Práctica 1 de Sistemas Operativos

Raúl Zambrano Maestre

I.T. Informática de Gestión
Convocatoria de Junio

ÍNDICE

MEMORIA EXPLICATIVA DE LA PRÁCTICA

La práctica consiste en un pequeño interprete de comandos, desarrollado en lenguaje shell y lenguaje C, sobre sistema operativo GNU/Linux que realice las funciones especificadas.

Los usuarios que pueden usar el interprete mini-shell están almacenados en el fichero fusuarios, donde se guardan el nombre de usuario y la contraseña. Este fichero se encuentra en el directorio de configuración cfg.
Los datos proporcionados por el usuario son comprobados por el interprete y se permitirá o no entrar al mini-shell.

El interprete, reconoce las órdenes que se indican en el fichero fordenes que se encuentra almacenado en el directorio cfg. Este fichero contiene todos los comandos que el usuario puede ejecutar, así como la fecha y hora del último acceso.

Cada orden ejecutada, correcta o incorrectamente es almacenada en un fichero histórico, denominado fhistorico y que almacena como máximo 50 órdenes, si el número de órdenes almacenadas supera este límite se borra la orden más antigua, para que se pueda almacenar una más reciente.

Las órdenes que permite ejecutar el interprete de comandos son:

arbol_directorio: Esta orden permite mostrar el contenido completo del directorio que recibe como parámetro, incluyendo los subdirectorios que contenga.

cambia_clave: Permite cambiar la palabra clave del usuario que se especifique como parámetro. La nueva clave se pedirá por duplicado para evitar confusiones en su escritura.

historico_ordenes: Mostrará tantas líneas del fichero histórico como se especifiquen por parámetro. Las órdenes que muestra son las ejecutadas más recientemente.

ejecutar-temporizador: Esta orden permite ejecutar un temporizador con la configuración pasada como parámetro en un nuevo terminal xterm, mientras que en el terminal anterior se pueden seguir ejecutando otras órdenes. Se deberá especificar:
Intervalo: el número de segundos que transcurrirá entre cada ocurrencia de ejecución del temporizador.
Ocurrencias: Nº de veces que se va a ejecutar el temporizador antes de pasar a la ejecución de otro.
Acción: Tarea que va a realizar, valores numéricos del 1 al 5 que representan:

1. Mostrar los grupos de usuarios existentes en el sistema, debidamente formateados y los usuarios que pertenecen a dicho grupo(fichero /etc/group).

2. Localizar un fichero pasado como parámetro y mostrar las ubicaciones donde se ha encontrado (Comando locate).
3. Mostrar si un fichero pasado como primer parámetro existe en un directorio pasado como segundo parámetro.
4. Ordenación de todos y cada uno de los ficheros enviados como parámetros, hasta un máximo de 3, almacenando la versión ordenada en un directorio denominado “version_ordenada”.
5. Comprobar si en el fichero de usuarios del sistema (/etc/passwd), existen los usuarios que se pasen como parámetros (como máximo 3).

Iniciar : Orden que permitirá iniciar el minishell con un usuario distinto al actual, realizando todas las comprobaciones de error necesarias.

Salir: Del programa mini-shell se saldrá al ejecutar la orden salir, permitiendo devolver el control al shell del sistema.

