ejecucion = 7\e[0m  "
28         printf "%50s\n \e[96m Lectura,Escritura           = 6\e[0m  "
29         printf "%50s\n \e[96m Lectura, Ejecucion           = 5\e[0m  "
30         printf "%50s\n \e[96m Lectura                     = 4\e[0m  "
31         printf "%50s\n \e[96m Escritura,Ejecucion         = 3\e[0m  "
32         printf "%50s\n \e[96m Escritura                   = 2\e[0m  "
33         printf "%50s\n \e[96m Ejecucion                   = 1\e[0m  "
34         printf "%50s\n \e[96m Sin Permiso                 = 0\e[0m  "
35
36     }
37
38 #####FUNCION CARPETA #####
Unix script file | length : 4.62:
```

Otra función, llamada creando carpeta, agregamos variables y usamos condicionales.



```
38 #####FUNCION CARPETA #####
39
40 echo ""
41 echo "-----"
42 echo ""
43
44 function creandocarpeta
45
46 {
47     if [ $propietario -lt 0 ]; then
48         echo -e "\e[101mPermiso propietario invalido\e[0m"
49         carpeta
50
51         elif [ $propietario -gt 7 ]; then
52         echo -e "\e[101mPermiso propietario invalido\e[0m"
53         carpeta
54
55
56         elif [ $grupo -lt 0 ]; then
57             echo -e "\e[101mPermiso grupo invalido\e[0m"
58             carpeta
59
60             elif [ $grupo -gt 7 ]; then
61                 echo -e "\e[101mPermiso grupo invalido\e[0m"
62                 carpeta
63
64         elif [ $otros -lt 0 ]; then
65             echo -e "\e[101mPermiso otro invalido\e[0m"
66             carpeta
67
68             elif [ $otros -gt 7 ]; then
69                 echo -e "\e[101mPermiso otro invalido\e[0m"
70                 carpeta
71         else
72
73             mkdir $nombre
74
75             chmod $propietario$grupo$otros $nombre
```

Unix script file length : 4.6

Cerramos condicional y creamos una nueva función llamada carpeta.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana
Ejercicio6.sh x
75         chmod $propietario$grupo$otros $nombre
76         echo ""
77         echo -e "\e[93m $nombre Creada Exitosamente\e[0m"
78
79
80
81     fi
82 }
83
84 echo ""
85 echo "-----"
86 echo ""
87
88 function carpeta
89 {
90     echo ""
91     printf "%50s\n \e[38;5;220m Ingrese el numero segun el permiso \e[0m  "
92     echo ""
93
94     permisos
95
96     printf "%50s\n \e[38;5;82m Permisos de Propietario\e[0m  "
97     read propietario
98
99     printf "%50s\n \e[38;5;82m Permisos de Grupo\e[0m  "
100    read grupo
101
102    printf "%50s\n \e[38;5;82m Permisos para otros\e[0m  "
103    read otros
104
105    creandocarpeta
106
107 }
108
109 echo ""
110 echo "-----"
111 echo ""
112
```

Una nueva función con condicionales y variables.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Vent
Ejercicio6.sh x
110  echo "=====
111  echo ""
112
113  ##### FUNCION PARA ARCHIVO #####
114  function creandoarchivo
115
116      {
117      if [ $propietario -lt 0 ]; then
118          echo -e "\e[101mPermiso propietario invalido\e[0m"
119          archivo
120
121          elif [ $propietario -gt 7 ]; then
122          echo -e "\e[101mPermiso propietario invalido\e[0m"
123          archivo
124
125
126          elif [ $grupo -lt 0 ]; then
127          echo -e "\e[101mPermiso grupo invalido\e[0m"
128          archivo
129
130          elif [ $grupo -gt 7 ]; then
131          echo -e "\e[101mPermiso grupo invalido\e[0m"
132          archivo
133
134          elif [ $otros -lt 0 ]; then
135          echo -e "\e[101mPermiso otro invalido\e[0m"
136          archivo
137
138          elif [ $otros -gt 7 ]; then
139          echo -e "\e[101mPermiso otro invalido\e[0m"
140          archivo
141      else
142
143          sudo nano $nombre
144
145          chmod $propietario$grupo$otros $nombre
146          echo ""
147          echo -e "\e[93m $nombre Creada Exitosamente\e[0m"
```

Unix script file length : 4.62

Crear función llamada archivo. Iniciar case.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana
Ejercicio6.sh x
147         echo -e "\e[93m $nombre Creada Exitosamente\e[0m"
148
149
150
151     fi
152 }
153
154 echo ""
155 echo "-----"
156 echo ""
157
158 function archivo
159 {
160     echo ""
161     printf "%50s\n \e[38;5;220m Ingrese el numero segun el permiso \e[0m  "
162     echo ""
163
164     permisos
165
166     printf "%50s\n \e[38;5;82m Permisos de Propietario\e[0m  "
167     read propietario
168
169     printf "%50s\n \e[38;5;82m Permisos de Grupo\e[0m  "
170     read grupo
171
172     printf "%50s\n \e[38;5;82m Permisos para otros\e[0m  "
173     read otros
174
175     creandoarchivo
176
177 }
178
179
180 case $op in
181     1) carpeta ;;
182     2) archivo;;
183
184 
```

Unix script file | length : 4.623



Cerrar case y ese sería el fin del código.

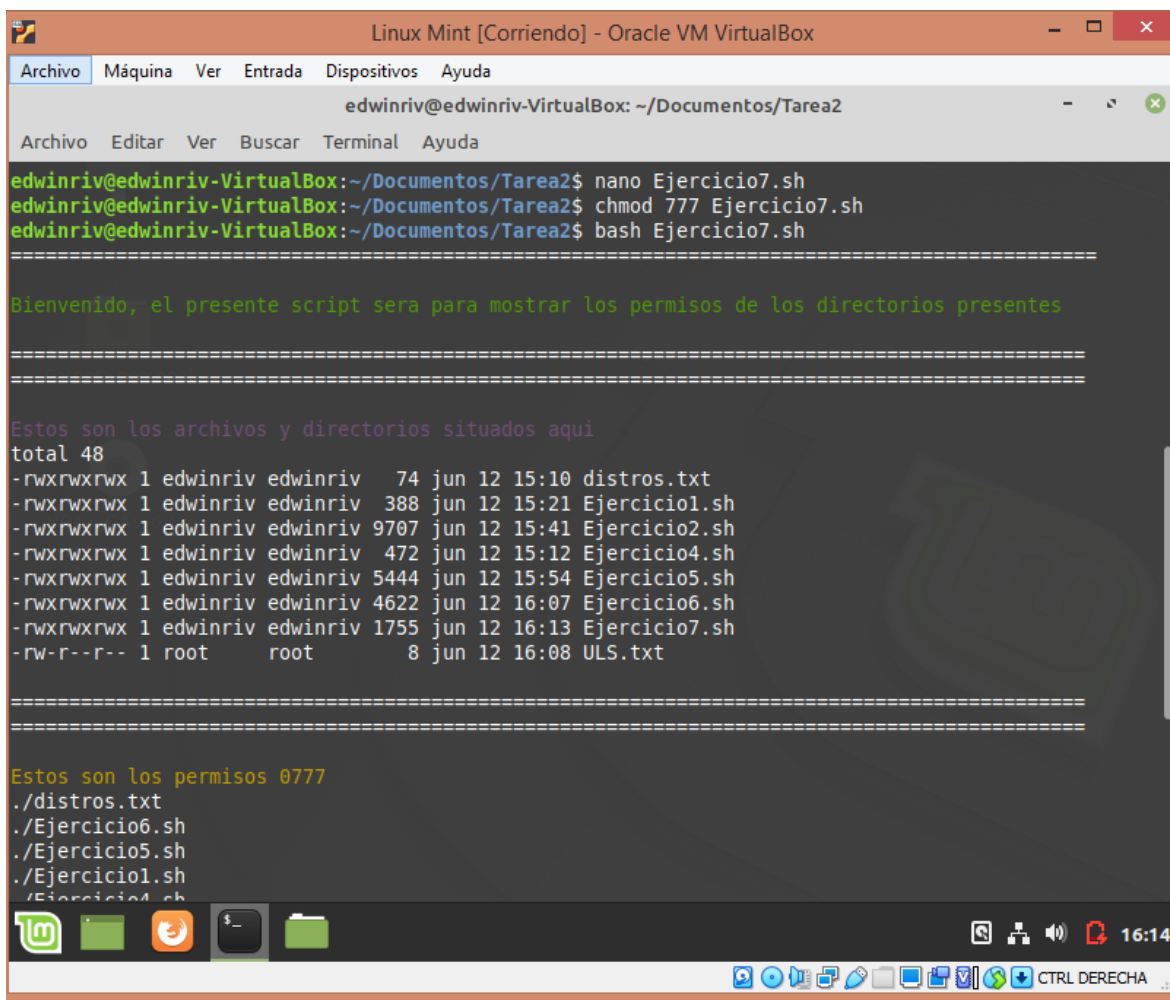
```
158 function archivo
159 {
160     echo ""
161     printf "%50s\n \e[38;5;220m Ingrese el numero segun el permiso \e[0m "
162     echo ""
163
164     permisos
165
166     printf "%50s\n \e[38;5;82m Permisos de Propietario\e[0m "
167     read propietario
168
169     printf "%50s\n \e[38;5;82m Permisos de Grupo\e[0m "
170     read grupo
171
172     printf "%50s\n \e[38;5;82m Permisos para otros\e[0m "
173     read otros
174
175     creandoarchivo
176
177 }
178
179
180 case $op in
181     1) carpeta ;;
182     2) archivo;;
183
184
185
186
187
188 esac
189
190
191 echo ""
192 echo "-----"
193 echo ""
194
```

