



Proyecto final GNU-LINUX

Reynoso Ortega Francisco Javier

Oscar Manuel Suaznavar Avrizu

Septiembre 2023

PROTECO

Linux

Contents

1. Objetivo:	2
2. Desarrollo	2
2.1. Login	2
2.2. Terminal prebe	4
2.3. Comando ayuda	6
2.4. Comando infosis	6
2.5. Comando fecha	9
2.6. Comando búsqueda	9
2.7. Comando Juegos	11
2.8. Comando Magica	12
2.9. Comando Piedra	15
2.10. Comando Creditos	17
2.11. Comando Musica	20
3. Conclusiones por buddy	24

Introducción

El proyecto "Terminal de trabajo PREBE" tiene como objetivo principal evaluar y poner a prueba los conocimientos adquiridos por el prebecario durante el curso de Linux. Además, busca fomentar su capacidad de investigación, creatividad y análisis en la resolución de problemas relacionados con sistemas operativos y programación.

Especificaciones

El proyecto se centra en el desarrollo de un programa que simule una terminal de trabajo. Esta terminal deberá proporcionar al usuario diversas funcionalidades, incluyendo la gestión de archivos, la ejecución de comandos, la consulta de información del sistema, la reproducción de música y la posibilidad de jugar un juego programado por el usuario. A continuación, se detallan las especificaciones del proyecto:

- Se implementará un sistema de acceso que requerirá que los usuarios ingresen un nombre de usuario y contraseña al ejecutar el programa. Estas credenciales deben ser válidas en el sistema operativo anfitrión, lo que garantizará que solo los usuarios autorizados puedan utilizar la terminal.
- El usuario interactuará con la terminal a través de una línea de comandos personalizada. Esta línea de comandos puede mostrar información adicional, pero debe incluir obligatoriamente:
 - El nombre de usuario activo.
 - La ubicación actual (carpeta) del usuario en el sistema.
- La terminal debe ser capaz de interpretar correctamente los comandos programados por el usuario, así como los comandos disponibles en el sistema operativo anfitrión.
- La única forma de salir de la terminal será utilizando el comando "salir", que será implementado por el usuario como parte del proyecto. No se permitirá el uso de los comandos Ctrl + C o Ctrl + Z para forzar el cierre del programa.

1. Objetivo:

Que el prebecario demuestre los conocimientos adquiridos durante el curso de Linux, así mismo, que ponga a prueba su capacidad de investigación, creatividad y análisis para poder cumplir con las especificaciones del proyecto.

2. Desarrollo

2.1. Login

En este comando lo que se hizo fue programar un título bonito, y después hicimos que se pidiera el usuario pero el usuario se busca en el sistema si no se encuentra aparece una leyenda que dice "no existe en el

sistema”, si existe entonces entra al usuario y te pide su contraseña y tienes 3 intentos un ejemplo de su funcionamiento entrando de forma correcta es el siguiente:

```
> ./TerminalPrebe.sh
Hola, por favor ingresa tu nombre de usuario:
frank
Contraseña:
Inicio de sesión exitoso para el usuario frank
BBBBBBBB  II  EEEEEEEE  NNN  NN  VV  VV  EEEEEEEE  NNN  NN  II  DDDD  0000  II
BB  BB  II  EE  NN N  NN  VV  VV  EE  NN N  NN  II  D  D  0  0  II
BBBBBB  II  EEEEEEEE  NN  N  NN  VV  VV  EEEEEEEE  NN  N  NN  II  D  D  0  0  II
BB  BB  II  EE  NN  N  NN  VV  VV  EE  NN  N  NN  II  D  D  0  0
BBBBBBBB  II  EEEEEEEE  NN  NNNN  VV  EEEEEEEE  NN  NNNN  II  DDDD  0000  0

Escribe 'Ayuda' para ver una lista de comandos especiales disponibles :)

Carpeta actual: /home/frank/Desktop/Terminal_Prebe

Bienvenido frank, ¿Qué deseas hacer?
```

Código

```
1  #!/bin/bash
2  trap ' ' 2 20
3
4  chmod +x ./*
5
6  echo "Hola, por favor ingresa tu nombre de usuario:"
7  read -r usuario
8  echo "Contraseña:"
9  read -sr contrasena
10
11  # Comprueba si el usuario existe en el sistema
12  if id "$usuario" &>/dev/null; then
13      # Intenta autenticar al usuario usando su contraseña
14      if su -c "exit" - "$usuario" <<< "$contrasena" 2>/dev/null; then
15          echo "Inicio de sesión exitoso para el usuario $usuario"
16
17          # Colores de texto
18          rojo="\033[31m"
19          verde="\033[32m"
20          amarillo="\033[33m"
21          azul="\033[34m"
22          magenta="\033[35m"
23          cian="\033[36m"
24          reset="\033[0m"
25          echo -e "${amarillo}BBBBBBBB  II  EEEEEEEE  NNN  NN  VV  VV  EEEEEEEE  NNN
↵  NN  II  DDDD  0000  II ${reset}"
```

```

26  echo -e "${azul}BB      BB      II EE      NN N      NN      VV      VV      EE      NN N      NN
    ↪ II      D      D      O      O II ${reset}"
27  echo -e "${verde}BBBBBB      II EEEEEEE      NN N      NN      VV      VV      EEEEEEE      NN N      NN
    ↪ II      D      D      O      O II ${reset}"
28  echo -e "${rojo}BB      BB      II EE      NN N      NN      VV VV      EE      NN N      NN
    ↪ II      D      D      O      O      ${reset}"
29  echo -e "${magenta}BBBBBBBB      II EEEEEEE      NN      NNNN      VV      EEEEEEE      NN      NNNN
    ↪ II      DDDD      OOOO      O ${reset}"
30

