Comandos de GNU/Linux y programas C

Jordi Calopa Bosch

Revisión 2023.07

Indice

1. Introducción

- Todos los ejemplos que se muestran en este documento han sido testeados por el autor, mayoritariamente en un entorno GNU/Linux (Debian), por lo que no es posible garantizar que funcionen correctamente en otro sistema. Se entregan sin absolutamente ninguna garantía.

- Esta revisión del documento sustituye a cualquier otra emitida con anterioridad.

2. Comandos de Linux

2.1 Sintaxis de los comandos

Comando1 [opciones] argumento_1 argumento_2 ... argumento_N

2.2 Redirecciones

> Es un operador de redirección que dirige la salida estándar hacia el fichero indicado a continuación

Ejemplo: *ls >file*

>> Redirige la salida estándar hacia otro fichero (append) sin sobrescribir

Ejemplo: *date >>archivo*
cat file1 file2 >file3
cat file2 >>file1

< Redirige la entrada estándar desde un determinado fichero

Ejemplo: *mail jordi <carta*

2.3 Pipelines

Para enviar sin pipelines a *jordi* una lista de nuestros ficheros:

ls >FicheroTemporal
mail jordi < FicheroTemporal
rm FicheroTemporal

Con pipelines ... *ls | mail jordi*

Con el operador de tubería | se pueden concatenar tantos comandos como se desee.

2.4 Procesos en Background

Para ejecutar un programa en backcground es preciso el carácter & al final del comando:

Ejemplo:

program <datos.d >resultados.r &

Si además necesitamos mantener la ejecución del programa incluso después de salir de nuestra sesión, deberemos utilizar el comando nohup.

Ejemplo:

nohup program

Las salidas del programa se redirigen a un fichero llamado *nohup.out*.

2.5 Comandos

ac

Muestra información sobre el tiempo de conexión de los usuarios

Ejemplo 1

ac -d

```
Dec  2   total       2.30
Dec  3   total       5.46
Today    total       1.62
```

Observaciones

Es posible que previamente sea preciso instalar: apt install acct

adduser

Permite añadir usuarios al sistema y agregar usuarios a grupos existentes

Ejemplo 1

Añade el usuario jordi al grupo vboxusers

adduser jordi vboxusers

apt

Utilidad para el manejo de paquetes (Advanced Packaging Tool) en Debian

Ejemplo 1

Instalar BD mariadb en Debian

apt install mariadb-server

Ejemplo 2

Instalar conector python para mysql

apt-get install python3-mysql.connector

Ejemplo 3

Instalar virtualbox

apt install virtualbox

Ejemplo 4

Instalar KVM

apt-get install qemu-kvm libvirt-bin bridge-utils virt-manager

Ejemplo 5

Instalar MySQL en Ubuntu

apt-get install mysql-server

Ejemplo 6

Instalar MySQL en Kali

sudo apt update
sudo apt install mysql-community-server
sudo systemctl enable --now mysql

Ejemplo 7

Instalar conector MySQL - C/C++

apt-get install libmysqlcppconn-dev

Ejemplo 8 – Instalar VirtualBox en Debian 11

Requisitos
apt install build-essential linux-headers-amd64
apt install gcc
apt install gnupg2
apt install linux-headers-$(uname -r) dkms

Descargar el paquete Debian
virtualbox-7.0_7.0.4-154605~Debian~bullseye_amd64.deb

Instalar VirtualBox
export PATH=$PATH:/usr/local/sbin:/usr/sbin:/sbin ‘ Si es necesario
dpkg -i virtualbox-7.0_7.0.4-154605~Debian~bullseye_amd64.deb

arch

Visualiza el nombre de la máquina; equivalente al comando “*uname -m*”.

Ejemplo 1

arch

awk

Permite procesar ficheros y la manipulación del contenido de los mismos.

Ejemplo 1

Eliminar la primera columna del archivo aaa.txt y guardar la salida en un fichero denominado bbb.txt.

awk '{$1 = "";print>"bbb.txt" }' aaa.txt

bash

Establece el shell Bash por defecto, durante la sesión activa.