Unix script file | lenath : 4.623

7) Crear un script que al ejecutarse permita saber de una lista de archivos y carpetas cuál o cuáles tienen permisos de 0777, debe mostrar el nombre o nombres de los archivos con los permisos y en qué ruta están con fecha de creación.

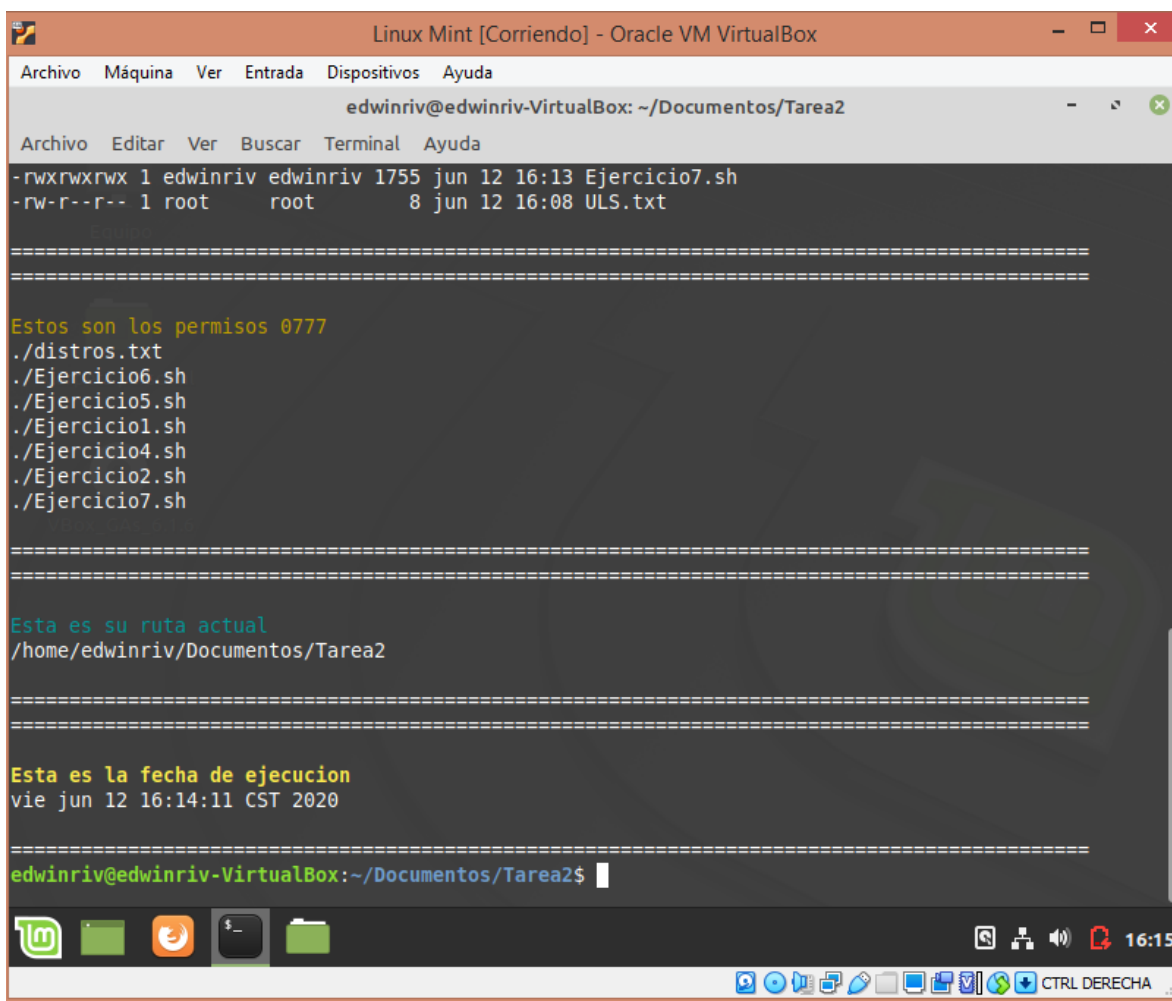
### Solución:

Después de las acciones básicas lo ejecutamos y se verá así. Un enunciado de inicio y luego se mostrarán los archivos y directorios situados ahí, con sus permisos, nombre, fecha, etc.



```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2
Archivo Editar Ver Buscar Terminal Ayuda
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ nano Ejercicio7.sh
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ chmod 777 Ejercicio7.sh
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$ bash Ejercicio7.sh
=====
Bienvenido, el presente script sera para mostrar los permisos de los directorios presentes
=====
Estos son los archivos y directorios situados aqui
total 48
-rwxrwxrwx 1 edwinriv edwinriv 74 jun 12 15:10 distros.txt
-rwxrwxrwx 1 edwinriv edwinriv 388 jun 12 15:21 Ejercicio1.sh
-rwxrwxrwx 1 edwinriv edwinriv 9707 jun 12 15:41 Ejercicio2.sh
-rwxrwxrwx 1 edwinriv edwinriv 472 jun 12 15:12 Ejercicio4.sh
-rwxrwxrwx 1 edwinriv edwinriv 5444 jun 12 15:54 Ejercicio5.sh
-rwxrwxrwx 1 edwinriv edwinriv 4622 jun 12 16:07 Ejercicio6.sh
-rwxrwxrwx 1 edwinriv edwinriv 1755 jun 12 16:13 Ejercicio7.sh
-rw-r--r-- 1 root root 8 jun 12 16:08 ULS.txt
=====
Estos son los permisos 0777
./distros.txt
./Ejercicio6.sh
./Ejercicio5.sh
./Ejercicio1.sh
./Ejercicio4.sh
```