CÓDIGO FUENTE COMENTADO

SHELL SCRIPT MINI-SHELL

```
Comprobar_directorio()
{
      if (test -d cfg)                    # Comprobamos que existe el directorio
            then  echo "Existe el directorio cfg."
            else
                  echo "Faltan parámetros de configuración."
                  echo "No existe el directorio cfg..."
                  exit
      fi
}

Comprobar_ficheros()
{
      if ((test -s cfg/fusuarios)&&(test -s cfg/fordenes))
      # Comprobamos que existen los ficheros
            then  echo "Ficheros correctos."
            else  echo "Faltan parámetros de configuración..."
                  echo "No existen los ficheros de configuración..."
                  exit
      fi
}

Peticion_login()
{
echo " "
encontrado=0
cont=3
while (test $encontrado -eq 0)# Mientras que no se encuentren login y password
do
      echo -n "Introduzca el nombre de usuario: "    # Solicitamos el usuario
      read login
      if (`grep -q "^$login" cfg/fusuarios`)         # Verificamos el login
            then  echo -n "Introduzca la contraseña: "
            # Si es correcto, pedimos password
                  read password
                  if (`grep -q "^$login:$password" cfg/fusuarios`)
                  # Verificamos password
                        then  encontrado=1      # Si coinciden, se ha encontrado
                        else  echo "La contraseña no es correcta."
                              let cont=$cont-1  # Decrementamos el contador
                              echo "Intentos restantes: "$cont
                              # Mostramos los intentos restantes
                  fi

            else  echo "El usuario $login no existe"
                  let cont=$cont-1
                  echo "Intentos restantes: "$cont
                  # Mostramos los intentos restantes
      fi

      if (test $cont -le 0)                     # si cont<=0 fin
            then  echo "Nº de intentos superado. Fin del programa"
                  encontrado=1
                  exit
      fi
```

```
done
}

Iniciar()
{
      if (test -z $1)                    # Comprobamos que se pasa un usuario
            then  echo "No ha introducido ningún nombre de usuario"
            else  if (`grep -q "^$1" cfg/fusuarios`)
                        then  echo -n "Introduzca la contraseña: "
                              read contra
                              if (`grep -q "^$1:$contra" cfg/fusuarios`)
                                    then  login=$1
                                    else  echo "Contraseña incorrecta"
                              fi

                        else  echo "El usuario $1 no existe"
                  fi
      fi
}

Arbol_directorio()
{
      if (test -d $1)         # Si el parámetro pasado es un directorio
            then  cd $1       # Ingresamos en el directorio
                  if (test -z $1)   # Si es un directorio vacío
                        then  echo "- DIRECTORIO: $ruta/$i VACÍO"

                        else  for i in `ls` # I toma los valores del commando ls
                              do
                                          if (test -d $i)   # Si es un directorio
                                          then  ruta=`pwd`  # ruta=dir. actual
                                                echo "- DIRECTORIO: $ruta/$i"
                                                Arbol_directorio $i
                                                # Llamada recursiva a la función

                                          else  ruta=`pwd`  # ruta=dir. actual
                                                echo "- FICHERO: $ruta/$i"
                                    fi
                              done
                  fi
                  cd ..       # Salimos del directorio
                  echo " "

            else  echo "ERROR. El directorio no existe."
      fi
}

Cambio()
{
      if (test -z $1)
            then  echo "Error, no ha escrito ningún nombre de usuario."

            else  if (`grep -q "^$1" cfg/fusuarios`)
                  then  echo -n "Introduzca la contraseña que desea cambiar: "
                              read clave
                              if (`grep -q "^$1:$clave" cfg/fusuarios`)
                                    then  echo -n "Introduzca la nueva: "
                                          read nuevo
                                          echo -n "Vuelva a teclear la nueva: "
                                          read renuevo
                                          if (test $nuevo = $renuevo)
                                          # Si coinciden ambas cadenas
                                                then  echo "Cambiando clave...."
```

```
                              grep -v "^$1:$clave" cfg/fusuarios > temp_dat
                              # Selecciona todas las líneas menos la indicada
                              cp temp_dat cfg/fusuarios
                              # Copia las líneas
                              rm temp_dat
                              echo "$1:$nuevo" >> cfg/fusuarios
                              else  echo "Error al reescribir la nueva."
                        fi

                                    else  echo "Contraseña incorrecta."
                              fi

                        else  echo "El usuario $1 no existe."
                  fi
      fi
}

Ejecutar()
{
      gcc -o temporizador temporiza.c     # Compila el fichero del temporizador
      xterm -e ./temporizador $var1 $var2 $var3 $var4 $var5 $var6 &
      # Ejecuta en backgound en un terminal, el ejecutable del temporizador
}

Peticion_orden()
{
      echo -n "$login@minishell$"              # Pinta el prompt del mini shell
      read orden var1 var2 var3 var4 var5 var6 # Lee el comando y los parámetros
      fecha=`date +'%d/%m/%Y-%H.%M'`           # Guarda la fecha en que se usó

      if (test -s fhistorial)                  # Si existe el fichero fhistorial
            then  if(test `grep -c "." fhistorial` -lt 50) #Si no hay 50 órdenes
                        then  echo "$orden" >> fhistorial  #Inserta la orden
                        else  tail -n 49 fhistorial >> temp_dat
                              # Almacena las últimas 49 órdenes temporalmente
                              cp temp_dat fhistorial
                              # Copia las 49 útimas órdenes en fhistorial
                              rm temp_dat
                              # Borra el fichero temporal
                              echo "$orden" >> fhistorial
                              # Inserta la última orden ejecutada
                  fi

            else  echo "$orden" > fhistorial
      fi


      if(`grep -q "^$orden:" cfg/fordenes`)
      # Si existe la orden en el fichero fordenes
            then  grep -v "^$orden:" cfg/fordenes > temp_dat
            # Seleccionamos las otras líneas y las guardamos temporalmente
                  cp temp_dat cfg/fordenes
                  # Las copiamos al fichero de órdenes
                  rm temp_dat
                  # Borramos el temporal
                  echo "$orden:$fecha:" >> cfg/fordenes
                  # Insertamos la orden con la fecha actual
                  case $orden in
                  # Dependiendo del comando insertado realizamos lo siguiente:
```

```
                        "arbol_directorio")
                              directorio=`pwd`
                              echo "LISTADO: "
                              echo " "
                              Arbol_directorio $var1
                              cd $directorio
                        ;;

                        "cambia_clave")
                              Cambio $var1
                        ;;

                        "historico_ordenes")
                              if (test -z $var1)
                                    then var1=10
                              fi
                              tail -n $var1 fhistorial
                        ;;

                        "ejecutar_temporizador")
                              Ejecutar $var1 $var2 $var3 $var4 $var5 $var6
                        ;;

                        "iniciar")
                              Iniciar $var1
                        ;;

                        "salir")
                              exit
                        ;;

                  esac

            else  echo "ERROR. Orden no encontrada."
      fi

      echo " "

      Peticion_orden
}


Comprobar_directorio    # Comprobamos que están los directorios
Comprobar_ficheros      # Comprobamos que están los ficheros
Peticion_login          # Pedimos el login/password
Peticion_orden          # Pedimos la ejecución de un comando
```