Ejemplo

bash

cal

Muestra el calendario

Ejemplo 1

Mostrar mes en curso

cal

Ejemplo 2

Mostrar el calendario del año 2019

cal 2019

Ejemplo 3

Mostrar el mes de julio del año 2019

cal 07 2019

Ejemplo 4

Mostrar el calendario del año 2019 en formato Juliano

cal -j 2019

cat

Lee y muestra la información de uno a varios ficheros

Ejemplo 1

Mostrar el contenido de *mifichero*

cat mifichero

Ejemplo 2

Mostrar el contenido de dos ficheros

cat mifich1 mifich2

Ejemplo 3

Concatenar dos ficheros sobre uno de salida

cat fich1 fich2 > fich3

Ejemplo 4

Cargar un fichero en una variable

var1=$(cat temp.txt)

chmod

Cambia los permisos de acceso a un fichero

Sintáxis

chmod xyz

x = Propietario
y = Grupo
z = Resto

0 = --- = sin acceso
1 = --x = ejecución
2 = -w- = escritura
3 = -wx = escritura y ejecución
4 = r-- = lectura
5 = r-x = lectura y ejecución
6 = rw- = lectura y escritura
7 = rwx = lectura, escritura y ejecución

Ejemplo 1

Asignar al fichero denominado "*mifichero*" permisos para que cualquier usuario pueda utilizarlo y modificarlo.

chmod 777 mifichero

Ejemplo 2

Asignar al fichero denominado "*mifichero*" permisos para que cualquier usuario pueda ejecutarlo y verlo, pero no pueda modificarlo.

chmod 755 mifichero

chown

Cambia el propietario de directorios y/o archivos

Ejemplo 1

Cambiar el propietario del directorio *Documentos*, indicándole que el nuevo propietario es *jordi*. Además con la opción -R le indicamos que actúe de forma recursiva sobre los subdirectorios y ficheros que dependen de él.

chown -R jordi Documentos

clear

Limpia la pantalla del terminal

Ejemplo 1

clear

convert

Este comando permite convertir entre formatos de imagen, así como cambiar el tamaño de una imagen, difuminar, recortar, dibujar y voltear, entre otras opciones.

Ejemplo 1

Convertir imagen de formato JPG a formato PPM

convert imagen.JPG imagen.PPM

cp

Copia archivos y directorios

Ejemplo 1

Copiar el contenido del directorios /home/jordi/JCB/* sobre otro de destino denominado /media/jordi/HDD1/JCB, que se encuentra en el disco HDD1.

cp -r -v /home/jordi/JCB/* /media/jordi/HDD1/JCB

date

Visualiza o modifica la fecha y la hora del sistema.

Ejemplo 1

visualizar fecha y hora

date

dd

Utilidad para la copia y conversión de datos.

Ejemplo 1

Generar una imagen ISO a partir de un CD

dd if=/dev/cdrom of=/home/jordi/Escritorio/cd.iso

Ejemplo 2

Convierte el contenido de un archivo a mayúsculas

dd if=aaa.txt of=bbb.txt conv=ucase

Ejemplo 3

Copiar imagen ISO a unidad USB

dd if=ubuntu-20.04.2.0-desktop-amd64.iso of=/dev/sde bs=512k

df

Muestra información de los sistemas de archivos montados.

Ejemplo 1

Mostrar la ocupación en KB.

df -k

Ejemplo 2

Mostrar la ocupación en MB.

df -m

du

Visualiza el tamaño de los directorios, a partir del directorio actual.

Ejemplo 1

Mostrar tamaño de subdirecorios del directorio actual

du

Ejemplo 2

Mostrar la ocupación del directorio Dir1

du -sh Dir1

echo

Visualiza una línea de texto

Opciones

-n	Visualiza el texto sin el Carriage Return final
-e	Activa el interprete de secuencia de escape
-E	Desactiva el interprete de secuencias de escape

Si la opción -e está activa:

\\	Backslash
\a	Alerta (BEL)
\b	Backspace
\e	Escape
\f	Form feed
\n	Nueva Línea
\r	Carriage Return (CR)
\t	Tabulación Horizontal
\v	Tabulación Vertical
\0NNN	Byte con el valor octal NNN
\xHH	Byte con el valor hexadecimal HH

