



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
Facultad de Ingeniería

GNU/Linux
Proyecto Final en SHELL

Alumnos:

Roldán Montero Hugo Alejandro
Marquez Sanchez Mirna Daniela

21 de Abril de 2023



Índice

1. Introducción	2
2. Desarrollo	2
3. Uso de la terminal	30
4. Conclusion	35



1. Introducción

En este proyecto se aplicarán los conocimientos adquiridos durante el curso de GNU/Linux, mediante la implementación de un emulador de una terminal, esta terminal cuenta con diferentes características establecidas, haciendo uso del lenguaje "bash.^{en} Shell Script, se tocarán temas como la estructura de linux, implementación de código, sentencias condicionales (if, case, while, for) y comandos esenciales en el uso de linux, junto con sus banderas.

2. Desarrollo

Para este proyecto utilizamos bash, en una terminal de Linux, sistema "Unix like" mediante la creación de scripts, se solicitaron en general las siguientes características, mas adelante se desglosará como hicimos cada uno de los scripts, explicados:

- Un sistema de acceso
- Interacción por línea de comandos
- Bloqueo de las señales para salir de la terminal (Ctrl + C / V)
- Creación del comando 'ayuda'
- Comando "infosis"
- Comando "buscar"
- Comando que muestre fecha y hora
- Comando que muestre los créditos del programador
- Comando de juego de gato



- Comando de juego de ahorcado

- Comando de reproductor MP3

- **Comando 'MP3':**

Este fue uno de los más laboriosos de utilizar, ya que requeríamos de previamente seleccionar las canciones que queríamos utilizar, para ingresarlas en una carpeta que contuviera las reproducciones, posteriormente se elegirá una canción con .opcion.el cual da la opción al usuario de elegir entre distintas canciones que hemos elegido para él/ella. Incluso al seleccionar algunas canciones que contengan información completa, nos muestra el año, artista, año. Lo que cumple con la función completa de cómo se ve un mp3 al que nosotros estamos acostumbrados a utilizar.

```
1      #!/bin/bash
2 trap  ''  2  20
3
4 prueba=/etc/sgml/pi
5
6 if [ -d "$prueba" ]
7 then
8     sudo mkdir /etc/sgml/pi2
9     clear
10 else
11     clear
12     echo "Porfavor ingresa la contrasena para reproducir musica"
13     sudo apt -y install mpg123
14     clear
15     sudo mkdir /etc/sgml/pi
16 fi
17
18 clear
19 bandera2=0
20
```



```
21 while [ $bandera2 -ne 1 ]
22 do
23
24     echo -e "\e[36mUbicacion de la carpeta puede ser ruta absoluta "
25     echo -e "o relativa antes de un espacio porfavor colocar '\ ' \e[0m"
26     read -e ubiact
27
28     opcion=0
29     band=1
30
31     while [ $opcion -ne 2 ]
32     do
33         clear
34
35         echo -e "Te encuentras en \e[1;32m$ubiact\e[0m "
36         echo "Menu del reproductor prebeshell: "
37         echo "Reproduce todas las canciones de la carpeta con '1'"
38         echo "Salir del reproductor prebeshell con '2'"
39         echo "deseas cambiar de carpeta '3' si "
40         echo "Que quieres hacer"
41         read -e opcion
42
43         if [ $opcion -le 0 -o $opcion -ge 4 ]
44         then
45             echo "Error numero no existe profavor ingrese cualquier tecla para
continuar"
46             read -e opcion
47             opcion=0;
48         elif [ $opcion -eq 3 ]
49         then
50             opcion=2
51         elif [ $opcion -eq 2 ]
52         then
```



```
53     bandera2=1
54
55
56     else
57         command clear
58
59
60         if [ $opcion -eq 1 ]
61         then
62
63             cont=1
64             for archivo in $ubiact/*.mp3; do
65
66                 canciones[$(($cont-1))]="${archivo##*/}"
67                 cont=$(($cont+1))
68             done
69
70             if [ $(which mpg123) ]
71             then
72
73                 while [ $band -eq 1 ]
74                 do
75                     echo -e "\e[34mEste es el Menu de reproduccion de directorio
76 \e[0m"
77
78                     echo -e "Estas en la carpeta \e[32m$ubiact\e[0m"
79
80                     cont=1
81                     for archivo in $ubiact/*.mp3; do
82                         echo "$cont. ${archivo##*/}"
83                         canciones[$(($cont-1))]="${archivo##*/}"
84                         cont=$(($cont+1))
85                     done
```