TEMPORIZA.C

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>
#include <signal.h>
#include <time.h>
#include <sys/time.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>
#include <dirent.h>

int interv=0;
int condicion=0;
int tarea=0;

char caso1[75]="cut -f1,4 -d\":\" /etc/group > fgrupos; cat fgrupos; rm
fgrupos;\0";
char caso2a[10]="locate\0";
char caso3a[14]="if (test -s \0";
char caso3b[3]="/";
char caso3c[90]="); then echo \"El fichero si existe.\"; else echo \"El fichero
no existe.\"; fi\0";
char caso4a[10]="sort \0";
char caso4b[25]=" > version_ordenada/\0";
char caso4c[30]="; echo \"Creando fichero\";\0";
char caso5a[14]="usuarios=\"\0";
char caso5b[120]="\"; for j in $usuarios; do if (grep -q \"^$j\" /etc/passwd);
then echo \"$j existe\"; else echo \"$j no existe\"; fi; done\0";

char comando[250]="";         // Almacena los comandos que ejecutará el S.O.
char direcc[30]="\0";         // Se almacena una ruta para hacer strtok
                              // y sacar el nombre del fichero
char espacio[3]=" \0";        // Cadena con un espacio
char delimitador[3]="/\0";    // Delimitador de los campos de una dir. o ruta

int correcto=1;               // Correcto=1 -> Se pueden abrir los ficheros

FILE *fichero;                // Puntero a los ficheros que se abrirán
char *ptr,*nombrefiche;       // Punteros usados en strtok

struct sigaction accion;
struct itimerval valortemporizador;

static void orden()
{
      system(comando);
      condicion--;
}

void inicializa_senial()           // Armado de señal
{
      accion.sa_handler=orden;
      accion.sa_flags=SA_RESTART;
      sigemptyset(&accion.sa_mask);
      sigaction(SIGPROF, &accion,NULL);
}
```