Ejemplo 1

```
cadena="abcdefghijklmnopqrstuvwxyz"
echo "Subcadena desde posición 3 ....." ${cadena:3}
echo "Subcadena desde posición 3 con longitud 7 .." ${cadena:3:7}
echo "Replace primera aparición ....." ${cadena/jkl/XXX}
echo "Replace todas las apariciones ....." ${cadena//rst/XXX}
echo "Replace si coincide al pricipio ....." ${cadena/#abc/XXX}
echo "Replace si coincide al final ....." ${cadena/%xyz/XXX}
echo "Eliminar subcadena si coincide desde inicio." ${cadena#abc}
echo "Eliminar subcadena si coincide desde final.." ${cadena%xyz}
echo "Longitud de la cadena ....." ${#cadena}
```

exit

Causa la terminación normal de un proceso. Dentro de un proceso batch (script) finaliza el propio proceso. Interactivamente desde el terminal, finaliza la sesión online.

Ejemplo 1

exit

export

Añadir variables de entorno.

Ejemplo 1

export PATH=$PATH:/usr/local/sbin:/usr/sbin:/sbin

find

Busca un fichero o conjunto de ficheros dentro del sistema de directorios.

Ejemplo 1

Busca fichero file1

find -name file1

Ejemplo 2

Busca file1 a partir del directorio raiz

find / -name file1

Ejemplo 3

Muestra el sistema de directorios

find / -type d

Ejemplo 4

Muestra el contenido del directorio /etc

find /etc

for

Bucle para ejecución condicional de sentencias

Sintáxis

For *condición*
do
sentencias
done

Ejemplo 1

Listar los números del 0 al 15

```
for j in {0..15}
  do
    echo $j
  done
```

Ejemplo 2

Listar los nombres de todos los elementos del directorio en curso

```
for filename in *.*
  do
    echo  $filename
  done
```

Ejemplo 3

Compila los programas .c que encuentra en el directorio.

```
prefijo="gcc -o /usr/local/bin/"
for filename in *.c
  do
    objeto=${filename%.c}
    comando="$prefijo$objeto $filename"
    echo $comando
    $comando
  done
```

ftp

Transferencia de ficheros entre distintas máquinas (local y remota)

Ejemplo 1: Desde Mvs a Unix

```
ftp                                    (estando en UNIX de HP)
>open maquina
>user usuario password
>bin
>get 'AAAA.BBBB.CCCC'  /home/XXX/xxxxxxxx
>bye
```

Ejemplo 2: Desde Unix a DOS

```
ftp                                    (estando en el DOS del PC)
>open maquina
>user usuario password
>bin
>get /home/XXX/nombrefichero
>bye
```

Ejemplo 3: Desde DOS a Unix

```
ftp                                    (estando en el DOS del PC)
>open maquina
>user usuario password
>bin
>put nombrefichero  /home/XXX/nombrefichero
>bye
```

Ejemplo 4: Proceso batch para cargar datos sobre un servidor FTP

```
@echo off
echo user nombre_servidor_FTP> Mputs.dat
echo contraseña>> Mputs.dat
echo bin>> Mputs.dat
echo cd html>> Mputs.dat
echo mput *.C>> Mputs.dat
echo mput *.CSS>> Mputs.dat
echo mput *.GIF>> Mputs.dat
echo mput *.HTM>> Mputs.dat
echo mput *.JPG>> Mputs.dat
echo mput *.PDF>> Mputs.dat
echo mput *.RAR>> Mputs.dat
echo mput *.TXT>> Mputs.dat
echo quit>> Mputs.dat
rem
ftp -n -i -s:Mputs.dat nombre_servidor_FTP
```

grep

Busca dentro de un fichero y visualiza las palabras que se corresponde con un determinado patrón.

Ejemplo 1

Busca word1 dentro del fichero Backup.log

grep word1 Backup.log

help

Muestra todos los comando que el intérprete tiene definidos internamente.

Ejemplo 1

Muestra la lista de comandos que tiene definidos

help

Ejemplo 2

Muestra la ayuda del comando echo

help echo

history

Muestra el histórico de Bash; los comandos entrados por consola.

Ejemplo 1

Guardar el histórico en un fichero

history > historico.txt

hostname

Visualiza el nombre del host del sistema.