Se mostraran los archivos que tengan permisos 777, en mi caso a todos los archivos les eh dado ese permiso y por eso me aparecen todos, luego nos muestra nuestra ruta actual y fecha de ejecución del Script.



```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
edwinriv@edwinriv-VirtualBox: ~/Documentos/Tarea2
Archivo Editar Ver Buscar Terminal Ayuda
-rwxrwxrwx 1 edwinriv edwinriv 1755 jun 12 16:13 Ejercicio7.sh
-rw-r--r-- 1 root      root      8 jun 12 16:08 ULS.txt

=====

Estos son los permisos 0777
./distros.txt
./Ejercicio6.sh
./Ejercicio5.sh
./Ejercicio1.sh
./Ejercicio4.sh
./Ejercicio2.sh
./Ejercicio7.sh

=====

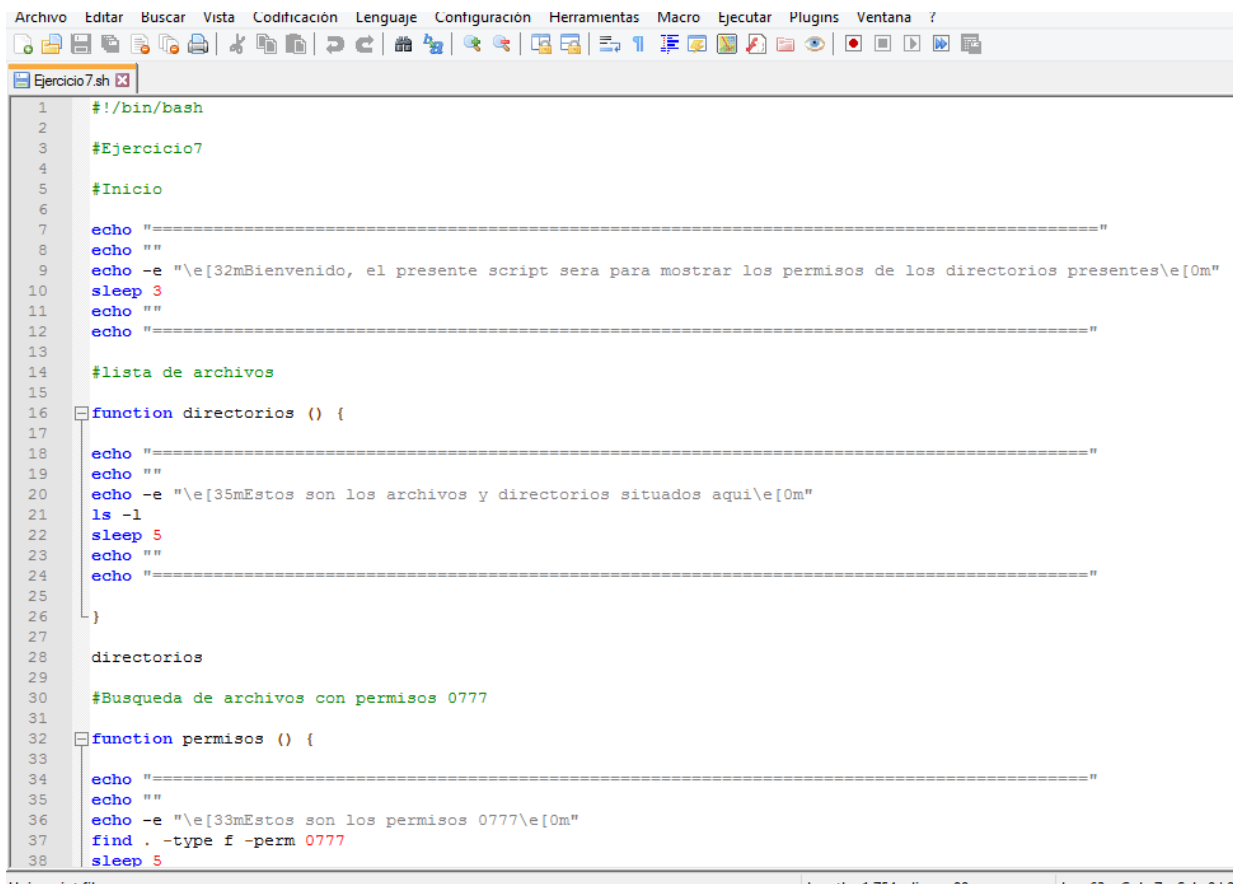
Esta es su ruta actual
/home/edwinriv/Documentos/Tarea2

=====

Esta es la fecha de ejecucion
vie jun 12 16:14:11 CST 2020

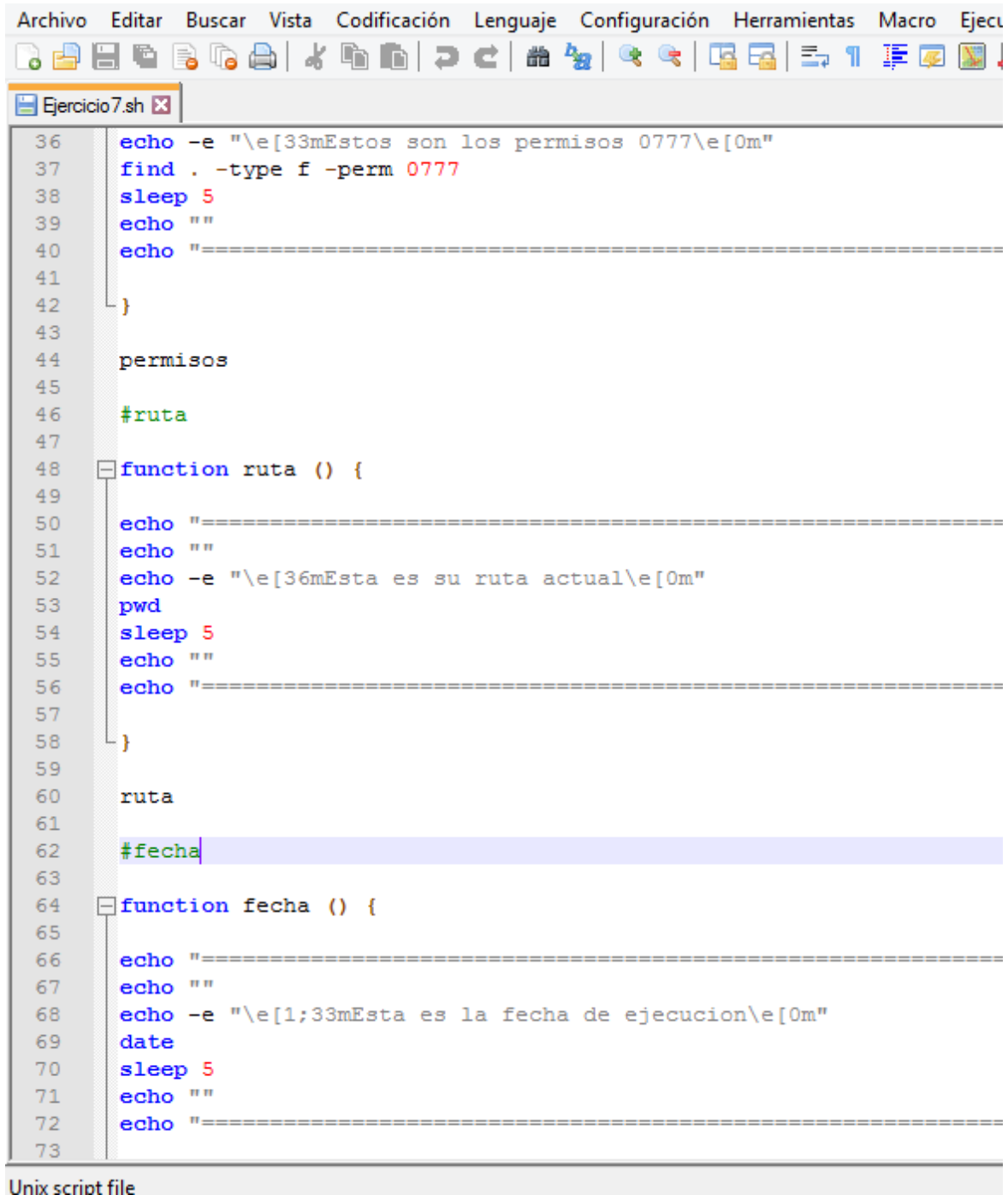
=====
edwinriv@edwinriv-VirtualBox:~/Documentos/Tarea2$
```

Después del mensaje de inicio crearemos una función llamada directorios, la cual con el comando digitado nos dará la información necesaria. Luego crearemos otra función llamada permisos, la cual servirá para buscar archivos con dichos permisos.



```
1  #!/bin/bash
2
3  #Ejercicio7
4
5  #Inicio
6
7  echo "=====
8  echo ""
9  echo -e "\e[32mBienvenido, el presente script sera para mostrar los permisos de los directorios presentes\e[0m"
10 sleep 3
11 echo ""
12 echo "=====
13
14 #lista de archivos
15
16 function directorios () {
17
18 echo "=====
19 echo ""
20 echo -e "\e[35mEstos son los archivos y directorios situados aqui\e[0m"
21 ls -l
22 sleep 5
23 echo ""
24 echo "=====
25
26 }
27
28 directorios
29
30 #Busqueda de archivos con permisos 0777
31
32 function permisos () {
33
34 echo "=====
35 echo ""
36 echo -e "\e[33mEstos son los permisos 0777\e[0m"
37 find . -type f -perm 0777
38 sleep 5
```

Iniciamos una función llamada ruta la cual servirá para ver la ruta actual de ejecución del Script. Una función para los datos de fecha.



```
36 echo -e "\e[33mEstos son los permisos 0777\e[0m"
37 find . -type f -perm 0777
38 sleep 5
39 echo ""
40 echo "=====
41
42 }
43
44 permisos
45
46 #ruta
47
48 function ruta () {
49
50 echo "=====
51 echo ""
52 echo -e "\e[36mEsta es su ruta actual\e[0m"
53 pwd
54 sleep 5
55 echo ""
56 echo "=====
57
58 }
59
60 ruta
61
62 #fecha
63
64 function fecha () {
65
66 echo "=====
67 echo ""
68 echo -e "\e[1;33mEsta es la fecha de ejecucion\e[0m"
69 date
70 sleep 5
71 echo ""
72 echo "=====
73
```

Unix script file

Y eso sería el código completo.