```
void inicializa_intervalo(int intervalo) // Inicializa el temporizador
{
      valortemporizador.it_interval.tv_sec=intervalo;
      valortemporizador.it_interval.tv_usec=0;
      valortemporizador.it_value=valortemporizador.it_interval;
      setitimer(ITIMER_PROF, &valortemporizador, NULL);
}

int main(int argc, char *argv[])
{
      int i,j;
        if (argc>=4)
      {
            interv=atoi(argv[1]);
            condicion=atoi(argv[2]);
            tarea=atoi(argv[3]);

            switch (tarea)
                  {
                        case 1:
                              if (argc==4)      strcat(comando,caso1);
                              else
                                    printf("Error, esta operación no necesita
                                    parámetros\n");
                              break;

                        case 2:
                              if (argc==5)
                                {
                                    strcat(comando,caso2a);
                                    strcat(comando,espacio);
                                    strcat(comando,argv[4]);
                                }

                              else  printf("Error,nº de parámetros incorrecto");
                              break;

                        case 3:
                              if (argc==6)
                                {
                                    strcat(comando,caso3a);
                                    strcat(comando,argv[5]);
                                    strcat(comando,caso3b);
                                    strcat(comando,argv[4]);
                                    strcat(comando,caso3c);

                                }

                              else  printf("Error,nº parámetros incorrecto\n");
                              break;

                        case 4:
                              if ((argc>4)&&(argc<8))
                                  {
                                    for(i=4; i<argc; i++)
                                          {
                                                fichero=fopen(argv[i],"rb");
                                                if (!fichero){
                                                printf("Error, al abrir.\n");
                                                correcto=0;
```

```
                                        break;
                                        }
                                        fclose(fichero);
                                    }

                                if (correcto==1)
                                        {
                                           for (j=4;j<argc;j++)
                                              {
                                              strcpy(direcc,argv[j]);
                                              ptr=strtok(direcc,delimitador);
                                    while ((ptr=strtok(NULL,delimitador)) != NULL)
                                              nombrefiche=ptr;
                                              strcat(comando,caso4a);
                                              strcat(comando,argv[j]);
                                              strcat(comando,caso4b);
                                              strcat(comando,nombrefiche);
                                              strcat(comando,caso4c);
                                              }
                                        }
                                    }

                                    else  printf("Error, parámetros incorrectos.\n");
                                    break;


                              case 5:
                                    if ((argc>4)&&(argc<8))
                                        {
                                          strcat(comando,caso5a);
                                          strcat(comando,argv[4]);
                                          if (argc>=6){
                                                      strcat(comando,espacio);
                                                      strcat(comando,argv[5]);
                                                      }
                                          if (argc==7){
                                                      strcat(comando,espacio);
                                                      strcat(comando,argv[6]);
                                                      }
                                          strcat(comando,caso5b);
                                         }

                                    else  printf("Error, nº de nombres incorrecto\n");

                                    break;

                              default:
                                    write(2,"ERROR, LA ORDEN NO EXISTE\n");
                                    break;
                           }

                     inicializa_senial();
                     inicializa_intervalo(interv);

                     while(condicion>0);
               }

               else printf("ERROR, faltan parámetros para ejecutar el temporizador.\n");
               getchar();        // Mantiene la ventana de xterm a la espera de lectura.

               return 0;
}
```

CÓDIGO FUENTE SIN COMENTAR

SHELL SCRIPT MINI-SHELL

```
Comprobar_directorio()
{
      if (test -d cfg)
            then  echo "Existe el directorio cfg."
            else
                  echo "Faltan parámetros de configuración."
                  echo "No existe el directorio cfg..."
                  exit
      fi
}