```

2.2. Terminal prebe

este fragmento de código es parte de un programa que simula una terminal de comandos en Linux. Permite al usuario ejecutar varios comandos y scripts, y proporciona una interfaz interactiva para interactuar con el sistema.

```

Escribe 'Ayuda' para ver una lista de comandos especiales disponibles :)

Carpeta actual: /home/frank/Desktop/Terminal_Prebe

Bienvenido frank, ¿Qué deseas hacer?

```

Código

```

1      carpeta_codigos=$(pwd)
2      echo
3      echo " Escribe 'Ayuda' para ver una lista de comandos especiales disponibles :)"
4      echo
5      echo
6
7      while true; do
8          # Obtener la carpeta actual
9          carpeta_actual=$(pwd)
10
11         echo
12         echo -e "${cian}Carpeta actual: $carpeta_actual ${reset}"
13         echo
14         echo -e "${cian}Bienvenido $usuario, ¿Qué deseas hacer? ${reset}"
15         # Leer la opción del usuario
16         read -r opcion

```

```
17
18     # Evaluar la opción
19     case $opcion in
20         "Ayuda")
21             source "$carpeta_codigos/ayuda.sh"
22             ;;
23
24         "infosis")
25             source "$carpeta_codigos/infosis.sh"
26
27             ;;
28
29         "time")
30             source "$carpeta_codigos/time.sh"
31             ;;
32
33         "find")
34             source "$carpeta_codigos/find.sh"
35             ;;
36
37         "creditos")
38             source "$carpeta_codigos/creditos.sh"
39             ;;
40
41         "juegos")
42             source "$carpeta_codigos/juegos.sh"
43             ;;
44
45         "musica")
46             source "$carpeta_codigos/Musica.sh"
47             ;;
48
49         "exit")
50             echo "Saliendo del sistema :("
51             exit 0
52             ;;
53
54         *)
55             eval "$opcion" || clear
56             ;;
57     esac
58
59     done
60 else
61     echo "Contraseña incorrecta para el usuario $usuario"
62 fi
63 else
64     echo "El usuario $usuario no existe en el sistema"
65 fi
66
```

```
67 trap - 2 20
```

2.3. Comando ayuda

Este comando contiene una descripción con echo de todos nuestros comandos.

```
Ayuda
Escribe 'Ayuda' para ver una lista de comandos disponibles :)
Escribe 'infosis' para ver la información del sistema :)
Escribe 'time' para ver la hora actual :)
Escribe 'find' para buscar un archivo en un directorio debes pasar 2 parametros:)
Escribe 'juegos' para ver nuestros juegos :)
Escribe 'creditos' para ver los créditos :)
Escribe 'musica' para ver los comandos de música :)
Escribe 'exit' para salir :(
```

Código

```
1  #!/bin/bash
2  trap ' ' 2 20
3
4      echo " Escribe 'Ayuda' para ver una lista de comandos disponibles :)"
5      echo " Escribe 'infosis' para ver la información del sistema :)"
6      echo " Escribe 'time' para ver la hora actual :)"
7      echo " Escribe 'find' para buscar un archivo en un directorio debes pasar 2 parametros:)"
8      echo " Escribe 'juegos' para ver nuestros juegos :)"
9      echo " Escribe 'creditos' para ver los créditos :)"
10     echo " Escribe 'musica' para ver los comandos de música :)"
11     echo " Escribe 'exit' para salir :("
12
13 trap - 2 20
14
```