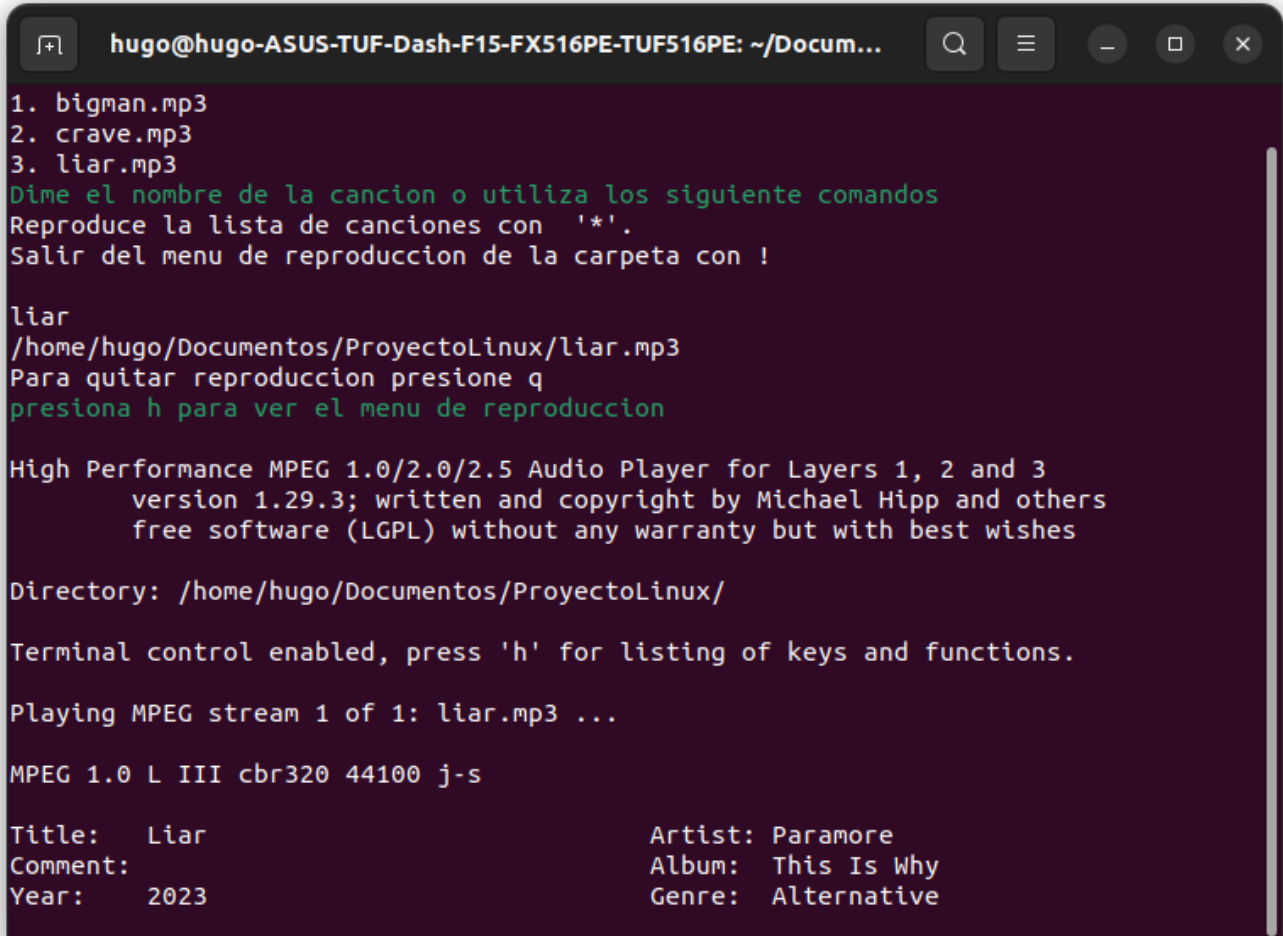


```
85         echo -e "\e[32mDime el nombre de la cancion o utiliza los
siguiente comandos\e[0m"
86         echo "Reproduce la lista de canciones con '*'."
87         echo "Salir del menu de reproduccion de la carpeta con !"
88         echo " "
89         read -e opcion
90
91         if [ "$opcion" = "!" ]
92         then
93             band=0
94
95         elif [ "$opcion" = "*" ]
96         then
97             command clear
98             echo ""
99             echo ""
100            echo "Para quitar reproduccion presione q"
101            echo -e "\e[32mpresiona h para ver el menu de
reproduccion\e[0m"
102            echo ""
103            command mpg123 -q "${ubiact}"/*
104
105        else
106            echo "${ubiact}/${opcion}.mp3"
107            echo "Para quitar reproduccion presione q"
108            echo -e "\e[32mpresiona h para ver el menu de
reproduccion\e[0m"
109            echo ""
110            command mpg123 -C "${ubiact}/${opcion}.mp3"
111
112            command clear
113            echo "Se acabaron las canciones presiona cualquier tecla
```



```
    para continuar"  
115         read le  
116     fi  
117  
118     command clear  
119  
120     opcion=0  
121     done  
122     band=1  
123     fi  
124     fi  
125  
126     fi  
127     done  
128 done  
129  
130 command clear
```


Figura 1: Ejemplo de reproducción



```
hugo@hugo-ASUS-TUF-Dash-F15-FX516PE-TUF516PE: ~/Docum...
1. bigman.mp3
2. crave.mp3
3. liar.mp3
Dime el nombre de la cancion o utiliza los siguiente comandos
Reproduce la lista de canciones con '*'.
Salir del menu de reproduccion de la carpeta con !

liar
/home/hugo/Documentos/ProyectoLinux/liar.mp3
Para quitar reproduccion presione q
presiona h para ver el menu de reproduccion

High Performance MPEG 1.0/2.0/2.5 Audio Player for Layers 1, 2 and 3
    version 1.29.3; written and copyright by Michael Hipp and others
    free software (LGPL) without any warranty but with best wishes

Directory: /home/hugo/Documentos/ProyectoLinux/

Terminal control enabled, press 'h' for listing of keys and functions.

Playing MPEG stream 1 of 1: liar.mp3 ...

MPEG 1.0 L III cbr320 44100 j-s

Title:   Liar
Comment:
Year:    2023
Artist:  Paramore
Album:   This Is Why
Genre:   Alternative
```

■ Comando 'juego de gato':

Este juego es interactivo para dos personas, a ambos jugadores al inicializar el programa, se le asigna un símbolo diferente, esto lo hacemos con "symbol", utilizamos una lógica de tipo matricial, ya que se seleccionan las columnas y filas para indicar dónde será puesto el símbolo del jugador, de ya contener un símbolo, nos da un mensaje que no se encuentra disponible, o si damos un valor fuera del rango, nuestro programa también lo detecta. De no haber ganador, podemos reiniciar el juego y comenzar otra vez.



```

1      #!/bin/bash
2
3  iniciarjuego()
4  {
5      echo "===== "
6      echo " | |           JUEGO DE GATO PADRE           | | "
7      echo " | |           :)           | | "
8      echo " | |           (2 jugadores)           | | "
9      echo "===== "
10     echo ""
11     echo "Preparando el juego...."
12     echo ""
13     jugadorgato=1
14     tablero=(- - - - - - - - -)
15     ganador=1
16     echo "***** "
17     echo " | |           | | "
18     echo " | |           INICIA     EL JUEGO!!!           | | "
19     echo " | |           | | "
20     echo "***** "
21 }
22
23 imprimirtablero()
24 {
25     echo "Rows\Columns    0    1    2 "
26     echo ""
27     echo "          -----"
28     echo "      0      | ${tablero[0]} | ${tablero[1]} | ${tablero[2]} | "
29     echo "          -----"
30     echo "      1      | ${tablero[3]} | ${tablero[4]} | ${tablero[5]} | "
31     echo "          -----"
32     echo "      2      | ${tablero[6]} | ${tablero[7]} | ${tablero[8]} | "
33     echo "          -----"