Comprobar_ficheros()
{
      if ((test -s cfg/fusuarios)&&(test -s cfg/fordenes))
            then  echo "Ficheros correctos."
            else  echo "Faltan parámetros de configuración..."
                  echo "No existen los ficheros de configuración..."
                  exit
      fi
}

Peticion_login()
{
echo " "
encontrado=0
cont=3
while (test $encontrado -eq 0)
do
      echo -n "Introduzca el nombre de usuario: "
      read login
      if (`grep -q "^$login" cfg/fusuarios`)
            then  echo -n "Introduzca la contraseña: "
                  read password
                  if (`grep -q "^$login:$password" cfg/fusuarios`)
                        then  encontrado=1
                        else  echo "La contraseña no es correcta."
                              let cont=$cont-1
                              echo "Intentos restantes: "$cont
                  fi

            else  echo "El usuario $login no existe"
                  let cont=$cont-1
                  echo "Intentos restantes: "$cont
      fi

      echo " "

      if (test $cont -le 0)               # si cont<=0 fin
            then  echo "Nº de intentos superado. Fin del programa"
                  encontrado=1
                  exit
      fi
done
}
```

```
Iniciar()
{
      if (test -z $1)
            then  echo "No ha introducido ningún nombre de usuario"
            else  if (`grep -q "^$1" cfg/fusuarios`)
                        then  echo -n "Introduzca la contraseña: "
                              read contra
                              if (`grep -q "^$1:$contra" cfg/fusuarios`)
                                    then  login=$1
                                    else  echo "Contraseña incorrecta"
                              fi

                        else  echo "El usuario $1 no existe"
                  fi
      fi
}

Arbol_directorio()
{
      if (test -d $1)
            then  cd $1
                  if (test -z $1)
                        then  echo "- DIRECTORIO: $ruta/$i VACÍO"

                        else  for i in `ls`
                              do
                                          if (test -d $i)
                                          then  ruta=`pwd`
                                                echo "- DIRECTORIO: $ruta/$i"
                                                Arbol_directorio $i

                                          else  ruta=`pwd`
                                                echo "- FICHERO: $ruta/$i"
                                    fi
                              done
                  fi
                  cd ..
                  echo " "

            else  echo "ERROR. El directorio no existe."
      fi
}

Cambio()
{
      if (test -z $1)
            then  echo "Error, no ha escrito ningún nombre de usuario."

            else  if (`grep -q "^$1" cfg/fusuarios`)
                        then  echo -n "Introduzca la contraseña a cambiar: "
                              read clave
                              if (`grep -q "^$1:$clave" cfg/fusuarios`)
                                    then  echo -n "Introduzca la nueva: "
                                          read nuevo
                                          echo -n "Vuelva a teclear la nueva: "
                                          read renuevo
                                          if (test $nuevo = $renuevo)
                                                then  echo "Cambiando clave...."

                              grep -v "^$1:$clave" cfg/fusuarios > temp_dat
```

```
                              cp temp_dat cfg/fusuarios
                              rm temp_dat
                              echo "$1:$nuevo" >> cfg/fusuarios
                                                 else  echo "Error al reescribir."
                                    fi

                                    else  echo "Contraseña incorrecta."
                              fi

                        else  echo "El usuario $1 no existe."
                  fi
      fi
}

Ejecutar()
{
      gcc -o temporizador temporiza.c
      xterm -e ./temporizador $var1 $var2 $var3 $var4 $var5 $var6 &
}

Peticion_orden()
{
      echo -n "$login@minishell$"
      read orden var1 var2 var3 var4 var5 var6
      fecha=`date +'%d/%m/%Y-%H.%M'`

      if (test -s fhistorial)
            then  if(test `grep -c "." fhistorial` -lt 50)
                        then  echo "$orden" >> fhistorial
                        else  tail -n 9 fhistorial >> temp_dat
                              cp temp_dat fhistorial
                              rm temp_dat
                              echo "$orden" >> fhistorial
                  fi

            else  echo "$orden" > fhistorial
      fi


      if(`grep -q "^$orden:" cfg/fordenes`)
            then  grep -v "^$orden:" cfg/fordenes > temp_dat
                  cp temp_dat cfg/fordenes
                  rm temp_dat
                  echo "$orden:$fecha:" >> cfg/fordenes

                  case $orden in

                        "arbol_directorio")
                              directorio=`pwd`
                              echo "LISTADO: "
                              echo " "
                              Arbol_directorio $var1
                              cd $directorio
                        ;;