2.4. Comando infosis

Este script proporciona información general sobre el sistema, incluyendo detalles sobre el hardware, el sistema operativo y el uso de recursos como la RAM y el espacio en disco. También incluye un encabezado ASCII decorativo para darle un toque visual.

```

Información del sistema :)
.... NO! ...
... MNO! ...
..... MNO!! ..... MNN00! ...
..... MMNO! ..... MNN00!! .
.... MNOONN00! MNNNNNNNNMPPPOII! MNN0!!!! .
... !O! NNO! MNNNNNNNNNNMPPPO00II!! NO! ....
..... ! MNNNNNNNNNNMPPPP0000III! ! ...
..... MNNNNNNNNNNMPPPP000000II!! .....
..... MNNMM000000PPPPPPPP0000MII! ...
..... MNNMM.. OPPMMP ..,OMI! ....
..... MNNM:: o.,OPMP,.o ::I!! ...
..... NNM:::.,,00PM!P,:::!! ....
.. MNNNNNN0000PM0!!IIPPO!!0! .....
... MNNNNNNNNO0:!!:!!IPPP00! ....
.. MNNNNNNOOMMNNIIPPP00!! .....
..... MNNONNNNNNNIIOO!.....
..... MN MNNNNNNIIOO! 00 .....
..... MNO! IiiiiiiiiiiiI 0000 .....
..... NNN.MNO! . 0!!!!!!!!!!0 . 00NO NO! .....
.... MNNNNNO! ...0000000000 . MNNNON!.....
..... MNNNNNO! .. PNNNNNNNN .. MNNNON!.....
..... 00! ..... ON! .....
.....

Nombre del equipo: frank-B0HK-WAX9X

Dirección IP: 192.168.1.73 172.17.0.1 2806:107e:1a:a5de:95a1:5161:bcb5:3ce1

Dirección MAC: 5c:3a:45:9d:66:db
02:42:32:a0:a5:a6

Fecha y hora: vie 22 sep 2023 21:07:13 CST

Versión del kernel: 6.2.0-33-generic

```


Código

```

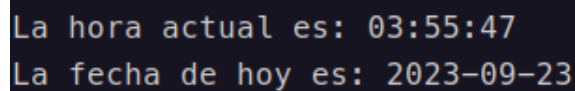
1  #!/bin/bash
2  trap ' ' 2 20
3
4  echo "Información del sistema :)"
5  echo ".... NO! ...          ... MNO! ...
6      .... MNO!! ..... MNNOO! ...
7      .... MMNO! ..... MNNOO!! .
8  .... MNOONNOO!  MMMMMMMMMPPPOII!  MNNO!!!! .
9  ... !O! NNO! MMMMMMMMMPPPOOOII!! NO! ....
10     ..... ! MMMMMMMMMPPPOOOIII! ! ...
11     ..... MMMMMMMMMPPPPPOOOOOII!! .....
12     ..... MMMMOOOOOOPPPPPPPPOOOOII! ...
13     ..... MMMM..  OPPMMP  .,OMI! ....
14     ..... MMMM::  o.,OPMP,.o  :I!! ...
15     .... NNM:::.,,OOPM!P,:::!! ....
16     .. MMNNNNNOOOOPMO!!IIPPO!!O! .....
17     ... MMMMMNNNOO:!!:!!IPPPPOO! ....
18     .. MMMMMNNOOMNNIIPPPPOO!! .....
19     ..... MMONNMNNNIIIOO!.....
20     ..... MN MOMMMNNNIIIOO! OO .....
21     ..... MNO! IiiiiiiiiiiiI OOOO .....
22     ..... NNN.MNO! . O!!!!!!O . OONO NO! .....
23     .... MNNNNNO! ...OOOOOOOOOO . MMNON!.....
24     ..... MNNNNO! .. PPPPPPPP .. MMNON!.....
25     ..... OO! ..... ON! .....
26     ..... "
27
28  echo
29  echo "Nombre del equipo: $(hostname)"
30  echo
31  echo "Dirección IP: $(hostname -I)"
32  echo
33  echo
34  echo "Dirección MAC: $(ip link show | awk '/ether/ {print $2}')"
35  echo
36  echo
37  echo "Fecha y hora: $(date)"
38  echo
39  echo
40  echo "Versión del kernel: $(uname -r)"
41  echo
42  echo
43  echo "Usuarios conectados: $(who -q)"
44  echo
45  echo
46  echo "Memoria RAM disponible: $(free -h)"
47  echo
48  echo
49  echo "Espacio en disco: $(df -h)"

```

```
50     echo
51     echo
52     # Obtener la arquitectura del sistema
53     arquitectura=$(arch)
54     echo "Arquitectura del sistema: $arquitectura"
55     echo
56     echo
57     # Obtener la versión del sistema operativo (requiere el paquete lsb-release)
58     if [ -x "$(command -v lsb_release)" ]; then
59         version_so=$(lsb_release -d | awk -F ":\\t" '{print $2}')
60         echo "Versión del sistema operativo: $version_so"
61     fi
62
63 trap - 2 20
```

2.5. Comando fecha

En resumen, este script muestra la hora y fecha actuales del sistema en la terminal y deshabilita la interrupción y suspensión del script mediante Ctrl+C y Ctrl+Z mientras se ejecuta. Después de mostrar la hora y fecha, se restauran las configuraciones de control de señales a su estado anterior.



```
La hora actual es: 03:55:47
La fecha de hoy es: 2023-09-23
```

Código

```
1  #!/bin/bash
2  trap ' ' 2 20
3
4
5  # Obtener la hora y fecha actual
6  grep "rtc_time" /proc/driver/rtc | awk '{print "La hora actual es: "$3}'
7  grep "rtc_date" /proc/driver/rtc | awk '{print "La fecha de hoy es: "$3}'
8
9  trap - 2 20
```

2.6. Comando búsqueda

Este script permite al usuario ingresar un directorio y un nombre de archivo, y luego verifica si el archivo con ese nombre existe en el directorio especificado o en el directorio actual si el primero no se encuentra. Luego, muestra un mensaje indicando si el archivo fue encontrado o no..