Ejemplo 1

hostname

ifconfig

Configurar los interfaces de red. Sin parámetros muestra el estado de todas las interfaces de red que están activos.

Ejemplo 1

Activar interfazz eth0

ifconfig eth0 up

Ejemplo 2

Desactivar la interfaz eth0

ifconfig eth0 down

ip

Comando para configurar los interfaces de red y enrutamientos. Sin parámetros muestra la sintaxis del comando.

Ejemplo 1

Mostrar detalles de las interfaces

ip addr show

Ejemplo 2

Activar la interfaz wlp3s0

ip link set wlp3s0 up

kill

Envía una señal a un proceso. Si se omite el número de señal se envía la 15 (SIGTERM) por defecto.

Ejemplo 1

Cancelar proceso identificado como 1234

kill 1234

ls

Visualiza información de los archivos, por defecto del directorio actual.

Ejemplo 1

Muestra archivos y directorios incluyendo los ocultos y ordenados por fecha de la última modificación.

ls -at

Ejemplo 2

Muestra todos los archivos y directorios cuyo nombre empieza por D.

ls D*

lspci

Visualiza los dispositivos PCI

Ejemplo 1

Mostrar dispositivos PCI en formato de arbol

lspci -t

lsusb

Visualiza los dispositivos USB.

Ejemplo 1

Muestra dispositivos USB (la opció -t indica con formato árbol)

lsusb -t

man

Visualiza los manuales de referencia del sistema.

Ejemplo 1

Muestra el manual del comando ls

man ls

Ejemplo 2

Muestra la descripcióin breve del comando ls

man -f ls

Ejemplo 3

Lista todos los manuales que contenga la palabra word1

man -k word1

passwd

El comando passwd cambia las contraseñas de las cuentas de usuario. Un usuario normal puede cambiar la contraseña de su propia cuenta únicamente; El superusuario puede cambiar la contraseña de cualquier cuenta.

Ejemplo 1

Cambiar contraseña de superuser

sudo passwd root

Ejemplo 2

Visualiza información de estados de las cuentas

passwd -a -S

printenv

Visualiza una o varias variables de entorno. Sin argumentos las visualiza todas.

Ejemplo 1

Ver el contenido de las variables de entorno

printenv

Ejemplo 2

Ver el contenido de la variable SHELL

printenv SHELL

ps

Muestra información de los procesos en curso.

Sintaxis

ps [OPTIONS]

Opciones

-e Visualiza todos los procesos del sistema
-x Visualiza los procesos independientes de un terminal

pwd

Visualiza el nombre completo (pathname) del directorio actual

Ejemplo 1

pwd

recode

Convierte ficheros entre distintos conjuntos de caracteres

Ejemplo 1

Muestra los distintos CharacterSet permitidos

recode -l

rename

Cambia el nombre de un conjunto de ficheros

Ejemplo1

Cambia, para todos los archivos JPG encontrados, el prefijo “IMG” por el de ”misfotos_2020”.

rename 's/IMG/misfotos_2020/g' *.JPG

Ejemplo2

Cambia los nombres de todos los elementos a minúsculas.

***rename** '**y/A-Z/a-z/**' **

Ejemplo3

Cambia los nombres de todos los elementos a mayúsculas.

***rename** '**y/a-z/A-Z/**' **

Ejemplo4

Con la opción -n, solamente muestra los cambios a realizar pero no los efectúa.

***rename -n** '**y/a-z/A-Z/**' **

Observaciones

Es posible que previamente sea preciso instalar: *apt install rename*

resize

Cambia el tamaño (de la ventana) del terminal

Ejemplo 1

Cambia el tamaño de la pantalla del terminal a 30 filas y 100 columnas.

resize -s 30 100

Observaciones

Es posible que previamente sea preciso instalar: *apt-get install xterm*

rm

Borra ficheros

Ejemplo 1

Borra fichero myfile.

rm myfile

rmdir

Borra Directorios

Ejemplo 1

Borra "*midirectorio*" y todo su contenido.