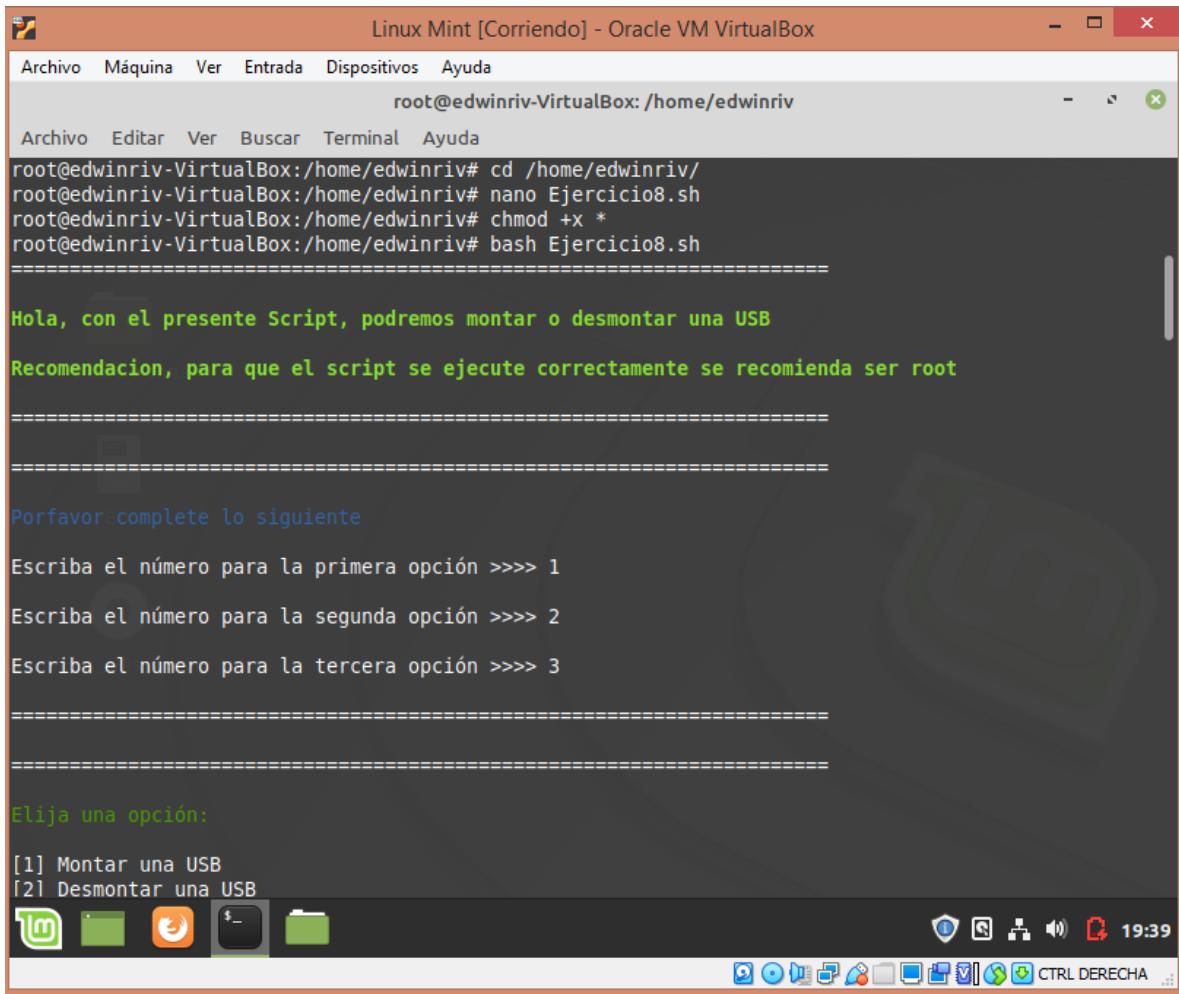
```
44 permisos
45
46 #ruta
47
48 function ruta () {
49
50     echo "=====
51     echo ""
52     echo -e "\e[36mEsta es su ruta actual\e[0m"
53     pwd
54     sleep 5
55     echo ""
56     echo "=====
57
58 }
59
60 ruta
61
62 #fecha
63
64 function fecha () {
65
66     echo "=====
67     echo ""
68     echo -e "\e[1;33mEsta es la fecha de ejecucion\e[0m"
69     date
70     sleep 5
71     echo ""
72     echo "=====
73
74 }
75
76 fecha
77
78 #fin
79
80
```

Unix script file

8) Crear un script que permita montar y desmontar un dispositivo de almacenamiento (USB o Disco Duro). Se debe mostrar cuál es la carpeta donde se monta el dispositivo y el dispositivo mismo con fecha de ejecución actual del script.

### Solución:

Accederemos a root antes de comenzar y después de seguir las acciones básicas, lo ejecutamos y se verá así. Un enunciado de inicio, pedirá indicaciones, hay que fijarse bien en las teclas dadas. Se mostrara un mini menú con tres opciones.



```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
root@edwinriv-VirtualBox: /home/edwinriv
Archivo Editar Ver Buscar Terminal Ayuda
root@edwinriv-VirtualBox:/home/edwinriv# cd /home/edwinriv/
root@edwinriv-VirtualBox:/home/edwinriv# nano Ejercicio8.sh
root@edwinriv-VirtualBox:/home/edwinriv# chmod +x *
root@edwinriv-VirtualBox:/home/edwinriv# bash Ejercicio8.sh
=====
Hola, con el presente Script, podremos montar o desmontar una USB
Recomendacion, para que el script se ejecute correctamente se recomienda ser root
=====
Porfavor complete lo siguiente
Escriba el número para la primera opción >>>> 1
Escriba el número para la segunda opción >>>> 2
Escriba el número para la tercera opción >>>> 3
=====
Elija una opción:
[1] Montar una USB
[2] Desmontar una USB
```

Comencé con la primera opción, hay que fijarnos bien y seguir las indicaciones.

```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
root@edwinriv-VirtualBox: /home/edwinriv
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
[2] Desmontar una USB
[3] Salir
>>>> 1
=====
Como primer paso sera el ver el sistema de archivo al que esta asignado el dispositivo

Disco /dev/sda: 25 GiB, 26843545600 bytes, 52428800 sectores
Unidades: sectores de 1 * 512 = 512 bytes
Tamaño de sector (lógico/físico): 512 bytes / 512 bytes
Tamaño de E/S (mínimo/óptimo): 512 bytes / 512 bytes
Tipo de etiqueta de disco: dos
Identificador del disco: 0xa5402647

Dispositivo Inicio Comienzo   Final Sectores Tamaño Id Tipo
/dev/sda1 *      2048 52426751 52424704   25G 83 Linux

Disco /dev/sdb: 14.4 GiB, 15472047104 bytes, 30218842 sectores
Unidades: sectores de 1 * 512 = 512 bytes
Tamaño de sector (lógico/físico): 512 bytes / 512 bytes
Tamaño de E/S (mínimo/óptimo): 512 bytes / 512 bytes
Tipo de etiqueta de disco: dos
Identificador del disco: 0x25465995

Dispositivo Inicio Comienzo   Final Sectores Tamaño Id Tipo
/dev/sdb1 *      63 30217823 30217761  14.4G b W95 FAT32
```