Bienvenido frank, ¿Qué deseas hacer?

find

Debes ingresar los parámetros de la siguiente manera: /ruta/a/mi/directorio miarchivo.txt

Ingresa el directorio: /home/frank/

Ingresa el nombre del archivo: D1.txt

El archivo D1.txt fue encontrado en el directorio /home/frank/.

Código

```
1  #!/bin/bash
2  trap ' ' 2 20
3
4  echo
5  echo "Debes ingresar los parámetros de la siguiente manera: /ruta/a/mi/directorio miarchivo.txt"
6  # Leer el directorio y el nombre del archivo
7  read -rp "Ingresa el directorio: " directorio
8  read -rp "Ingresa el nombre del archivo: " archivo_a_buscar
9
10 # Utiliza el comando find para buscar el archivo en el directorio especificado
11 #resultado=(find "directorio" -name "archivo_a_buscar")
12 echo
13 actual=$(pwd)
14 cd "$directorio" || echo no se encontró el directorio objetivo, se realizará la búsqueda en
↪ directorio actual
15 resultado="false"
16 for file in *; do
17     if [ "$file" == "$archivo_a_buscar" ]; then
18         resultado="true"
19     fi
20 done
21 echo
22 # Verifica si el archivo fue encontrado
23 if [ "$resultado" == "true" ]; then
24     echo "El archivo $archivo_a_buscar fue encontrado en el directorio $directorio."
25 else
26     echo "El archivo $archivo_a_buscar no fue encontrado en el directorio $directorio."
27 fi
28 cd "$actual" || return
29 echo
30
31 trap - 2 20
```

2.7. Comando Juegos

En resumen este comando nos da la oportunidad de elegir entre 2 juegos puede ser el Numero Magico o Piedra papel o tijeras

```

Bienvenido frank, ¿Qué deseas hacer?
juegos

      GAME

      ,....,
      ,:~::~<
      ,:~/^".
      ,:~/, ` e`.
      ,:~:|      \'.
      ,:~:| \____,~. c)
      ;:~:|      \ \  ' _'
      ;:~:|      \
      ;:~:|      _.=\
      `;:~:|.==` _.=\
      ' | _.=` _ _ \
      ` \ _..==` \ /
      . ' _ _ _ _ _ '
      /              \
jgs ('--.....--')
/'--.....--'\
`"--.....--"

Bienvenido a nuestros juegos :)
Escribe 'Magica' para jugar al numero magico :)
Escribe 'Piedra' para jugar a piedra papel o tijera :)
Escribe 'Salir' para salir de los juegos :(
Ingresa la palabra clave de la opción: 
```

Código

```

1  while true; do
2  echo "
3      ,....,
4      ,:~::~<
```

```

5      ,:~/^"\`~.
6      ,:~/, \` e\`.
7      ,:; | \`.
8      ,:| \_\_\_,-. c)
9      ;:| \\\` '-
10     ;:| \\\`
11     ;:| _.=\`\\
12     \`;|.=\` _.=\`\\
13     '|_.=\` _\\
14     \\\`_..==\`\\ /
15     .'.----.-'.
16     / \
17 jgs ('--.....--')
18     /'--.....--'\
19     \`\`"--.....--"\`\`"
20
21     echo "Bienvenido a nuestros juegos :)"
22     echo "Escribe 'Magica' para jugar al numero magico :)"
23     echo "Escribe 'Piedra' para jugar a piedra papel o tijera :)"
24     echo "Escribe 'Salir' para salir de los juegos :("
25     read -p "Ingresa la palabra clave de la opción: " opcion_juegos
26
27     case $opcion_juegos in
28

```

2.8. Comando Magica

En resumen este comando nos hará adivinar un número entre el 1 y el 100, en el cual te ira diciendo si el numero es mayor o menor dependiendo del numero que tu escojas