                        "cambia_clave")
                              Cambio $var1
                        ;;

                        "historico_ordenes")
                              if (test -z $var1)
                                    then var1=10
                              fi
```

```
                              tail -n $var1 fhistorial
                        ;;

                        "ejecutar_temporizador")
                              Ejecutar $var1 $var2 $var3 $var4 $var5 $var6
                        ;;

                        "iniciar")
                              Iniciar $var1
                        ;;

                        "salir")
                              exit
                        ;;

                  esac

            else  echo "ERROR. Orden no encontrada."
      fi

      echo " "

      Peticion_orden
}


Comprobar_directorio
Comprobar_ficheros
Peticion_login
Peticion_orden
```

TEMPORIZA.C

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>
#include <signal.h>
#include <time.h>
#include <sys/time.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>
#include <dirent.h>

int interv=0;
int condicion=0;
int tarea=0;

char caso1[75]="cut -f1,4 -d\":\" /etc/group > fgrupos; cat fgrupos; rm
fgrupos;\0";
char caso2a[10]="locate\0";
char caso3a[14]="if (test -s \0";
char caso3b[3]="/";
char caso3c[90]="); then echo \"El fichero si existe.\"; else echo \"El fichero
no existe.\"; fi\0";
char caso4a[10]="sort \0";
char caso4b[25]=" > version_ordenada/\0";
char caso4c[30]="; echo \"Creando fichero\";\0";
char caso5a[14]="usuarios=\"\0";
char caso5b[120]="\"; for j in $usuarios; do if (grep -q \"^$j\" /etc/passwd);
then echo \"$j existe\"; else echo \"$j no existe\"; fi; done\0";

char comando[250]="";
char direcc[30]="\0";

char espacio[3]=" \0";
char delimitador[3]="/\0";

int correcto=1;

FILE *fichero;
char *ptr,*nombrefiche;

struct sigaction accion;
struct itimerval valortemporizador;

static void orden()
{
      system(comando);
      condicion--;
}

void inicializa_senial()
{
      accion.sa_handler=orden;
      accion.sa_flags=SA_RESTART;
      sigemptyset(&accion.sa_mask);
      sigaction(SIGPROF, &accion,NULL);
}
```

```
void inicializa_intervalo(int intervalo)
{
      valortemporizador.it_interval.tv_sec=intervalo;
      valortemporizador.it_interval.tv_usec=0;
      valortemporizador.it_value=valortemporizador.it_interval;
      setitimer(ITIMER_PROF, &valortemporizador, NULL);
}

int main(int argc, char *argv[])
{
      int i,j;
        if (argc>=4)
      {
            interv=atoi(argv[1]);
            condicion=atoi(argv[2]);
            tarea=atoi(argv[3]);

            switch (tarea)
                  {
                        case 1:
                              if (argc==4)      strcat(comando,caso1);
                              else
                                    printf("Error, esta operación no necesita
                                    parámetros\n");
                              break;

                        case 2:
                              if (argc==5)
                                {
                                    strcat(comando,caso2a);
                                    strcat(comando,espacio);
                                    strcat(comando,argv[4]);
                                }

                              else  printf("Error,nº de parámetros incorrecto");
                              break;

                        case 3:
                              if (argc==6)
                                {
                                    strcat(comando,caso3a);
                                    strcat(comando,argv[5]);
                                    strcat(comando,caso3b);
                                    strcat(comando,argv[4]);
                                    strcat(comando,caso3c);
                                }

                              else  printf("Error,nº parámetros incorrecto\n");
                              break;

                        case 4:
                              if ((argc>4)&&(argc<8))
                                  {
                                    for(i=4; i<argc; i++)
                                          {
                                                fichero=fopen(argv[i],"rb");
                                                if (!fichero){
                                                printf("Error, al abrir.\n");
                                                correcto=0;
```

```
                                        break;
                                        }
                                        fclose(fichero);
                                    }

                                if (correcto==1)
                                        {
                                           for (j=4;j<argc;j++)
                                              {
                                              strcpy(direcc,argv[j]);
                                              ptr=strtok(direcc,delimitador);
                                    while ((ptr=strtok(NULL,delimitador)) != NULL)
                                              nombrefiche=ptr;
                                              strcat(comando,caso4a);
                                              strcat(comando,argv[j]);
                                              strcat(comando,caso4b);
                                              strcat(comando,nombrefiche);
                                              strcat(comando,caso4c);
                                              }
                                        }
                                    }

                                else  printf("Error, parámetros incorrectos.\n");
                                break;


                          case 5:
                                if ((argc>4)&&(argc<8))
                                    {
                                      strcat(comando,caso5a);
                                      strcat(comando,argv[4]);
                                      if (argc>=6){
                                                  strcat(comando,espacio);
                                                  strcat(comando,argv[5]);
                                                  }
                                      if (argc==7){
                                                  strcat(comando,espacio);
                                                  strcat(comando,argv[6]);
                                                  }
                                      strcat(comando,caso5b);
                                     }

                                else  printf("Error, nº de nombres incorrecto\n");

                                break;

                          default:
                                write(2,"ERROR, LA ORDEN NO EXISTE\n");
                                break;
                     }

               inicializa_senial();
               inicializa_intervalo(interv);

               while(condicion>0);
          }

          else printf("ERROR, faltan parámetros para ejecutar el temporizador.\n");
          getchar();

          return 0;
}
```