Es muy recomendable seguir las indicaciones dadas. Por último se enlistaran los archivos que contenga la USB en mi caso fueron dos carpetas.

```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
root@edwinriv-VirtualBox: /home/edwinriv
Archivo Editar Ver Buscar Terminal Ayuda
Dispositivo Inicio Comienzo Final Sectores Tamaño Id Tipo
/dev/sda1 * 2048 52426751 52424704 25G 83 Linux

Disco /dev/sdb: 14.4 GiB, 15472047104 bytes, 30218842 sectores
Unidades: sectores de 1 * 512 = 512 bytes
Tamaño de sector (lógico/físico): 512 bytes / 512 bytes
Tamaño de E/S (mínimo/óptimo): 512 bytes / 512 bytes
Tipo de etiqueta de disco: dos
Identificador del disco: 0x25465995

Dispositivo Inicio Comienzo Final Sectores Tamaño Id Tipo
/dev/sdb1 * 63 30217823 30217761 14.4G b W95 FAT32

Ahora que ya vimos los dispositivos de almacenamiento actualmente activos en el sistema, procedemos a montar la USB

Nota, la USB puede ser detectada como sdc1, sdb1, dependiendo de los dispositivos conectados, también se recomienda usar los directorios sugeridos

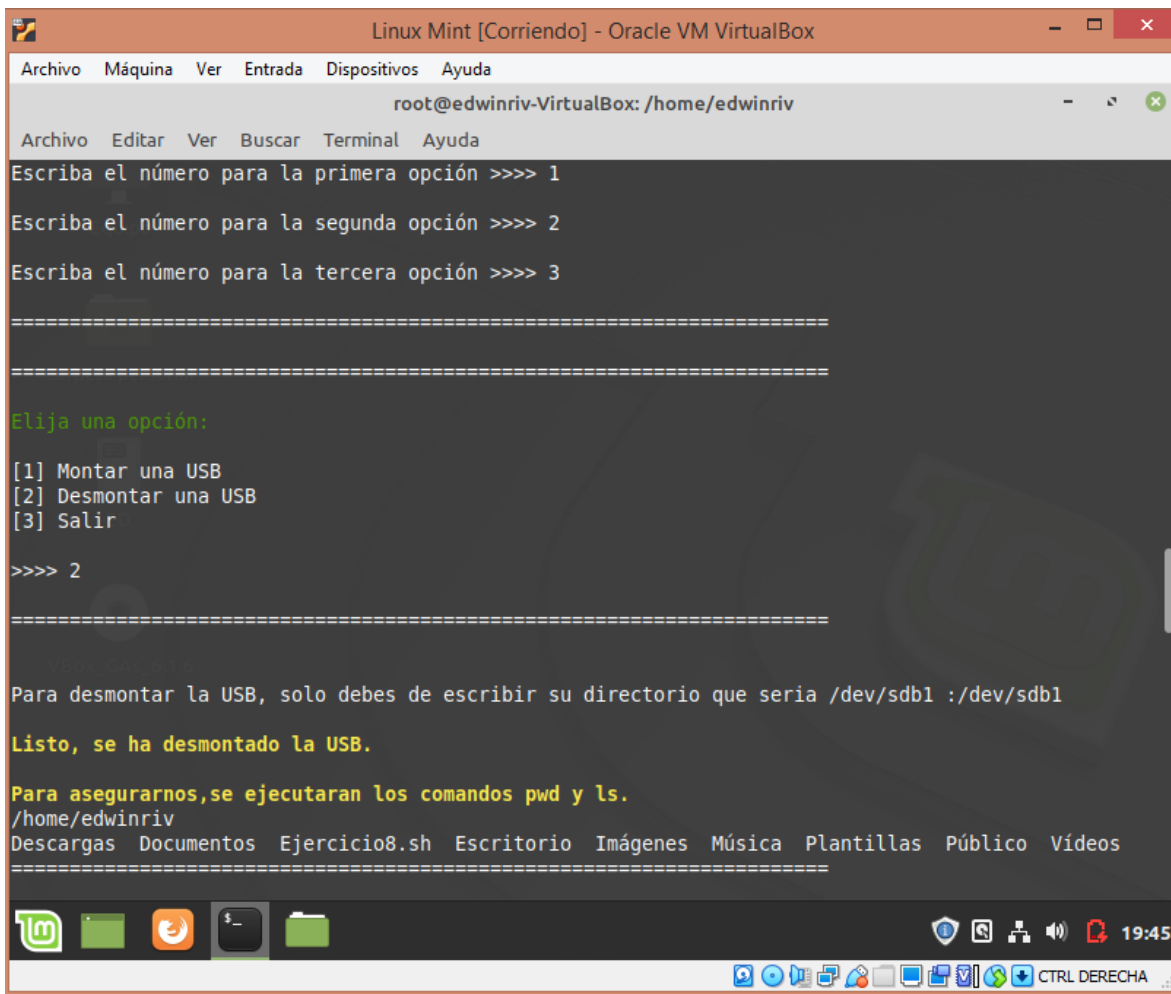
Porfavor escribir el siguiente directorio /root/USB/ : /root/USB/

Ahora digitaremos /dev/sdb1 /root/USB/ ./dev/sdb1 /root/USB/

Listo, aca esta la lista de los archivos de la USB.

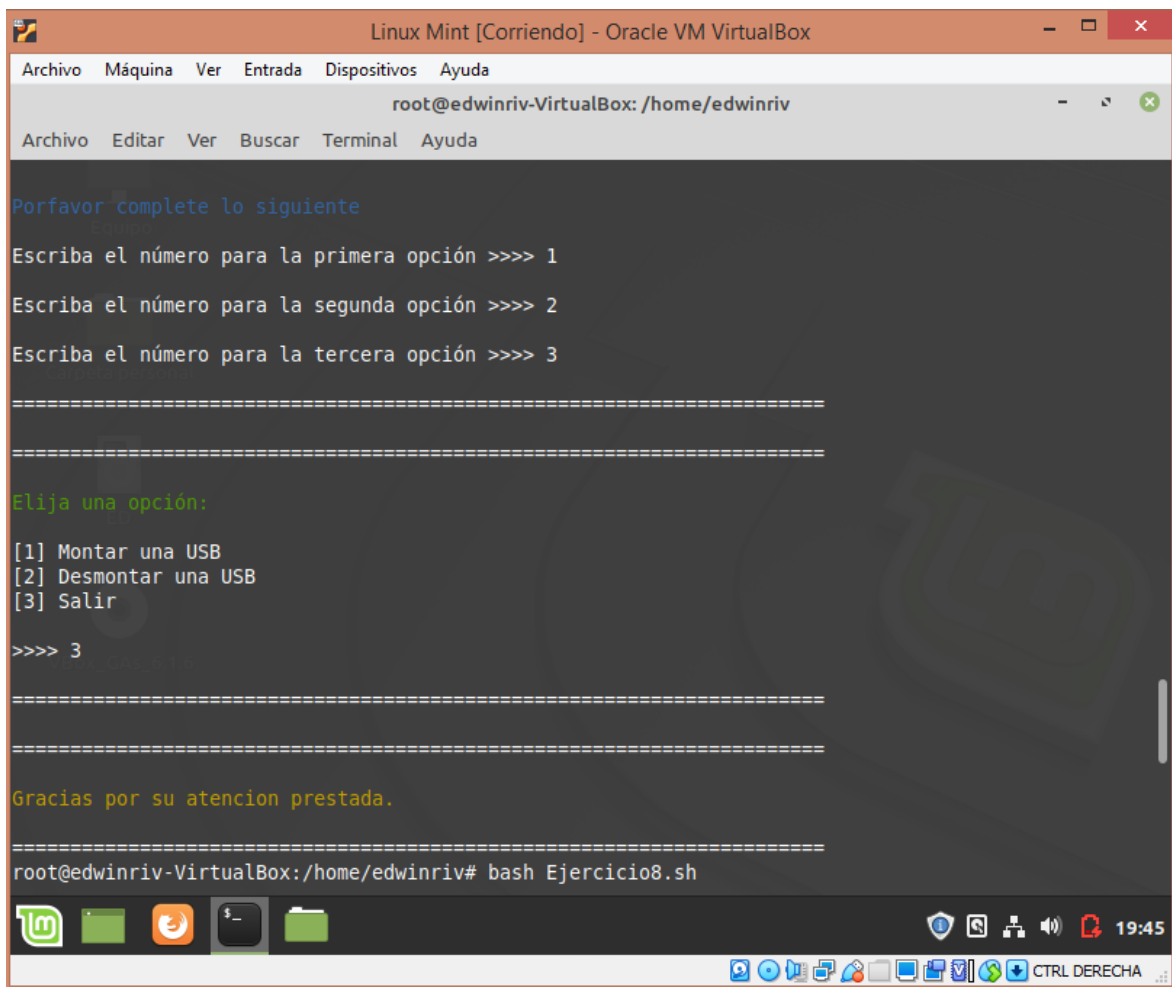
'System Volume Information' ULS
=====
```

Ahora usaremos la segunda opción. Como ven es muy fácil y hay que seguir las indicaciones.