Código

```

1 case $opcion_juegos in
2
3 "Magica")
4
5     echo "Bienvenido al juego del numero Magico."
6     echo "
7
8         -----
9         |OFFo oON          |
10        | .----- .      |
11        | | .----- .    | |
12        | | |              | | |
13        | | |              | | |
14        | | |              | | |
15        | | |              | | |
16        | | |              | | |
17        | | |              | | |
18        | | |              | | |
19        | | | '-----'   | |
20        | |__GAME BOY_____/ |
21        |         _____|
22        |         (Nintendo)  |
23        | _| |_  \\"\\\\"\\\\"\\\\"\\\\"  .-.  |
24        |-[ _ ]-      .-. ( ) |
25        |  |_|      ( ) '- ' |
26        |   '        '- ' A  |
27        |           B        |
28        |           --- ---  |
29        |           (___) (___) ,., |
30        |           select start ;;; |
31        |           ,;;' /
32        jgs|           ,;;'.'
33        '-----'
34
35
36     # Genera un número aleatorio
37     numero=$((RANDOM % 100 + 1))
38
39     # Bucle de juego
40     for i in {1..10}; do
41         # Solicitar un número
42         echo "Adivina el número mágico (1-100): "
43         read numero_jugador
44
45         # Comprobar si el número es correcto
46         if [ "$numero_jugador" -eq "$numero" ]; then
47             echo "¡Ganaste!"
48             break
49         elif [ "$numero_jugador" -lt "$numero" ]; then

```

```
50         echo "El número es mayor"
51     else
52         echo "El número es menor"
53     fi
54 done
55
56     # Mostrar el resultado
57     if [ "$numero_jugador" -ne "$numero" ]; then
58         echo "¡Perdiste!"
59     fi
60
61     ;;
62
```

2.9. Comando Piedra

Este comando nos permite jugar al piedra papel o tijeras contra la computadora

```
Bienvenido a nuestros juegos :)
Escribe 'Magica' para jugar al numero magico :)
Escribe 'Piedra' para jugar a piedra papel o tijera :)
Escribe 'Salir' para salir de los juegos :(
Ingresa la palabra clave de la opción: Piedra
Juguemos a Piedra, Papel o Tijera.
Elige una opción:
1. Piedra
2. Papel
3. Tijera
4. Salir
Ingresa el número de tu elección: 1
Jugador: Piedra
Computadora: Piedra
Resultado: Empate
Juguemos a Piedra, Papel o Tijera.
Elige una opción:
1. Piedra
2. Papel
3. Tijera
4. Salir
Ingresa el número de tu elección: 
```

Código


```
1  "Piedra")
2      # Código del juego de Piedra, Papel o Tijera aquí
3      verde="\033[32m"
4      while true; do
5          echo -e "${verde}Juguemos a Piedra, Papel o Tijera.${reset}"
6          echo "Elige una opción:"
7          echo "1. Piedra"
8          echo "2. Papel"
9          echo "3. Tijera"
10         echo "4. Salir"
11
12         read -p "Ingresa el número de tu elección: " eleccion
13
14         # Genera una elección aleatoria para la computadora (1 = Piedra, 2 = Papel, 3 =
15         ↪ Tijera)
16         computadora=$((RANDOM % 3 + 1))
17
18         case $eleccion in
19             1)
20                 jugador="Piedra"
21                 ;;
22             2)
23                 jugador="Papel"
24                 ;;
25             3)
26                 jugador="Tijera"
27                 ;;
28             4)
29                 echo "Gracias por jugar. ¡Hasta luego!"
30                 return 0
31                 ;;
32             *)
33                 echo "Opción no válida. Por favor, elige un número del 1 al 4."
34                 continue
35                 ;;
36         esac
37
38         case $computadora in
39             1)
40                 comp="Piedra"
41                 ;;
42             2)
43                 comp="Papel"
44                 ;;
45             3)
46                 comp="Tijera"
47                 ;;
48         esac
49
50         echo "Jugador: $jugador"
```

```
50     echo "Computadora: $comp"
51
52     # Determina el resultado
53     if [ "$jugador" == "$comp" ]; then
54         resultado="Empate"
55     elif [ "$jugador" == "Piedra" ] && [ "$comp" == "Tijera" ]; then
56         resultado="¡Ganaste!"
57     elif [ "$jugador" == "Papel" ] && [ "$comp" == "Piedra" ]; then
58         resultado="¡Ganaste!"
59     elif [ "$jugador" == "Tijera" ] && [ "$comp" == "Papel" ]; then
60         resultado="¡Ganaste!"
61     else
62         resultado="¡La computadora gana!"
63     fi
64
65     echo "Resultado: $resultado"
66 done
67 ;;
68
69 "Salir")
70     echo "Saliendo de los juegos."
71     return 0
72     ;;
73
74 *)
75     echo "Opción de juegos no válida :("
76     ;;
77 esac
78 done
```

2.10. Comando Credits

Este comando lo que hace es darle credits a los programadores de la terminal, mediante un estilo grafico