PROGRAMA EJECUTABLE

Para la ejecución de la práctica debemos ejecutar un terminal de comandos de Linux y otorgar permisos de ejecución al fichero Shell-script denominado mini-shell, para ello nos valdremos del comando *chmod*. A continuación bastará con ejecutar en el terminal *./mini-shell*

MANUAL DE USUARIO

Una vez iniciado el mini-shell, este nos pedirá el usuario y la contraseña, para lo cual disponemos de 3 intentos, en caso de que se utilizasen los 3 intentos y no se introdujesen un usuario/contraseña válidos, se finalizaría.

Si introducimos correctamente los datos de identificación nos aparecerá el prompt de nuestro mini-shell, formado por el login del usuario que ha iniciado sesión seguido del carácter @ y a continuación la cadena minishell$, y esperará a que el usuario le envíe órdenes a ejecutar. Las órdenes poseen los siguientes formatos:

arbol_directorio *nombre_directorio*

cambia_clave *usuario*

- Petición de contraseña anterior: *claveanterior*
- Nueva contraseña: *clavenueva*
- Repetir la nueva contraseña: *clavenueva*

historico_ordenes *numero_de_ordenes*

ejecutar-temporizador *intervalo ocurrencias accion param1 param2 ...*

- Intervalo: el número de segundos que transcurrirá entre cada ocurrencia de ejecución del temporizador.
- Ocurrencias: Nº de veces que se va a ejecutar el temporizador antes de pasar a la ejecución de otro.
- Acción: Tarea que va a realizar, valores numéricos del 1 al 5 que pertenecen a:
    1. Mostrar los grupos de usuarios existentes en el sistema, debidamente formateados y los usuarios que pertenecen a dicho grupo.
    2. Localizar un fichero pasado como parámetro y mostrar las ubicaciones donde se ha encontrado.
    3. Mostrar si un fichero pasado como primer parámetro existe en un directorio pasado como segundo parámetro.
    4. Ordenación de todos y cada uno de los ficheros enviados como parámetros, hasta un máximo de 3, almacenando la versión ordenada en un directorio denominado “version_ordenada”.
    5. Comprobar si en el fichero de usuarios del sistema (/etc/passwd), existen los usuarios que se pasen como parámetros (como máximo 3).
- Parámetro: será necesario para aquellas acciones que necesiten adicionalmente un parámetro adicional