rm -R midirectorio

sed

Editor de secuencias para editar y convertir textos

Ejemplo 1

Copiar el FicheroA sobre FicheroB añadiéndole el prefijo XXXXX

sed -e 's/^/XXXXX/' FicheroA > FicheroB

Ejemplo 2

Copiar el FicheroA sobre FicheroB añadiéndole el sufijo YYYYY

sed -e 's/$/YYYYY/' FicheroA > FicheroB

seq

Generar una secuencia numérica

Ejemplo 1

De 1 a 100 incrementando de tres en tres

seq 1 3 100

sleep

Provocar un tiempo de retardo

Ejemplo 1

Retardo de tres segundos

sleep 3

source

Permite lanzar un proceso, desde otro script, sin modificar las variable de entorno.

Ejemplo 1

source NombreScript.sh

su

Permite cambiar de usuario o identificarse como superusuario.

Sintaxis

su [OPTIONS] [USERNAME]

top

Visualiza el estado del sistema y los procesos activos. La información en pantalla se refresca automáticamente cada 3 segundos, por defecto.

Ejemplo 1

Muestra información y refresca cada 5 segundos

top -d5

uname

Muestra información del sistema

Ejemplo 1

Informa del sistema operativo

uname -o

Observaciones

-a Todo
-s Nombre del Kerne
-n Nombre de host del nodo de red
-r Release de la Versión del kernel
-v Versión del kernel
-m Nombre del hardware de la máquina
-p Procesador
-i Plataforma de hardware
-o Sistema Operativo

wget

Descargador de contenidos de red (no interactivo)

Ejemplo 1

Descarga el contenido de "*NombreWeb*".

wget -r -p NombreWeb

Ejemplo 2

Descarga los ficheros pdf de "*NombreWeb*".

wget -r -p -A "*.pdf" NombreWeb

who

Ejemplo 1

who

Observaciones

-a todos

-b	hora del último arranque del sistema
-d	procesos muertos
-H	línea de encabezados de columna
--ips	Ver ips en lugar de nombres de host
-l	procesos de inicio de sesión del sistema
-p	procesos activos generados por init
-q	todos los nombres de usuario y la cantidad de usuarios conectados
-r	nivel de ejecución actual
-s	solo nombre, línea y hora (predeterminado)
-t	el último cambio de reloj del sistema
-T, -w	agregar el estado del mensaje del usuario como +, - o?
-u	lista de usuarios conectados

whoami

Visualiza el usuario activo

Ejemplo 1

whoami

Observaciones

--version Visualiza información de la versión del comando.