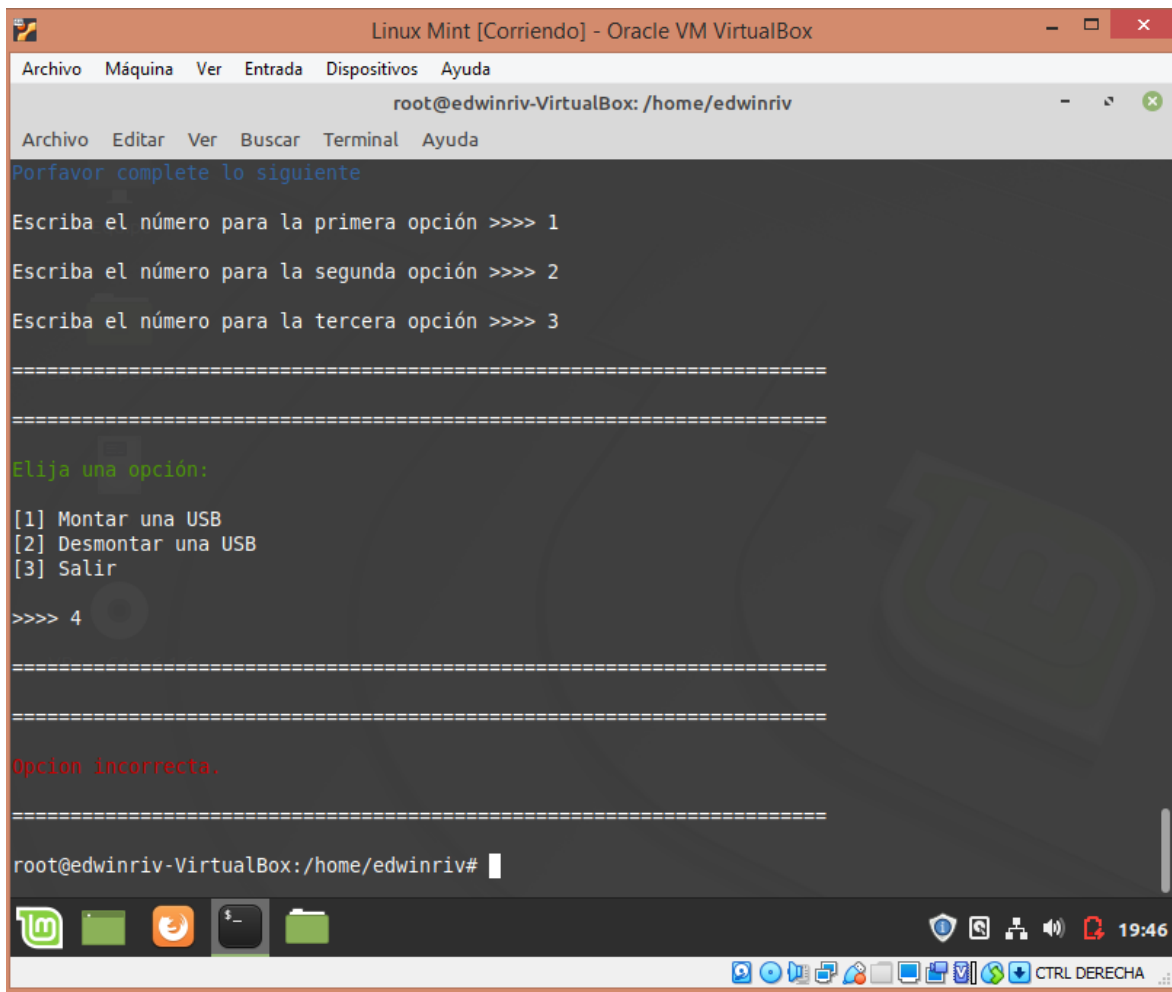


```
Linux Mint [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
root@edwinriv-VirtualBox: /home/edwinriv
Archivo Editar Ver Buscar Terminal Ayuda
Escriba el número para la primera opción >>>> 1
Escriba el número para la segunda opción >>>> 2
Escriba el número para la tercera opción >>>> 3
=====
=====
Elija una opción:
[1] Montar una USB
[2] Desmontar una USB
[3] Salir
>>>> 2
=====
Para desmontar la USB, solo debes de escribir su directorio que seria /dev/sdb1 :/dev/sdb1
Listo, se ha desmontado la USB.
Para asegurarnos, se ejecutaran los comandos pwd y ls.
/home/edwinriv
Descargas Documentos Ejercicio8.sh Escritorio Imágenes Música Plantillas Público Vídeos
=====
```

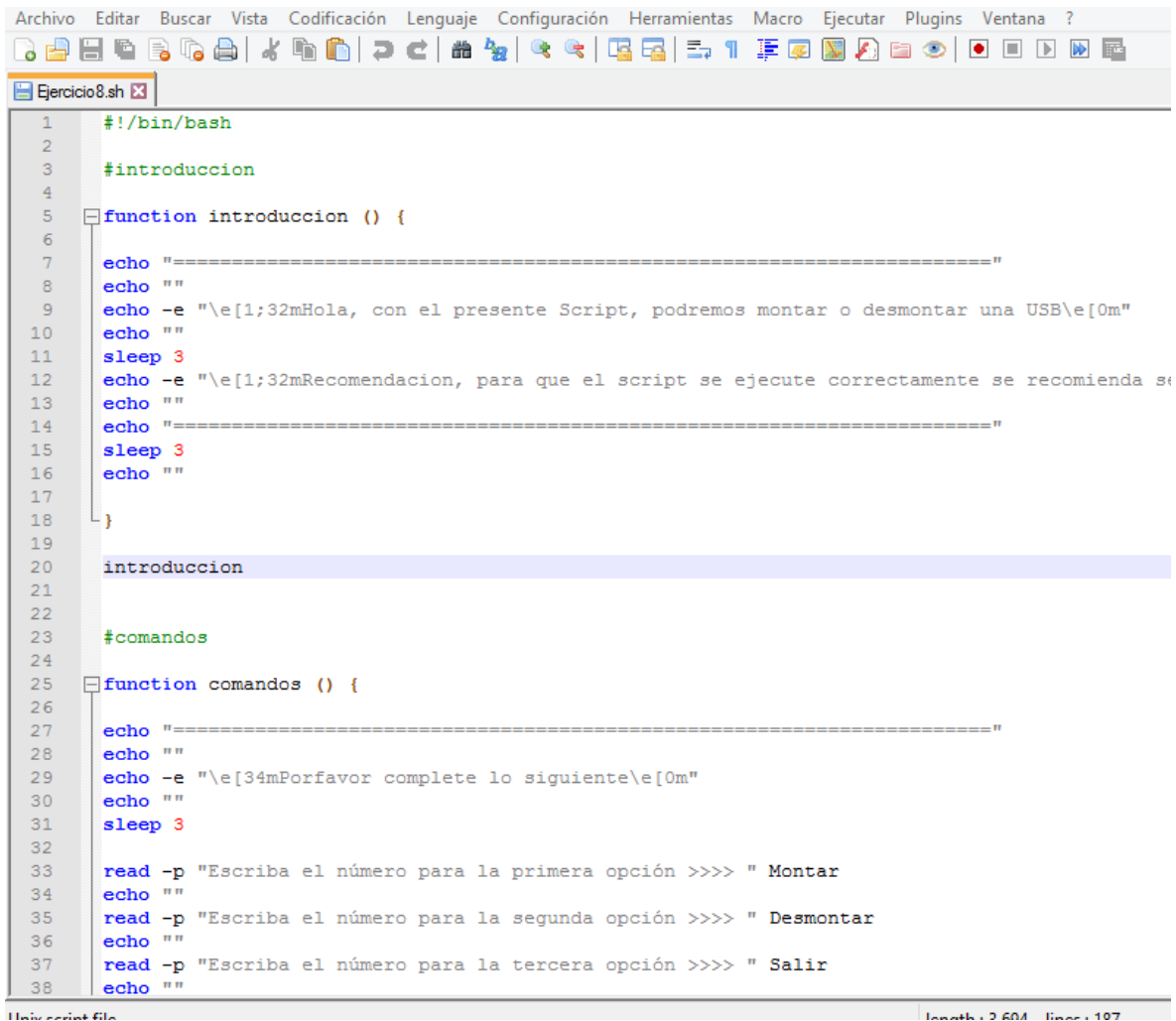
Esta sería la opción tres.