Créditos del Programador

¡Gracias por usar nuestro programa!

```

_____/%%*%%*%%*%%*%%*%%*%%\_____/
//~~~~~                               ~~~~~:I
I:      ,;;;;;;;;,      |      O F F I C I A L      :I
I:      /:/:/:/\;\      |      ID DEVELOPERS      :I
I:      //:/:-'\;\      |      <-><-><-><-><-><-><-><-><-><-> :I
I:      ;:/ _ _ \;|;      |      :I
I:      ||; | | ||      | Name: Frank Ortega      :I
I:      |;| ^ ;||      | Email: francisco.reynoso2000@gmail.com :I
I:      ||;\ '-=-' /|;|      | GitHub: https://github.com/FranciscoJRO :I
I:      \ |;|.____.;|;/      |      :I
I:      =)      (=      |      :I
I:      .-:\,      ,\;-      |      :I
I:      /\      \';@;' /      \      |      :I
I:      /      '-,-' \      |      :I
*jgs_____ *
\%%*%%*%%*%%*%%*%%*%%/

```

```

_____/%%*%%*%%*%%*%%*%%*%%\_____/
//~~~~~                               ~~~~~:I
I:      ,;;;;;;;;,      |      O F F I C I A L      :I
I:      /:/:/:/\;\      |      ID DEVELOPERS      :I
I:      //:/:-'\;\      |      <-><-><-><-><-><-><-><-><-><-> :I
I:      ;:/ _ _ \;|;      |      :I
I:      ||; | | ||      | Name: Oscar Suaz      :I
I:      |;| ^ ;||      | Email: omanuel_suaznavar@hotmail.com :I
I:      ||;\ '-=-' /|;|      | GitHub: https://github.com/OscarSuaz :I
I:      \ |;|.____.;|;/      |      :I
I:      =)      (=      |      :I
I:      .-:\,      ,\;-      |      :I
I:      /\      \';@;' /      \      |      :I
I:      /      '-,-' \      |      :I
*jgs_____ *

```

Código

```

1      #!/bin/bash
2      trap ' ' 2 20
3
4
5
6      verde="\033[32m"
7      reset="\033[0m"
8      echo -e "${verde}Créditos del Programador"

```

```

9
10 echo
11 echo "¡Gracias por usar nuestro programa!"
12
13
14 echo "
15 echo "      _____/%%*%%*%%*%%*%%*%%*\_____
16 echo "      //~~~~~:I"
17 echo "      I:      ,; ; ; ; ; ,      |      O F F I C I A L      :I"
18 echo "      I:      /; ; /; / \ \; \      |      ID DEVELOPERS      :I"
19 echo "      I:      //; /// - ' \ \; \      |      <-><-><-><-><-><-><-><-><-><-> :I"
20 echo "      I:      ;;/ _ _ _ \; |;      |      :I"
21 echo "      I:      ||; | |      |||      |      Name:  Frank Ortega      :I"
22 echo "      I:      |; |      ^      ;||      |      Email:  francisco.reynoso2000@gmail.com      :I"
23 echo "      I:      ||; \ '-=-' / |; |      |      GitHub: https://github.com/FranciscoJRO      :I"
24 echo "      I:      \ |; ; . _ _ _ . ; |; /      |      :I"
25 echo "      I:      =)      (=      |      :I"
26 echo "      I:      .-:\,      ,\;- .      |      :I"
27 echo "      I:      /\      \'; @; '/      \      |      :I"
28 echo "      I:      /      '-.-'      \      |      :I"
29 echo "      *jgs_____|_____*"
30 echo "      \%%*%%*%%*%%*%%*%%*\%%*%%*%%*%%*%%*\/"
31 echo
32 echo
33 echo
34 echo "
35 echo "      _____/%%*%%*%%*%%*%%*%%*\_____
36 echo "      //~~~~~:I"
37 echo "      I:      ,; ; ; ; ; ,      |      O F F I C I A L      :I"
38 echo "      I:      /; ; /; / \ \; \      |      ID DEVELOPERS      :I"
39 echo "      I:      //; /// - ' \ \; \      |      <-><-><-><-><-><-><-><-><-><-> :I"
40 echo "      I:      ;;/ _ _ _ \; |;      |      :I"
41 echo "      I:      ||; | |      |||      |      Name:  Oscar Suaz      :I"
42 echo "      I:      |; |      ^      ;||      |      Email:  omanuel_suaznavar@hotmail.com      :I"
43 echo "      I:      ||; \ '-=-' / |; |      |      GitHub: https://github.com/OscarSuaz      :I"
44 echo "      I:      \ |; ; . _ _ _ . ; |; /      |      :I"
45 echo "      I:      =)      (=      |      :I"
46 echo "      I:      .-:\,      ,\;- .      |      :I"
47 echo "      I:      /\      \'; @; '/      \      |      :I"
48 echo "      I:      /      '-.-'      \      |      :I"
49 echo "      *jgs_____|_____*"
50 echo "      \%%*%%*%%*%%*%%*%%*\%%*%%*%%*%%*%%*\/"
51 trap - 2 20
52
53
54