```



```
34 }
35
36 setInput()
37 {
38     lugares=$(( $1 * 3 + $2 ))
39     if [ ${tablero[$lugares]} == "-" ]
40     then
41         tablero[$lugares]=$3
42         jugadorgato=$((jugadorgato%2+1))
43     else
44         echo -e "\n--CUIDADO-- ESTE LUGAR YA EST  OCUPADO"
45         echo -e "Intenta otra vez!!!!"
46     fi
47 }
48
49 checarganador()
50 {
51     simbolos 0 1 2
52     simbolos 3 4 5
53     simbolos 6 7 8
54     simbolos 0 4 8
55     simbolos 2 4 6
56     simbolos 0 3 6
57     simbolos 1 4 7
58     simbolos 2 5 8
59 }
60
61 simbolos()
62 {
63     if [ ${tablero[$1]} != "-" ]&&[ ${tablero[$1]} == ${tablero[$2]} ]&&[ ${
        tablero[$2]} == ${tablero[$3]} ]
64     then
65         ganador=0
```



```
66     fi
67 }
68
69 iniciarjuego
70 while true
71 do
72     echo ""
73     if [ $jugadorgato == 1 ]
74     then
75         symbol=X
76     else
77         symbol=O
78     fi
79
80     echo "Jugador $jugadorgato es tu turno: ($symbol)"
81     echo ""
82     imprimirtablero
83     echo ""
84     echo "#####"
85     echo "||"
86     echo "||"
87     echo "||"
88     echo "||"
89     echo "||"
90     echo "#####"
91     echo ""
92
93     while true
94     do
95         printf "Teclea una letra: "; read -r inputCommand a b
96         case $inputCommand in
97             "A")
98             setInput $a $b $symbol
```



```
99         break
100     ;;
101     "B")
102     iniciarjuego
103     ;;
104     "C")
105     exit 0
106     break
107     ;;
108     *)
109     echo ""
110     echo "Comando no disponible, intenta otra vez"
111     ;;
112     esac
113
114
115     #read -r inputCommand a b
116     #if [ $inputCommand == "A" ]
117     #then
118     #setInput $a $b $symbol
119     #break
120     #elif [ $inputCommand == "reset" ]
121     #then
122     #    iniciarjuego
123     #    break
124     #else
125     #    echo ""
126     #    echo "Comando no disponible, intenta otra vez"
127     #fi
128     done
129
130     checarganador
131
```



```
132         if [ $ganador != 1 ]
133     then
134         jugadorgato=$((jugadorgato%2+1))
135         imprimirtablero
136         echo "===== "
137         echo "| |                                     | |"
138         echo "| |             FIN DE LA PARTIDA             | |"
139         echo "| |     jugador $jugadorgato ($symbol) gana     | |"
140         echo "| |                                     | |"
141         echo "===== "
142         echo ""
143         echo "Escribe la palabra \"reset\" para reiniciar el juego"
144         read -r inputCommand n
145         while true
146         do
147             if [ $inputCommand == "reset" ]
148         then
149                 iniciarjuego
150                 break
151             fi
152         done
153     fi
154 done
```

Figura 2: Juego en terminal

```
mirna@mirna-HP-ENVY-m4-Notebook-PC: ~/Escritorio/terminalprebe
Archivo Editar Ver Buscar Terminal Ayuda
Jugador 1 es tu turno: (X)
Filas\Columnas  0  1  2

  0      | - | X | X |
  1      | - | 0 | - |
  2      | - | 0 | - |

#####
||                Inserta la letra :                ||
||      A. para jugar Ej: A (fila) (columna)      ||
||      B. Reset                                  ||
||      C. Salir                                  ||
||                #####                            ||
Tecllea una letra: A 0 0
Filas\Columnas  0  1  2

  0      | X | X | X |
  1      | - | 0 | - |
  2      | - | 0 | - |

=====
||                FIN DE LA PARTIDA                ||
||      jugador 1 (X) gana                          ||
||                =====                          ||

Escribe la palabra "reset" para reiniciar el juego
```

■ Comando 'juego de ahorcado':

En este juego utilizamos una metodología muy sencilla, donde cada palabra ingresada por el usuario es repetida con un echo y después de eso es comprobada con la palabra inicial a adivinar. Sigue la estructura de que es para 2 o más jugadores, jugarán 1 vs 1, el jugador ingresa la palabra que desea que el otro jugador adivine, así pues el jugador adivinador tiene 5 intentos de adivinar, de no ser así, hicimos gráficamente un pequeño personaje que irá apareciendo al no ser correcta la palabra, esto con la variable de 'error' que compara letra por



letra. Hasta adivinar la palabra, o no; dejamos la adivinanza en manos de nuestro usuario.

```
1 #!/bin/bash
2
3 ,
4
5  _ _ _ _
6 |   \ |
7 |   |
8 |   |
9 |   |
10 _ _ _ _ | _ _ _
11
12
13  _ _ _ _
14 |   \ |
15 |   |
16 0   | ERROR. Esa letra no es correcta
17 |   |
18 |   |
19 _ _ _ _ | _ _ _
20
21
22  _ _ _ _
23 |   \ |
24 |   |
25 0   |
26 |   | ERROR 2. Esa letra no es correcta
27 |   |
28 _ _ _ _ | _ _ _
29
30
31  _ _ _ _
32 |   \ |
```




```
33      |      |
34      0      |
35      /\      | ERROR 3. Ya tienes tres errores, ya tienes casi la mitad del
          cuerpo    CUIDADO    !!
36          |
37  -----|-----
38
39
40      -----
41      |      \|
42      |      |
43      0      |
44      /\      |
45      /      | ERROR 4. Te queda una oportunidad mas, Tu PUEDES VAMOS, salva a
          Panchito
46  -----|-----
47
48
49      -----
50      |      \|
51      |      |
52      0      |
53      /\      |
54      / \     | ERROR 5. No te tocaba carnal, PERDISTE
55  -----|-----
56      ,
57
58  clear
59  echo -n " QUE PALABRA VAMOS A ADIVINAR?: "
60  read word
61  echo
62  # echo "Pulsa enter para continuar..."
63  # read continuar
```



```
64 clear
65 letra="*"
66
67 fallos=5
68
69 letras='echo $word | sed "s/[~${letra}]/"/g'
70
71 sust=$letras
72
73 while [ "$sust" != "$word" ]
74
75     do
76         # clear
77         echo
78         echo "LAS LETRAS SON $sust"
79         echo
80         sed -n "$fallos,$((fallos+7))p" $0
81         echo
82         echo LETRAS INTRODUCIDAS HASTA AHORA: $a
83         echo
84         echo -n "INTRODUCE UNA LETRA: "
85         read letra
86         a=${a}$letra
87         palabra=${letra}$palabra
88
89         echo
90         existe='echo ${word} | grep ${letra}'
91
92         if [ "$existe" = "" ]
93             then
94                 echo "Esa letra no es correcta, la letra - ${letra} - no es
95 parte de la palabra"
96                 error=$((error + 1))
```



```
96         if [ $error = 5 ]
97             then
98 #               clear
99         fallos=$((fallos+9))
100         sed -n "$fallos,$((fallos+7))p" $0
101         echo "Has cometido $error errores, RIP"
102             echo
103         echo
104         exit
105             else
106                 if [ $error = 1 ]
107                     then
108                         echo "Has cometido $error errores"
109                         fallos=$((fallos+9))
110                     else
111                         fallos=$((fallos+9))
112                     fi
113                 fi
114
115         else
116             sust='echo $word | sed "s/[^${palabra}]/*/g'
117             fi
118
119             if [ "${sust}" = "${word}" ]
120                 then
121                     echo "      FELICIDADES      !!! HAS DESCUBIERTO LA
PALABRA SECRETA QUE ERA: ${word}."
122                     echo
123                     echo
124                     fi
125
126         done
```

```
mirna@mirna-HP-ENVY-m4-Notebook-PC:~$
```

```
Archivo Editar Ver Buscar Terminal Ayuda
```

```
INTRODUCE UNA LETRA: L
```

```
LAS LETRAS SON *OL*
```

```
  | \ |  
  |  \|  
0  |  
  |  
_____|_____ ERROR 2. Esa letra no es correcta
```

```
LETRAS INTRODUCIDAS HASTA AHORA: OIRL
```

```
INTRODUCE UNA LETRA: H
```

```
LAS LETRAS SON HOL*
```

```
  | \ |  
  |  \|  
0  |  
  |  
_____|_____ ERROR 2. Esa letra no es correcta
```

```
LETRAS INTRODUCIDAS HASTA AHORA: OIRLH
```

```
INTRODUCE UNA LETRA: A
```

```
¡¡¡FELICIDADES!!! HAS DESCUBIERTO LA PALABRA SECRETA QUE ERA: HOLA.
```

```
mirna@mirna-HP-ENVY-m4-Notebook-PC:~/Escritorio/terminalprebes$
```