zypper

Utilidad para el manejo de paquetes para OpenSuse

Ejemplo 1

Instalar MySql en OpenSuse

zypper install mysql
rcmysql start

2.6 Resumen de comandos

ac	Informa sobre el tiempo de conexión de los usuarios
adduser	Permite añadir usuarios al sistema y agregar usuarios a grupos existentes
apt update	Actualiza la lista de paquetes
apt upgrade	Instala últimas versiones de paquetes
arch	Visualiza el nombre de la máquina; equivalente al comando “uname -m”.
awk	Permite procesar ficheros y la manipulación del contenido de los mismos.
bash	Establece el shell Bash por defecto, durante la sesión activa.
cal	Muestra el calendario
cat	Lee/muestra la información de uno o varios ficheros
chmod	Cambia los permisos de acceso a un fichero
chown	Cambia el propietario de directorios y/o archivos
clear	Limpia la pantalla del terminal
convert	Permite convertir entre formatos de imagen, así como cambiar el tamaño de una imagen, difuminar, recortar, dibujar y voltear, entre otras opciones.
cp	Copia archivos y directorios
date	Visualiza o modifica la fecha y la hora del sistema.
dd	Utilidad para la copia y conversión de datos.
df	Muestra información de los sistemas de archivos montados.
du	Visualiza el tamaño de los directorios, a partir del directorio actual.
dpkg -i *.*deb	Instala el paquete *nombre.deb*
dpkg -l	Lista de paquetes instalados
echo	Visualiza lineas de texto
exit	Causa la terminación normal de un proceso.
export	Añadir variables de entorno.
find	Busca un fichero o conjunto de ficheros dentro del sistema de directorios.
for	Bucle para ejecución condicional de sentencias
ftp T	Transferencia de ficheros entre distintas máquinas (local y remota)
grep	Busca dentro de un fichero y visualiza las palabras que se corresponde con un determinado patrón.
help	Muestra todos los comando que el intérprete tiene definidos internamente.
history	Muestra el histórico de Bash; los comandos entrados por consola.
hostname	Visualiza el nombre del host del sistema.
ip	Configurar los interfaces de red y enrutamientos. Sin parámetros muestra la sintaxis del comando. En versiones anteriores se denominaba ifconfig.
kill	Envía una señal a un proceso. Si se omite número de señal se envía la 15 por defecto.
last	Muestra los arranques del sistema y los usuarios que han accedido al mismo.
ls	Visualiza información de los archivos, por defecto del directorio actual.
lspci	Visualiza los dispositivos PCI
lsusb	Visualiza los dispositivos USB.
man	Visualiza los manuales de referencia del sistema.
passwd	Cambia las contraseñas de las cuentas de usuario.
printenv	Visualiza una o varias variables de entorno. Sin argumentos las visualiza todas.
ps	Muestra información de los procesos en curso.
pwd	Visualiza el pathname del directorio actual
recode	Convierte ficheros entre distintos conjuntos de caracteres
rename	Cambia el nombre de un conjunto de ficheros
resize	Cambia el tamaño de la ventana del terminal
rm	Borra ficheros
rmdir	Borra Directorios
sed	Editor de secuencias para editar y convertir textos
seq	Generar secuencias numéricas
sleep	Provocar un tiempo de espera
source	Permite lanzar un proceso, desde otro script, sin modificar las variable de entorno.
su	Permite cambiar de usuario o identificarse como superusuario.
systemctl	Administración del sistema y servicios
top	Visualiza estado del sistema y procesos activos.
uname	Muestra información del sistema
wget	Descargador de contenidos de red (no interactivo)
who	Ver usuarios conectados
whoami	Visualiza el usuario activo

3. Scripts de ejemplo

amalgama.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: amalgama.sh                                              *
#* Sistema: GNU/Linux                                                *
#* Autor:   JCB                                                      *
#*                                                                   *
#* Sinópsis: Proceso que copia el contenido de los scripts en un  *
#* fichero de texto, intercalando varias líneas de separación.    *
#*                                                                   *
#*********************************************************************
clear
#
fileOUT="/usr/local/bin/scripts.txt"
fileOUT="./scripts.txt"
let "contador = 0"
sep=" "
lin="-------------------------------------------"
echo $sep > $fileOUT
#
for fileINP in *.sh
   do
     let "contador = contador + 1"
     echo " $contador. $fileINP " >> $fileOUT
     echo $lin >> $fileOUT
     echo $sep >> $fileOUT
     cat $fileINP >> $fileOUT
     echo $sep >> $fileOUT
     echo $sep >> $fileOUT
     echo $sep >> $fileOUT
     echo $sep >> $fileOUT
     echo $sep >> $fileOUT
     echo $fileINP
   done
#
exit
```

backupgm.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: backupgm.sh                                              *
```

```
#* Sistema: GNU/Linux                                                  *
#* Autor:   JCB                                                        *
#*                                                                     *
#* Sinópsis: Copia de respaldo de programas                            *
#*                                                                     *
#*********************************************************************
clear
#
# Copia de programas
#
if test -d /home/jordi/Documentos/PgmsC
then
  echo -e "\e[92mEl directorio de programas ya existe \e[m"
else
  echo -e "\e[92mGenerando directorio ./PgmsC \e[m"
  mkdir /home/jordi/Documentos/PgmsC
fi
#
cp -r -v ./*.c /home/jordi/Documentos/PgmsC
#
exit
```