```

2.11. Comando Musica

este script es un reproductor de música básico en la línea de comandos que permite al usuario reproducir canciones individuales, listas de reproducción y cambiar de directorio para reproducir música. También proporciona controles para pausar, ajustar el volumen y otras funciones relacionadas con la reproducción de música.

```
Carpeta actual: /home/frank/Desktop/Terminal_Prebe

Bienvenido frank, ¿Qué deseas hacer?
musica

/home/frank/Desktop/Terminal_Prebe/Musica.sh: line 133: cd: /home/frank/Música: No such file or directory

Las siguientes opciones son todas en base a tu directorio actual
1) Reproducir una canción
2) Reproducir una carpeta especifica de canciones (que se encuentre en la carpeta actual)
3) Reproducir todas las canciones sueltas de carpeta actual
4) Cambiar de directorio donde reproducir musica
5) Salir
Carpeta actual: /home/frank/Music
¿Que desea hacer?
Opción: 

/home/frank/Desktop/Terminal_Prebe/Musica.sh: line 133: cd: /home/frank/Música: No such file or directory

Las siguientes opciones son todas en base a tu directorio actual
1) Reproducir una canción
2) Reproducir una carpeta especifica de canciones (que se encuentre en la carpeta actual)
3) Reproducir todas las canciones sueltas de carpeta actual
4) Cambiar de directorio donde reproducir musica
5) Salir
Carpeta actual: /home/frank/Music
¿Que desea hacer?
Opción: 1
Nombre de canción incluyendo la extensión (.mp3): Tu Falta De Querer.mp3
*-----*
                CONTROLES DEL REPRODUCTOR
*-----*
Pausar/Reanudar:      s
Volumen:              + o -
Repetir canción:      b
Salir:                q
*-----*
```

Código

```

1      #!/bin/bash
2  trap ' ' 2 20
3
4  # Colores de texto
5  rojo="\033[31m"
6  verde="\033[32m"
7  amarillo="\033[33m"
8  azul="\033[34m"
9  magenta="\033[35m"
10 cian="\033[36m"
11 reset="\033[0m"
12
13 # Función que imprime el título del programa
14 title(){
15     echo -e "${rojo}                ${reset}"
16     echo -e "${verde}                ${reset}"
17     echo -e "${amarillo}            ${reset}"
18     echo -e "${azul}                ${reset}"
19     echo -e "${magenta}            ${reset}"
20     echo -e "${cian}                ${reset}"
21 }
22
23 #Valida si está instalado el reproductor, si no lo está, da opción de instalarlo
24 validampg(){
25     reproductor=$(whereis mpg123)
26     if [[ "$reproductor" == "mpg123:" ]]; then
27         echo "No se encontró el programa de reproducción de musica, desea instalarlo? [Y/n]"
28         read -r opcion
29         if [[ $opcion =~ ^ (Yes|Y|y|) $ ]]; then
30             sudo apt install mpg123
31         else
32             echo "No se instalará el reproductor de música"
33             exit
34         fi
35     fi
36 }
37
38 #Display del menú de opciones basicas/iniciales del reproductor
39 menuBasic(){
40     echo "Las siguientes opciones son todas en base a tu directorio actual"
41     echo "1) Reproducir una canción"
42     echo "2) Reproducir una carpeta especifica de canciones (que se encuentre en la carpeta actual)"
43     echo "3) Reproducir todas las canciones sueltas de carpeta actual"
44     echo "4) Cambiar de directorio donde reproducir musica"
45     echo "5) Salir"
46 }
47
48 #imprime en pantalla las acciones de utilidad para el reproductor dependiendo del argumento
49 #1 para solo 1 canción, 2 para una lista/carpeta de canciones
50 controlador(){

```

```

50  if [[ "$1" == "1" ]]; then
51      echo -e "${rojo}*-----*${reset}"
52      echo -e "${azul}                CONTROLES DEL REPRODUCTOR${reset}"
53      echo -e "${rojo}*-----*${reset}"
54      echo -e "${azul}  Pausar/Reanudar:          s${reset}"
55      echo -e "${azul}  Volumen:                + o -${reset}"
56      echo -e "${azul}  Repetir canción:         b${reset}"
57      echo -e "${azul}  Salir:                  q${reset}"
58      echo -e "${rojo}*-----*${reset}"
59  else
60      echo -e "${rojo}*-----*${reset}"
61      echo -e "${azul}                CONTROLES DEL REPRODUCTOR${reset}"
62      echo -e "${rojo}*-----*${reset}"
63      echo -e "${azul}  Pausar/Reanudar:          s${reset}"
64      echo -e "${azul}  Volumen:                + o -${reset}"
65      echo -e "${azul}  Mostrar playlist         l${reset}"
66      echo -e "${azul}  Siguiente canción:       f${reset}"
67      echo -e "${azul}  Canción anterior:        d${reset}"
68      echo -e "${azul}  Repetir canción:         b${reset}"
69      echo -e "${azul}  Repetir playlist         [${reset}"
70      echo -e "${azul}  Salir:                   q${reset}"
71      echo -e "${rojo}*-----*${reset}"
72  fi
73 }
74
75 #Reproduce una canción
76 unaCancion(){
77     read -rp "Nombre de canción incluyendo la extensión (.mp3): " song
78     song=$(echo "$song" | tr " " "\\ ") #Si el texto de la canción contiene espacios, reemplaza los
79     ↪ espacios por "\ "
80     controlador "1"
81     mpg123 -C --title -q "$song"
82 }
83
84 #Reproduce una lista de canciones dentro del directorio actual
85 playlist(){
86     echo "En caso de no poder acceder a la carpeta, se regresará al menú principal"
87     read -rp "Escriba el nombre de la carpeta con las canciones: " nom
88     #nom=$(echo "$nom" | tr " " "\\ ")
89     cd "$nom" || return
90     cd ..
91     echo "La reproducción por defecto es aleatoria, desea que se mantenga así?"
92     read -rp "[Y/n]" rand
93     controlador "2"
94     if [[ $rand =~ ^ (Yes|Y|y)$ ]]; then
95         mpg123 -C --title -q -z "$nom"/*.mp3
96     else
97         mpg123 -C --title -q "$nom"/*.mp3
98     fi
99 }