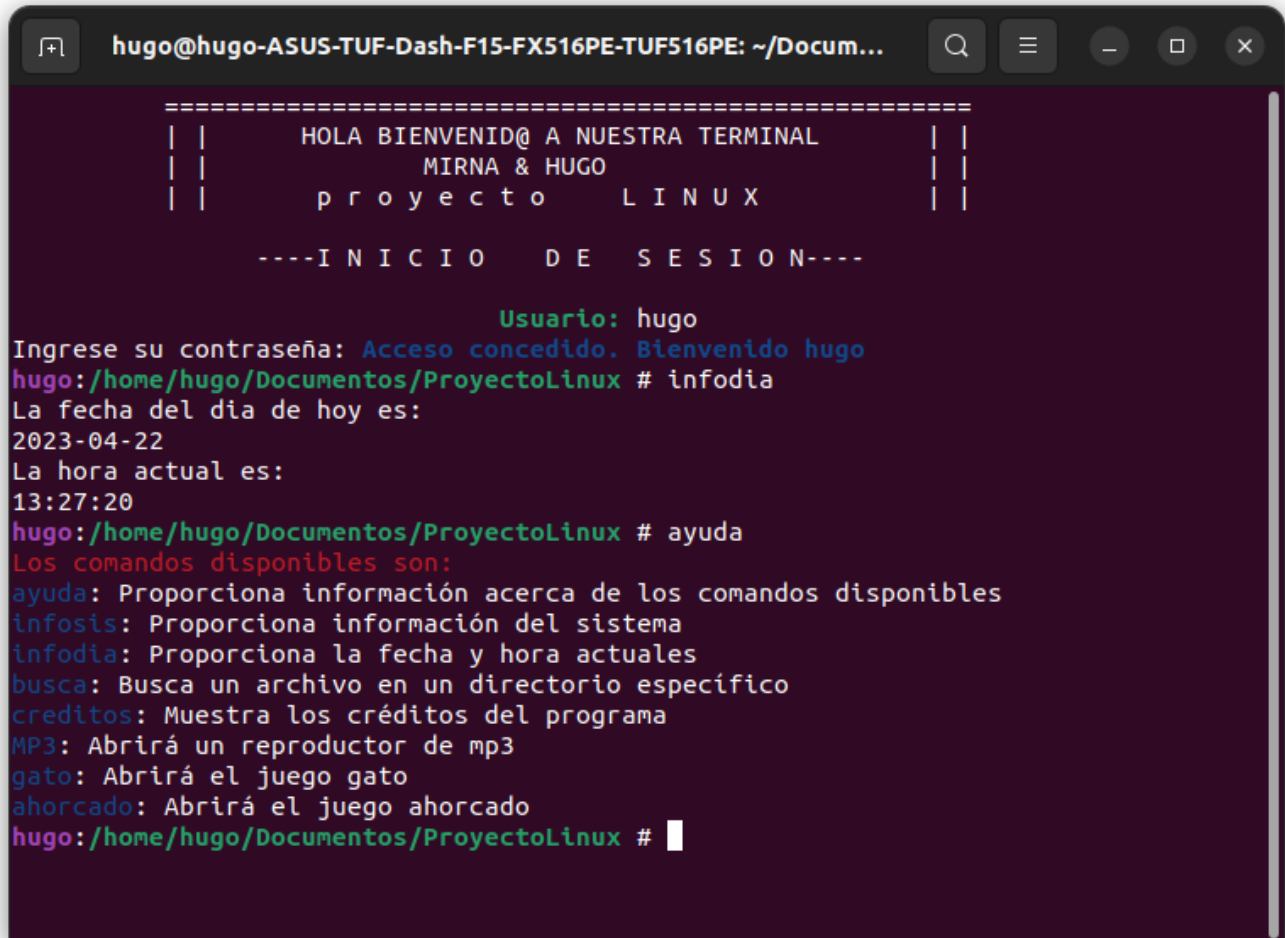


■ Comando 'ayuda'

Este script, utiliza básicamente los comandos 'echo' cuyo uso es mostrar en pantalla lo deseado, junto con la bandera -e, que nos permitirá interpretar los colores deseados, finalmente, se investigaron los códigos de colores y se aplicaron como se muestra en el código:

```
1      #!/bin/bash
2  rojo="\033[31m"
3  verde="\033[32m"
4  azul="\033[34m"
5  reset_color="\033[0m"
6
7  echo -e "${rojo}Los comandos disponibles son:"
8  echo -e "${azul}ayuda${reset_color}: Proporciona informacion acerca de los
    comandos disponibles"
9  echo -e "${azul}infosis${reset_color}: Proporciona informacion del sistema"
10 echo -e "${azul}infodia${reset_color}: Proporciona la fecha y hora actuales"
11 echo -e "${azul}busca${reset_color}: Busca un archivo en un directorio
    especifico"
12 echo -e "${azul}creditos${reset_color}: Muestra los creditos del programa"
13 echo -e "${azul}MP3${reset_color}: Abrira un reproductor de mp3"
14 echo -e "${azul}gato${reset_color}: Abrira el juego gato"
15 echo -e "${azul}ahorcado${reset_color}: Abrira el juego ahorcado"
```