Y este enunciado sale por si nos equivocamos al elegir una opción.

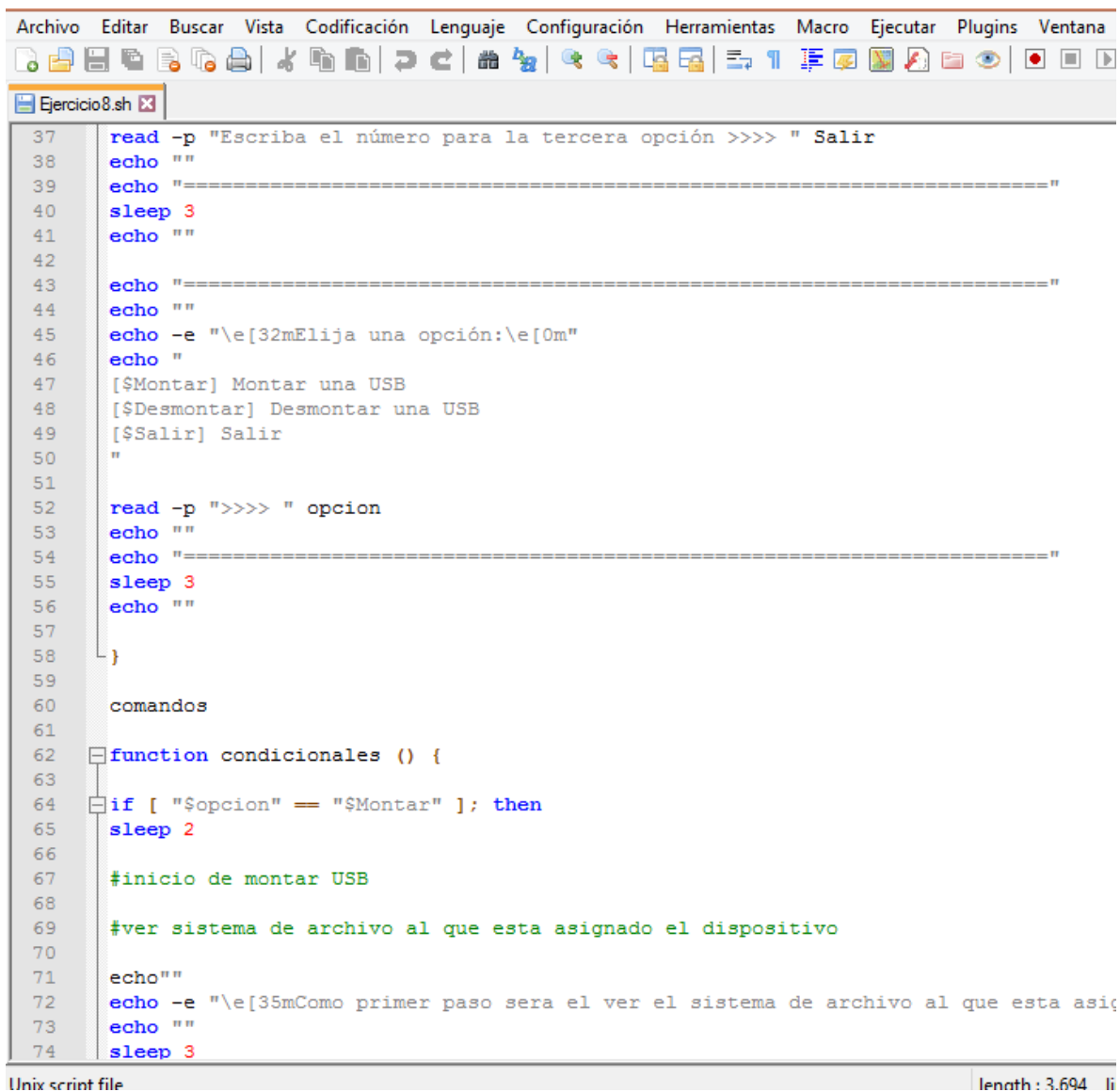


Iniciaremos con la función llamada introducción, en la cual irán los enunciados de inicio. Luego comenzaremos una función llamada comandos en la cual irán los comandos para crear el mini menú, donde el usuario deberá de asignar las teclas.



```
1  #!/bin/bash
2
3  #introduccion
4
5  function introduccion () {
6
7  echo "=====
8  echo ""
9  echo -e "\e[1;32mHola, con el presente Script, podremos montar o desmontar una USB\e[0m"
10 echo ""
11 sleep 3
12 echo -e "\e[1;32mRecomendacion, para que el script se ejecute correctamente se recomienda s
13 echo ""
14 echo "=====
15 sleep 3
16 echo ""
17
18 }
19
20 introduccion
21
22
23 #comandos
24
25 function comandos () {
26
27 echo "=====
28 echo ""
29 echo -e "\e[34mPorfavor complete lo siguiente\e[0m"
30 echo ""
31 sleep 3
32
33 read -p "Escriba el número para la primera opción >>>> " Montar
34 echo ""
35 read -p "Escriba el número para la segunda opción >>>> " Desmontar
36 echo ""
37 read -p "Escriba el número para la tercera opción >>>> " Salir
38 echo ""
```

Y acá sigue la creación de la segunda función. Finalizamos y creamos una nueva función llamada condicional.



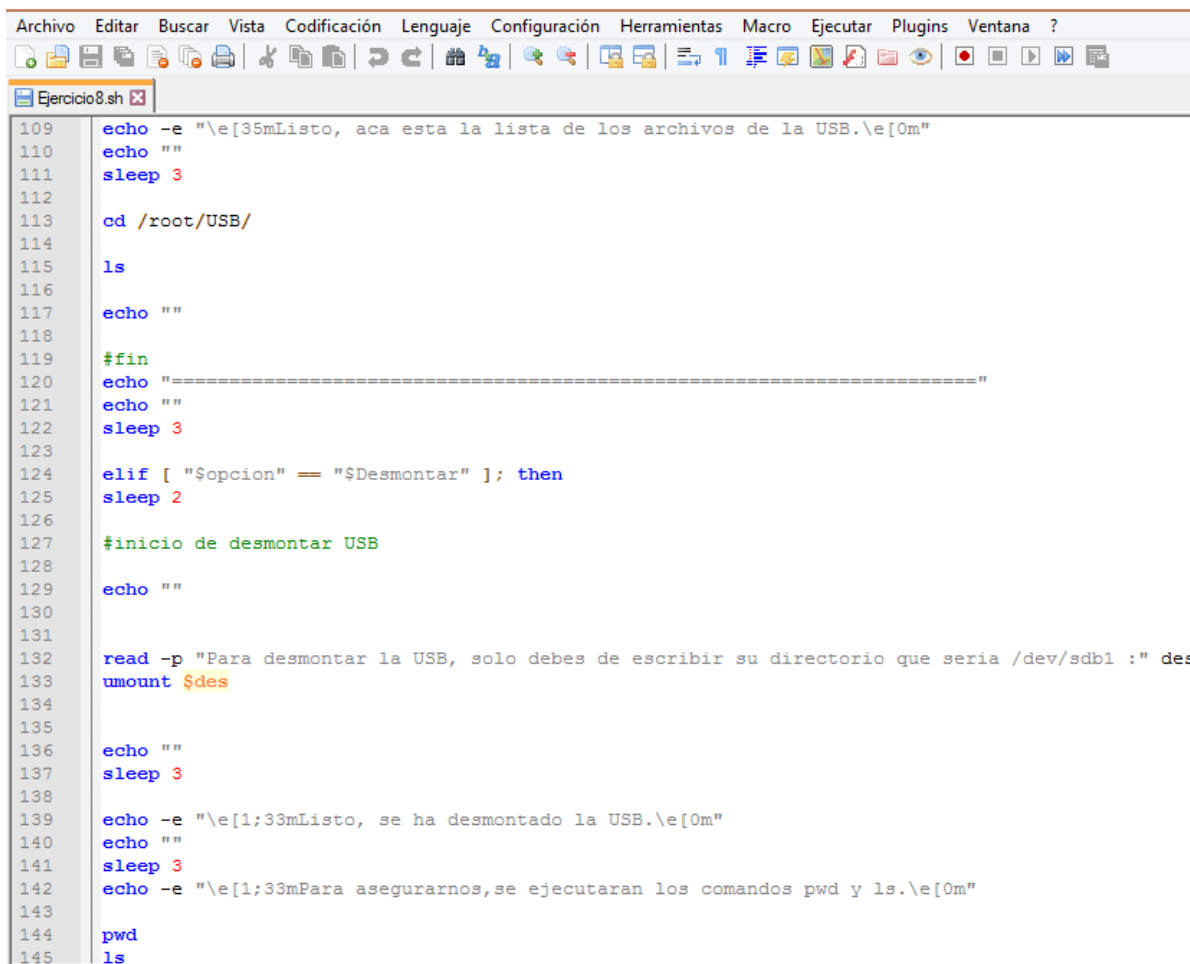
```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana
Ejercicio8.sh
37 read -p "Escriba el número para la tercera opción >>>> " Salir
38 echo ""
39 echo "=====
40 sleep 3
41 echo ""
42
43 echo "=====
44 echo ""
45 echo -e "\e[32mElija una opción:\e[0m"
46 echo "
47 [$Montar] Montar una USB
48 [$Desmontar] Desmontar una USB
49 [$Salir] Salir
50 "
51
52 read -p ">>>> " opcion
53 echo ""
54 echo "=====
55 sleep 3
56 echo ""
57
58 }
59
60 comandos
61
62 function condicionales () {
63
64 if [ "$opcion" == "$Montar" ]; then
65     sleep 2
66
67     #inicio de montar USB
68
69     #ver sistema de archivo al que esta asignado el dispositivo
70
71     echo ""
72     echo -e "\e[35mComo primer paso sera el ver el sistema de archivo al que esta asi
73     echo ""
74     sleep 3
```