```

```

100 }
101
102 cambiarCarp(){
103     read -rp "Escribe la ruta absoluta del directorio donde se encuentran tus canciones" ruta
104     cd "$ruta" || echo "No se pudo cambiar de directorio, nos quedamos en el directorio actual"
105 }
106
107 musicActual(){
108     len=$(ls *.mp3 | wc -l)
109     #echo "len"
110     #return
111     if [ "$len" = 1 ] ; then
112         controlador "1"
113         mpg123 -C --title -q ./mp3
114
115     elif [ "$len" -ge 2 ] ; then
116         echo "La reproducción por defecto es aleatoria, desea que se mantenga así?"
117         read -rp "[Y/n]" rand
118         controlador "2"
119         if [[ $rand =~ ^((Yes|Y|y|)|$) ]]; then
120             mpg123 -C --title -q -z ./mp3
121         else
122             mpg123 -C --title -q ./mp3
123         fi
124     else
125         echo no hay canciones en la carpeta
126         return
127     fi
128 }
129
130 Reproductor(){
131     title
132     validampg
133     cd "$HOME/Música" || cd "$HOME/Music" || echo no se pudo encontrar una carpeta de musica se
134     ↪ utilizará la actual.
135
136     echo
137     echo
138     cic=0
139     while cic=0 ; do
140         carpeta_actual=$(pwd)
141         menuBasic
142         echo -e "${cyan}Carpeta actual: $carpeta_actual ${reset}"
143         echo -e "${cyan}¿Que desea hacer?${reset}"
144         read -rp "Opción: " accion
145         case $accion in
146             "1")
147                 unaCancion
148                 clear
149                 ;;
150             "2")

```



```
150         playlist
151         clear
152         ;;
153         "3")
154         musicActual
155         ;;
156         "4")
157         cambiarCarp
158         clear
159         ;;
160         "5")
161         cic=$cic+1
162         return
163         ;;
164         *)
165         echo "Opción no valida"
166     esac
167 done
168 }
169
170
171 Reproductor
172
173 trap - 2 20
```

3. Conclusiones por buddy

Reynoso Ortega Francisco Javier:

Durante la realización de este proyecto de la "Terminal de Trabajo PREBE", he experimentado un crecimiento sustancial en mis habilidades técnicas y en mi comprensión de los sistemas Linux. Me siento mucho más competente en la administración de sistemas y la programación en Bash. La creación de scripts personalizados y la implementación de funciones únicas me han dado confianza en mi capacidad para automatizar tareas y resolver problemas de manera eficiente. Además, la experiencia de diseñar un sistema de autenticación y trabajar en la seguridad de los sistemas Unix me ha brindado una comprensión más profunda de la importancia de la seguridad en la informática. En general, este proyecto ha sido una experiencia enriquecedora que ha fortalecido mi conjunto de habilidades y mi confianza en mis capacidades técnicas.

Suaznavar Arvizu Oscar Manuel:

Este proyecto de la "Terminal de Trabajo PREBE" ha sido una experiencia de aprendizaje excepcional. Me ha permitido sumergirme en el mundo de la administración de sistemas Linux y la programación en Bash de una manera práctica y desafiante. A medida que desarrollaba comandos personalizados y funciones, he mejorado mi habilidad para resolver problemas y automatizar tareas, lo que considero invaluable en mi trayectoria profesional. La seguridad y la autenticación se convirtieron en temas clave que abordé con éxito, lo que me hizo comprender la importancia de la seguridad en sistemas Unix. En resumen, este proyecto me ha brindado conocimientos técnicos sólidos y una mayor confianza en mis habilidades informáticas, y estoy emocionado por las oportunidades futuras que se presentarán gracias a esta experiencia.