Figura 4: Ejemplo comando ayuda



```
hugo@hugo-ASUS-TUF-Dash-F15-FX516PE-TUF516PE: ~/Docum...  
=====HOLA BIENVENID@ A NUESTRA TERMINAL=====  
| | MIRNA & HUGO | |  
| | p r o y e c t o L I N U X | |  
-----INICIO DE SESION-----  
                Usuario: hugo  
Ingrese su contraseña: Acceso concedido. Bienvenido hugo  
hugo:/home/hugo/Documentos/ProyectoLinux # infodia  
La fecha del día de hoy es:  
2023-04-22  
La hora actual es:  
13:27:20  
hugo:/home/hugo/Documentos/ProyectoLinux # ayuda  
Los comandos disponibles son:  
ayuda: Proporciona información acerca de los comandos disponibles  
infosis: Proporciona información del sistema  
infodia: Proporciona la fecha y hora actuales  
busca: Busca un archivo en un directorio específico  
creditos: Muestra los créditos del programa  
MP3: Abrirá un reproductor de mp3  
gato: Abrirá el juego gato  
ahorcado: Abrirá el juego ahorcado  
hugo:/home/hugo/Documentos/ProyectoLinux #
```

■ Comando 'juego de gato':

Este fue uno de los más laboriosos de utilizar, ya que requeríamos de previamente seleccionar las canciones que queríamos utilizar, para ingresarlas en una carpeta que contuviera las reproducciones, posteriormente se elegirá una canción con .opcion.el cual da la opción al usuario de elegir entre distintas canciones que hemos elegido para él7 ella. Incluso al seleccionar algunas canciones que contengan información completa, nos muestra el año, artista, año. Lo que cumple con la función completa de cómo se ve un mp3 al que nosotros estamos



acostumbrados a utilizar.

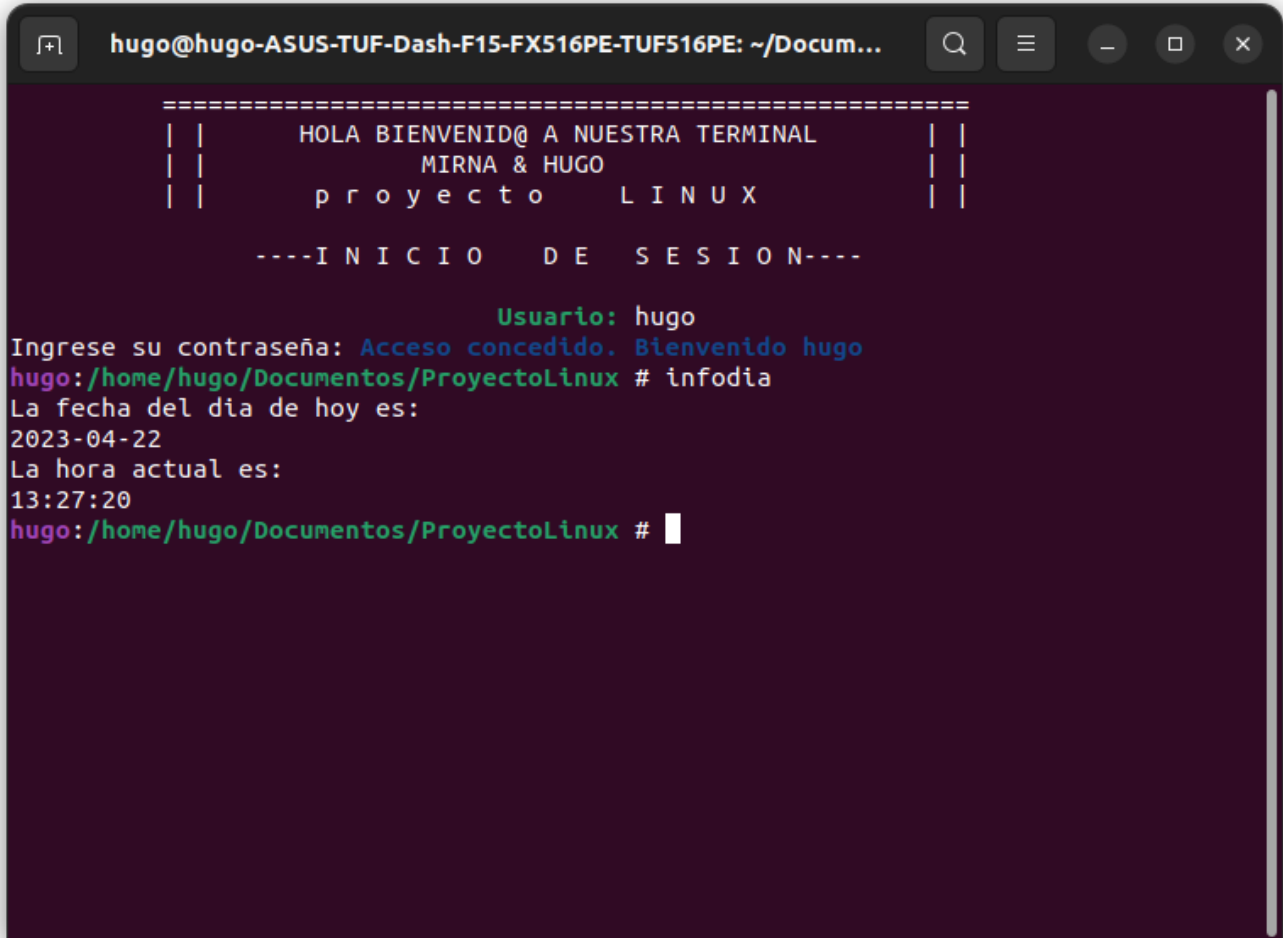
■ **Comando: 'infodia' :**

En este caso, se utilizó la carpeta **/proc/driver/rtc** debido a que en ese archivo, se almacenan los datos de la fecha del sistema, usamos los **grep** para canalizar la salida solamente a 'rtc-date' y finalmente, otro **grep** que busque solamente valores numéricos del 0 al 9, seguidos por un guión, la bandera **-O** indica que este ultimo, deberá ser la salida a mostrar.