backupsc.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: backupsc.sh                                               *
#* Sistema: GNU/Linux                                                 *
#* Autor:   JCB                                                       *
#*                                                                    *
#* Sinópsis: Proceso que hace una copia de todos y cada uno de        *
#*           los scripts bash del directorio actual, si no existe     *
#*           dicha copia.                                             *
#*                                                                    *
#*********************************************************************
clear
#
for filename in *.sh
do
   if [ -f ./files/$filename.txt ]
   then
      echo "$filename.txt ya existe..."
      continue
   fi
   cp $filename ./files/$filename.txt
done
#
exit
```

chmodall.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: chmodall.sh                                              *
#* Sistema: GNU/Linux                                                *
#* Autor:   JCB                                                      *
#*                                                                   *
#* Sinópsis: Cambia los permisos de acceso de todos los elementos *
#*           de un directorio                                        *
#*                                                                   *
#*********************************************************************
clear
#
for fichero in $(ls /home/jcb/Documentos/ScriptsBash)
do
  chmod 777 "$fichero"
done
exit
```

colortrm.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: colortrm.sh                                              *
#* Sistema: GNU/Linux                                                *
#* Autor:   JCB                                                      *
#*                                                                   *
#* Sinópsis: Visualiza los posibles colores del terminal          *
#*                                                                   *
#*********************************************************************
clear
#
for((contador=1; contador<=107; contador++))
   do
     if [ ${contador} -gt 29 ] && [ ${contador} -lt 38 ]; then
      echo -e "\e[${contador}mColor ${contador}\e[0m"
     fi
     if [ ${contador} -gt 39 ] && [ ${contador} -lt 48 ]; then
      echo -e "\e[${contador}mColor ${contador}\e[0m"
     fi
     if [ ${contador} -gt 89 ] && [ ${contador} -lt 98 ]; then
      echo -e "\e[${contador}mColor ${contador}\e[0m"
     fi
     if [ ${contador} -gt 99 ] && [ ${contador} -lt 108 ]; then
      echo -e "\e[${contador}mColor ${contador}\e[0m"
     fi
   done
exit
```

compilar.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: compilar.sh                                              *
#* Sistema: GNU/Linux                                                *
#* Autor:   JCB                                                      *
#*                                                                   *
#* Sinópsis: Compila todos los programas .c que encuentra en         *
#* el directorio y guarda los ejecutables en /usr/local/bin/         *
#*                                                                   *
#*********************************************************************
clear
#
echo $0
prefijo="gcc -o /usr/local/bin/"
for filename in *.c
   do
     objeto=${filename%.c}
     comando="$prefijo$objeto $filename"
     echo $comando
     $comando
   done
exit
```

copiaseg.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: copiaseg.sh                                              *
#* Sistema: GNU/Linux                                                *
#* Autor:   JCB                                                      *
#*                                                                   *
#* Sinópsis: Copia de seguridad sobre discos HDD2 y HDD3             *
#*                                                                   *
#*********************************************************************
clear
#
cp -r -v -f /home/jordi/Documentos/* /media/jordi/HDD2/JCB
cp -r -v -f /home/jordi/Documentos/* /media/jordi/HDD3/JCB
tar      -cpvzf      "/media/jordi/HDD2/Backup_`date      +%d%m%Y%H%M`.tgz"
/home/jordi/Documentos/*
#
exit
```

DBbackup.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: DBbackup.sh                                              *
#* Sistema: GNU/Linux                                                *
```

```
#* Autor:   JCB                                                        *
#*                                                                     *
#* Sinópsis: Volcado de la BD DMSTO sobre fichero plano                *
#*                                                                     *
#*********************************************************************
clear
#
f1="/usr/local/bin/DumpDMS_"
f2=$(date +%Y%m%d)
f3=".SQL"
file=$f1$f2$f3
echo "Dump de la BD sobre el fichero $file"
mysqldump -u root -pxxxxxx DMSTO > $file
if [ $? -eq 0 ]
then
   echo "Volcado OK."
else
   echo "Error durante el backup de la base de datos"
read dummy
fi
#
exit
```

DBrestor.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: DBrestor.sh                                              *
#* Sistema: GNU/Linux                                                *
#* Autor:   JCB                                                      *
#*                                                                   *
#* Sinópsis: Restore de la base de datos DMSTO                       *
#*                                                                   *
#*********************************************************************
clear
#
mysql -u root --password=xxxxxx DMSTO < /usr/local/bin/RESTORE.SQL
#
RC=$? ; writelog "$0 RC=$RC $USER $UID $HOSTNAME $(date +%d-%m-%Y)"
echo -e "\e[96m " ; read -p " *** End of Job $0 *** RC=$RC *** " dummy
```

desktop.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: desktop.sh                                               *
#* Sistema: GNU/Linux                                                *
#* Autor:   JCB                                                      *
#*                                                                   *
#* Sinópsis: Modificar configuración escritorio                      *
```

```
#*                                                                    *
#********************************************************************
clear
#
gsettings set org.gnome.dsktop.background primary-color '#0000FF'
gsettings set org.gnome.dsktop.background picture-options 'centered'
#
exit
```