Unix script file | length: 3.694 li

Acá continuaremos con los comandos para montar la USB.

```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
Ejercicio8.sh x
73  echo ""
74  sleep 3
75
76  fdisk -l
77  sleep 5
78
79  #identificar la USB
80
81  echo ""
82  echo -e "\e[35mAhora que ya vimos los dispositivos de almacenamiento actualmente activos
83  echo ""
84  sleep 3
85  echo -e "\e[35mNota, la USB puede ser detectada como sdc1, sdb1, dependiendo de los disp
86  echo ""
87
88  sleep 5
89
90  #creando carpeta para la USB
91
92  read -p "Porfavor escribir el siguiente directorio /root/USB/ : " dir
93  mkdir $dir
94
95
96  sleep 3
97  echo ""
98
99  #montar la USB
100
101
102  read -p "Ahora digitaremos /dev/sdb1 /root/USB/ :" mon
103  mount $mon
104
105
106  sleep 3
107  echo ""
108
109  echo -e "\e[35mListo, aca esta la lista de los archivos de la USB.\e[0m"
```

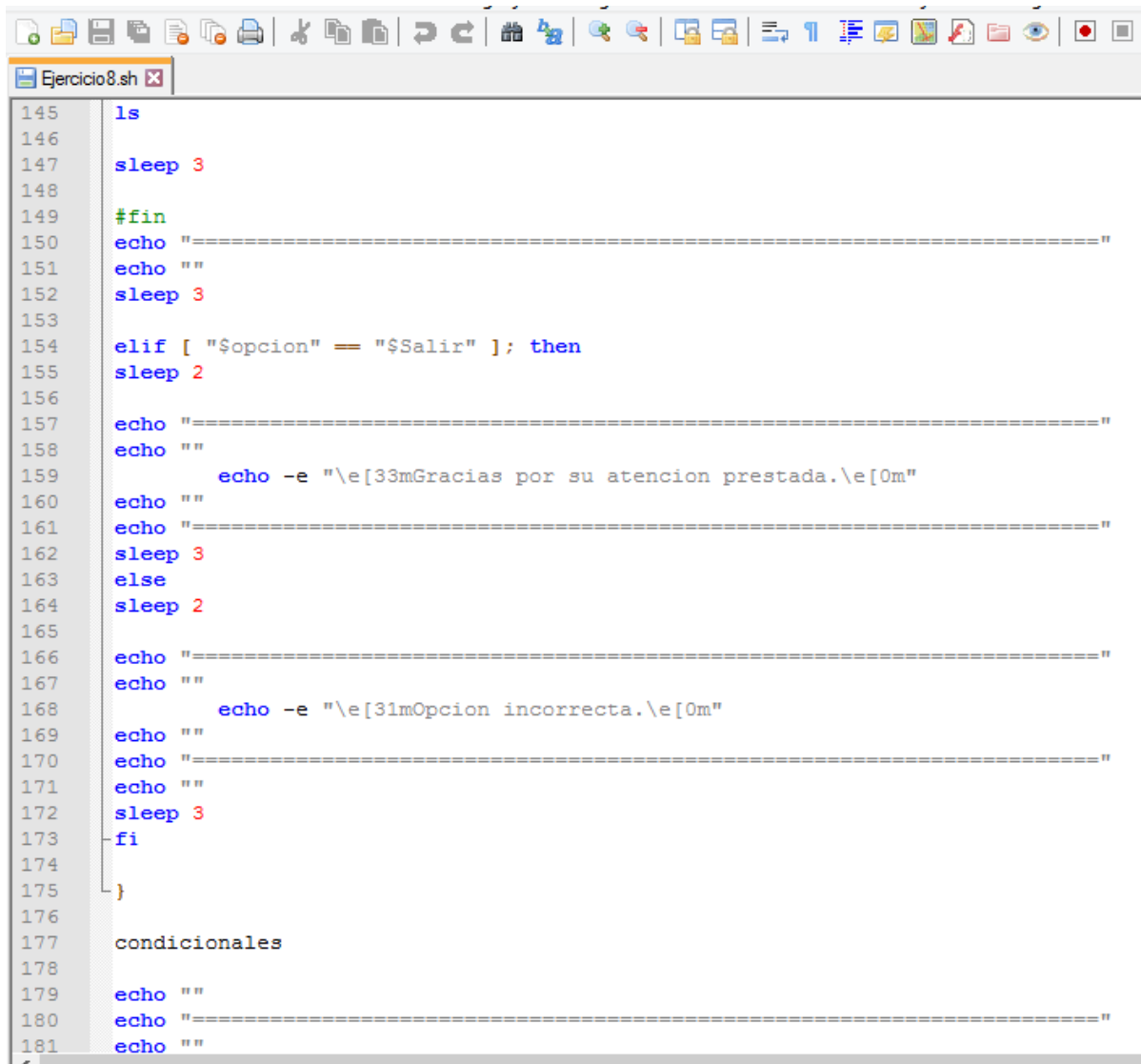
Como vemos, serán múltiples comandos solo para montar la USB. Mientras que para desmontarla será más sencillo.



```
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Herramientas  Macro  Ejecutar  Plugins  Ventana  ?
Ejercicio8.sh
109  echo -e "\e[35mListo, aca esta la lista de los archivos de la USB.\e[0m"
110  echo ""
111  sleep 3
112
113  cd /root/USB/
114
115  ls
116
117  echo ""
118
119  #fin
120  echo "-----"
121  echo ""
122  sleep 3
123
124  elif [ "$opcion" == "$Desmontar" ]; then
125  sleep 2
126
127  #inicio de desmontar USB
128
129  echo ""
130
131
132  read -p "Para desmontar la USB, solo debes de escribir su directorio que seria /dev/sdb1 :" des
133  umount $des
134
135
136  echo ""
137  sleep 3
138
139  echo -e "\e[1;33mListo, se ha desmontado la USB.\e[0m"
140  echo ""
141  sleep 3
142  echo -e "\e[1;33mPara asegurarnos, se ejecutaran los comandos pwd y ls.\e[0m"
143
144  pwd
145  ls
```



Estos serían los últimos enunciados y final del código.



```
145 ls
146
147 sleep 3
148
149 #fin
150 echo "=====
151 echo ""
152 sleep 3
153
154 elif [ "$opcion" == "$Salir" ]; then
155 sleep 2
156
157 echo "=====
158 echo ""
159     echo -e "\e[33mGracias por su atencion prestada.\e[0m"
160 echo ""
161 echo "=====
162 sleep 3
163 else
164 sleep 2
165
166 echo "=====
167 echo ""
168     echo -e "\e[31mOpcion incorrecta.\e[0m"
169 echo ""
170 echo "=====
171 echo ""
172 sleep 3
173 fi
174
175 }
176
177 condicionales
178
179 echo ""
180 echo "=====
181 echo ""
```