En el caso de la hora, se utilizaron las variables de entorno simplemente para mostrar la hora que marca el reloj, quisimos usar dos métodos diferentes para variar el programa.

```
1  #!/bin/bash
2  echo "La fecha del dia de hoy es:"
3  cat /proc/driver/rtc | grep 'rtc_date' | grep '[0-9]*-[0-9]*-[0-9]*' -o
4
5  echo "La hora actual es: "
6  printf "%(%H:%M:%S)T\n"
7
```

Figura 5: Ejemplo comando infodia, que muestra hora y fecha



```
=====
| |      HOLA BIENVENID@ A NUESTRA TERMINAL      | |
| |      MIRNA & HUGO                            | |
| |      p r o y e c t o   L I N U X              | |
=====

----I N I C I O   D E   S E S I O N----

                                Usuario: hugo
Ingrese su contraseña: Acceso concedido. Bienvenido hugo
hugo:/home/hugo/Documentos/ProyectoLinux # infodia
La fecha del día de hoy es:
2023-04-22
La hora actual es:
13:27:20
hugo:/home/hugo/Documentos/ProyectoLinux #
```

■ Comando: Infosis

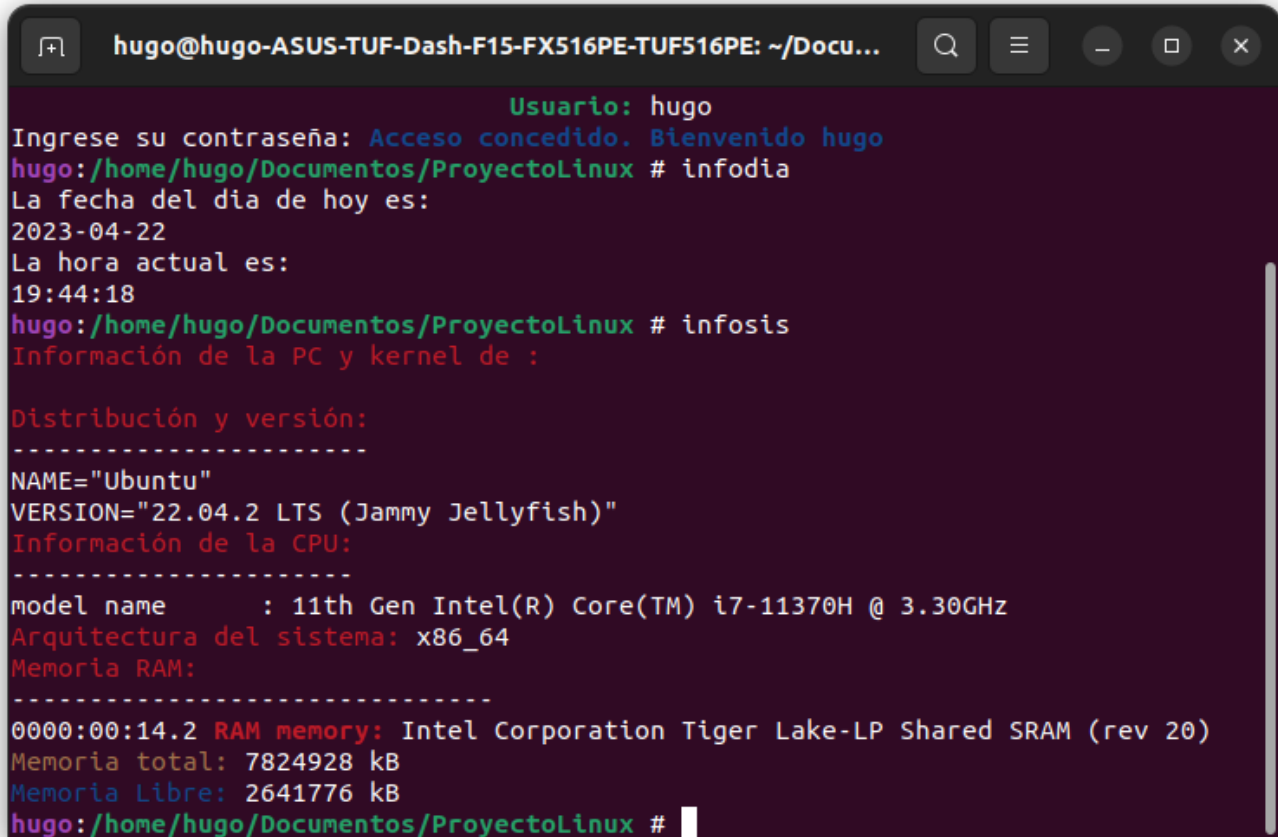
Para mostrar la información del sistema, primordialmente, se obtuvo de las carpetas de la raíz del sistema, desde etc, hasta proc, de igual forma, se utilizaron cat y grep para canalizar las salidas deseadas según el archivo y se utilizaron colores para contrastar de una mejor manera las salidas.

Para este comando hay muchas alternativas, intenté usar cada una de ellas en el programa, desde cat y grep, hasta utilizar comandos como variables para mostrar la salida.



```
1 #!/bin/bash
2 rojo="\033[31m"
3 verde="\033[32m"
4 azul="\033[34m"
5 amarillo="\033[33m"
6 sincolor="\033[0m"
7
8 echo -e "${rojo}Informacion de la PC y kernel:"
9 echo ""
10
11 echo -e "${rojo}Distribucion y version:${sincolor}"
12 echo "-----"
13 cat /etc/os-release | grep -e "^NAME=" -e "^VERSION="
14
15 echo -e "${rojo}Informacion de la CPU:${sincolor}"
16 echo "-----"
17 cat /proc/cpuinfo | grep -m 1 "model name"
18 echo -e "${rojo}Arquitectura del sistema:${sincolor} $(uname -m)"
19
20 echo -e "${rojo}Memoria RAM:${sincolor}"
21 echo "-----"
22 lspci | grep -i --color 'RAM memory:'
23 echo -e "${amarillo}Memoria total:${sincolor} $(cat /proc/meminfo | grep -i
    "Memtotal:" | grep '[1-9].*' -o)"
24 echo -e "${azul}Memoria Libre:${sincolor} $(cat /proc/meminfo | grep -i "
    Memfree:" | grep '[1-9].*' -o)"
25
```

Figura 6: Ejemplo comando infosis, que muestra algunas especificaciones del sistema



```

hugo@hugo-ASUS-TUF-Dash-F15-FX516PE-TUF516PE: ~/Docu...
                                Usuario: hugo
Ingrese su contraseña: Acceso concedido. Bienvenido hugo
hugo:/home/hugo/Documentos/ProyectoLinux # infodia
La fecha del día de hoy es:
2023-04-22
La hora actual es:
19:44:18
hugo:/home/hugo/Documentos/ProyectoLinux # infosis
Información de la PC y kernel de :

Distribución y versión:
-----
NAME="Ubuntu"
VERSION="22.04.2 LTS (Jammy Jellyfish)"
Información de la CPU:
-----
model name      : 11th Gen Intel(R) Core(TM) i7-11370H @ 3.30GHz
Arquitectura del sistema: x86_64
Memoria RAM:
-----
0000:00:14.2 RAM memory: Intel Corporation Tiger Lake-LP Shared SRAM (rev 20)
Memoria total: 7824928 kB
Memoria Libre: 2641776 kB
hugo:/home/hugo/Documentos/ProyectoLinux #

```

■ Comando: 'creditos'

En este comando se utilizó solamente el comando echo, imprimiendo cada uno de los datos mostrados en pantalla. En este caso no utilizamos colores para que se vea mas clásico.

```

1 clear
2 echo -e "\e[97;41m*~::~::~::~::~::~::~::~::~::~*\e[0m"
3 echo -e "\e[97;41m| El desarrollo de esta terminal|\e[0m"
4 echo -e "\e[97;41m|     fue posible gracias a:    |\e[0m"
5 echo -e "\e[97;41m*~::~::~::~::~::~::~::~::~::~*\e[0m"
6 echo -e "\e[97;41m| Marquez Sanchez Mirna Daniela |\e[0m"
7 echo -e "\e[97;41m| ~::~::~::~::~::~::~::~::~::~|\e[0m"

```



```
8 echo -e "\e[97;41m| Roldan Montero Hugo Alejandro |\e[0m"
9 echo -e "\e[97;41m*~::~::~::~::~::~::~::~::~::~::~*\e[0m"
10 echo -e "\e[97;41m| Gracias por usar nuestra terminal |\e[0m"
11 echo -e "\e[97;41m|           Hasta pronto! :) |\e[0m"
12 echo -e "\e[97;41m*~::~::~::~::~::~::~::~::~::~::~*\e[0m"
13
14
```

Figura 7: Ejemplo comando credits

■ Comando: 'buscar'

En este comando se utilizó principalmente el comando find.



Comenzamos solicitando al usuario la ruta absoluta y el nombre del archivo que desea encontrar. Para posteriormente abrir dicho directorio. Cuando se abre el directorio, guardaremos en la variable 'resultado' el nombre del archivo encontrado, la sentencia se leería así: 'encontrar un archivo de nombre 'nombre ingresado por el usuario'".

Finalmente se le da formato a la salida y se usa la sentencia if dependiendo si se encuentra o no el archivo.

```
1      #!/bin/bash
2      # ==--==--==--> Solamente son colores
3      R='\033[1;31m' # Rojo
4      G='\033[1;32m' # Verde
5      Y='\033[1;33m' # Amarillo
6      B='\033[1;34m' # Azul
7      M='\033[1;35m' # Mangenta
8      W='\033[0m' # Blanco
9      Glig='\e[1;32m' # Verde claro
10     #Pide los parametros
11
12     printf "${Glig}Ingrese la ruta absoluta del archivo${W}:"
13
14     read -r directorio
15
16     printf "${Glig}Ingrese el nombre del archivo con su extension${W}:"
17
18     read -r archivo
19
20     cd $directorio
21
22     # Realiza la busqueda del archivo en el directorio especificado
23     resultado=$(find -type f -name "$archivo")
24
25     if [ -n "$resultado" ]; then
26         printf "Archivo encontrado: $R $resultado ${W} en ${M} $directorio ${W}\
```



```

        n"
27 else
28     printf "Archivo ${R}'$archivo'${W} no encontrado en el directorio ${M}'
        $directorio'${W}\n"
29 fi
30

```

Figura 8: Ejemplo comando buscar

```

hugo@hugo-ASUS-TUF-Dash-F15-FX516PE-TUF516PE: ~/Documentos/Proye...
* ~ ~ ~ ~ ~ *
| El desarrollo de esta terminal |
| fue posible gracias a: |
* ~ ~ ~ ~ ~ *
| Márquez Sánchez Mirna Daniela |
| ~ ~ ~ ~ ~ |
| Roldán Montero Hugo Alejandro |
* ~ ~ ~ ~ ~ *
| Gracias por usar nuestra terminal |
| Hasta pronto! :) |
* ~ ~ ~ ~ ~ *
hugo:/home/hugo/Documentos/ProyectoLinux # buscar
Ingrese la ruta absoluta del archivo:/home/hugo/Documentos/ProyectoLinux
Ingrese el nombre del archivo con su extensión:Linux GEN 44.txt
Archivo encontrado: ./Linux GEN 44.txt en /home/hugo/Documentos/ProyectoLinux
hugo:/home/hugo/Documentos/ProyectoLinux #

```

■ Comando: 'salir'

En este caso se imprime una salida para despedir al usuario para posteriormente salir de la terminal con la sentencia exit.

```

1  #!/bin/bash
2  printf "\t  ===== \n"

```



```
3 printf "\t | |                MUCHAS GRACIAS                | |\n"
4 printf "\t | |                vuelve pronto                | |\n"
5 printf "\t | |                :)                | |\n"
6 printf "\n\t\t\t----C I E R R E      D E      S E S I O N----\n"
7     exit 0
8
9
```

Figura 9: Ejemplo comando salir

```
hugo@hugo-ASUS-TUF-Dash-F15-FX516PE-TUF516PE: ~/Docum...
hugo:/home/hugo/Documentos/ProyectoLinux # sal
El comando sal no existe. Utiliza ayuda para verificar los comandos disponibles.
hugo:/home/hugo/Documentos/ProyectoLinux # salir
=====
| |                MUCHAS GRACIAS                | |
| |                vuelve pronto                | |
| |                :)                | |
| |
| |                ----C I E R R E      D E      S E S I O N----
hugo@hugo-ASUS-TUF-Dash-F15-FX516PE-TUF516PE:~/Documentos/ProyectoLinux$
```



3. Uso de la terminal

Primero se definirá la función para evitar que se salga de la terminal utilizando los comandos Ctrl + Z/C.

En este caso el comando trap, evita que se ejecute cierta sentencia, SIGINT y SIGTSTP, corresponden a los comandos mencionados anteriormente. Si se detecta, llama a la función ignoresignal e imprime en pantalla el mensaje de error.

Posteriormente, se implementó el sistema de acceso, se leen el nombre y contraseña dados por el usuario, para posteriormente corroborar, en la ruta del sistema, si existe el usuario en el sistema y su contraseña es la dada, entonces brindará el acceso, de otra forma, repetirá el ciclo para intentar de nuevo.

Una vez brindado el acceso, se imprime el usuario proporcionado y la ruta actual del sistema con la variable pwd, que recordando, este comando proporciona al usuario la ubicación actual. posteriormente lee el comando.

Para saber como actuar según el comando proporcionado, se implementó un case, en donde cada uno de los casos es un comando de los mencionados en la sección anterior, la acción será ejecutar cada uno de los scripts asociados, en el caso de salir, saldrá del programa.

Para el caso default, primero corroborará si el comando existe en el sistema, si lo hace, entonces ejecutará el comando desde el sistema, de otra forma, se mostrará el mensaje de error.



```
1 #!/bin/bash
2
3 # ==--==--==--> Solamente son colores
4 R='\033[1;31m' # Rojo
5 G='\033[1;32m' # Verde
6 Y='\033[1;33m' # Amarillo
7 B='\033[1;34m' # Azul
8 M='\033[1;35m' # Mangenta
9 W='\033[0m' # Blanco
10 Glig='\e[1;32m' # Verde claro
11
12 #Para no salirse
13
14 ignore_signal() {
15     printf "Se ha detectado la se al $1. No se permite salir. Usa el comando '
16     salir' para cerrar la terminal.\n"
17 }
18
19 # Configurar la funci n ctrl + c/z
20 trap 'ignore_signal SIGINT' SIGINT
21 trap 'ignore_signal SIGTSTP' SIGTSTP
22
23 # Sistema de acceso para los usuarios
24 # ==--==--==-->
25
26 clear
27
28 printf "\t ===== \n"
29 printf "\t | |          HOLA BIENVENID@ A NUESTRA TERMINAL          | |\n"
30 printf "\t | |          MIRNA & HUGO                                | |\n"
31 printf "\t | |          p r o y e c t o      L I N U X              | |\n"
32 printf "\n\t\t\t---I N I C I O      D E      S E S I O N---\n"
33
34 # ==--==--==--> Solicitud de datos (interaccion con el usuario)
```




```
33 # solicitar el nombre de usuario y la contrase a
34
35
36 while true; do
37     printf "$G\n\t\t\t\t\tUsuario: $W"
38     read -p "" username
39     read -sp "Ingrese su contrase a: " password
40     # Verifica si el usuario y la contrase a son v lidos en el sistema
operativo anfitri n
41     if id -u "${username}" >/dev/null 2>&1 && echo "${password}" | su - "${
username}" -c 'exit' 2>/dev/null; then
42         echo -e "${B}Acceso concedido. Bienvenido ${username}${W}"
43         break
44     else
45         printf "\n\n$R Usuario o contrase a incorrectos. Intente nuevamente.$W\n
\n"
46     fi
47 done
48
49 while true; do
50     # Leer la entrada del usuario
51     printf "${M}${username}${W}:${G}$(pwd)${W} # "; read command
52     case $command in
53     salir)
54         exec ./salir.sh
55         break
56     ;;
57     ayuda)
58         ./ayuda.sh
59     ;;
60     infosis)
61         ./infosis.sh
62     ;;
```



```
63     infodia)
64         ./infodia.sh
65         ;;
66     MP3)
67         ./MP3.sh
68         ;;
69     buscar)
70         ./buscar.sh
71         ;;
72     creditos)
73         ./creditos.sh
74         ;;
75     gato)
76         ./gato.sh
77         ;;
78     *)
79         if command "$command" >/dev/null 2>&1; then
80             "$command"
81         else
82             printf "El comando $$command$W no existe. Utiliza ayuda para verificar los
comandos disponibles.\n"
83         fi
84         ;;
85     esac
86 done
```

```
=====
| |          HOLA BIENVENID@ A NUESTRA TERMINAL           | |
| |              MIRNA & HUGO                             | |
| |      p r o y e c t o       L I N U X                   | |
-----INICIO DE SESION-----

                        Usuario: hugo
Ingrese su contraseña: Acceso concedido. Bienvenido hugo
hugo:/home/hugo/Documents/ProyectoLinux # infodia
La fecha del dia de hoy es:
2023-04-22
La hora actual es:
13:27:20
hugo:/home/hugo/Documents/ProyectoLinux # ayuda
Los comandos disponibles son:
ayuda: Proporciona información acerca de los comandos disponibles
infosis: Proporciona información del sistema
infodia: Proporciona la fecha y hora actuales
busca: Busca un archivo en un directorio específico
creditos: Muestra los créditos del programa
MP3: Abrirá un reproductor de mp3
gato: Abrirá el juego gato
ahorcado: Abrirá el juego ahorcado
hugo:/home/hugo/Documents/ProyectoLinux # ^ZSe ha detectado la señal SIGTSTP. No se permite salir. Usa
el comando 'salir' para cerrar la terminal.
^C
El comando no existe. Utiliza ayuda para verificar los comandos disponibles.
hugo:/home/hugo/Documents/ProyectoLinux # ^ZSe ha detectado la señal SIGTSTP. No se permite salir. Usa
el comando 'salir' para cerrar la terminal.
^C
El comando no existe. Utiliza ayuda para verificar los comandos disponibles.
hugo:/home/hugo/Documents/ProyectoLinux # ^CShe ha detectado la señal SIGINT. No se permite salir. Usa
el comando 'salir' para cerrar la terminal.
```



4. Conclusion

- **Mirna:** Fue un proyecto un tanto complicado, ya que había conceptos los cuáles ya teníamos un tanto olvidados e igualmente funcionalidades que habíamos visto de manera superficial en nuestras clases y ahora tendríamos que utilizarlos a su 100 por ciento, era difícil tratar de mejor o modificar algo en nuestra terminal para hacerla diferente, más .^adornada.^eimplementar mejoras, pero como todo programador, mueves 1 cosa y fallan 20, es un proyecto ambicioso en donde tratamos de sacar el mejor provecho, aprender cosas nuevas y juntar todos los conocimientos para llegar a esto, algo tangible, algo grande, algo que es nuestro desde la raíz, por lo que lograrlo es gratificante para nosotros y ver por todo el camino que hemos pasado desde que somos prebes y empezamos a las pocas semanas por este curso y todo lo que hemos recorrido.
- **Hugo:** Este proyecto fue un reto para mí, debido a que varias de las cosas que pedía tenía que investigarlas, ya que si bien, se trataron algunos temas, algunas sentencias o archivos necesarios, era necesario documentarse para saber que archivo guarda específicamente cada cosa para poder imprimirlo en pantalla, si bien, una vez que lo vas empezando, se hace mas ameno, fue un proyecto que disfruté de realizar, ya que reafirmé mis conocimientos en linux, en bash y en latex, además, la satisfacción al ver que tu programa funciona como esperabas es grande.

Finalmente, es un buen ejercicio para reafirmar todos aquellos conocimientos y adquirir otros mas, además de práctica, por lo tanto, me siento satisfecho con el proyecto a entregar. :)