FTPbajar.sh

```
#!/bin/bash
#********************************************************************
#* Proceso: FTPbajar.sh                                             *
#* Sistema: GNU/Linux                                               *
#* Autor:   JCB                                                     *
#*                                                                  *
#* Sinópsis: Download de ficheros del servidor FTP                  *
#*                                                                  *
#********************************************************************
clear
#
ftp -inv XXXXXXXX<<FINFTP
       user YYYYYYYY ZZZZZZZZ
       binary
       cd /html
       get FFFFFFFF.TXT
       bye
FINFTP
exit
```

FTPsubir.sh

```
#!/bin/bash
#********************************************************************
#* Proceso: FTPsubir.sh                                             *
#* Sistema: GNU/Linux                                               *
#* Autor:   JCB                                                     *
#*                                                                  *
#* Sinópsis: Subir ficheros al servidor FTP                         *
#*                                                                  *
#********************************************************************
clear
#
ftp -inv XXXXXXXX<<FINFTP
       user YYYYYYYY ZZZZZZZZ
       binary
       lcd /home/jcb/Documentos
       cd /html
       put index.html
```

```
       put dos.pdf
       put img.jpg
       bye
FINFTP
exit
```

history.sh

```
#   #!/bin/bash
#*********************************************************************
#* Proceso: history.sh                                               *
#* Sistema: GNU/Linux                                                *
#* Autor:   JCB                                                      *
#*                                                                   *
#* Sinópsis: Proceso que vuelca el histórico bash en un fichero      *
#* que después visualiza por consola.                                *
#*                                                                   *
#*********************************************************************
clear
#
# cat ~/.bash_histor  > /usr/local/bin/histbash.txt
history > /usr/local/bin/histbash.txt
cat /usr/local/bin/histbash.txt
#
exit
```

lanpings.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: lanpings.sh                                              *
#* Sistema: GNU/Linux                                                *
#* Autor:   JCB                                                      *
#*                                                                   *
#* Sinópsis: Búsqueda lenta de direcciones activas dentro de         *
#*           una red local (si no es posible utilizar nmap).         *
#*                                                                   *
#*********************************************************************
clear
#
for i in {0..255}
do
   h="192.168.1.$i"
   ping -c 1 -q "$h" &>/dev/null
   if [ $? -eq 0 ]
   then
      echo "$h OK"
   fi
done
exit
```

showarg.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: showarg.sh                                               *
#* Sistema: GNU/Linux                                                *
#* Autor:   JCB                                                      *
#*                                                                   *
#* Sinópsis: Muestra argumentos de entrada al script                 *
#*                                                                   *
#*********************************************************************
clear
#
for i in $@
do
    echo "argumento: $i"
done
#
exit
```

showdoc.sh

```
#!/bin/bash
#*********************************************************************
#* Proceso: showdoc.sh                                               *
#* Sistema: GNU/Linux                                                *
#* Autor:   JCB                                                      *
#*                                                                   *
#* Sinópsis: Visualizar documento de información PDF                 *
#*                                                                   *
#*********************************************************************
clear
#
evince ./docum/$1
#
exit
```

4. Referencia breve del lenguaje C

4.1 Declaración de constantes

```
Sintáxis: #define(nombre)(valor)

Ejemplo: #define NOMBRECONSTANTE 1000
```

4.2 Tipos definidos por el usuario

```
Sintáxis: typedef(tipo de dato)(nombre);

Ejemplo: typedef int KILOGRAMOS;
```

4.3 Declaración de variables

```
Sintáxis: (variable type)(nombre 1)(nombre 2),...;

Ejemplo: int cantidad, precio;
         char caracter1, caracter2;
         int matriz[2][5];
         char vector[10];

Sintáxis: (variable type)(nombre)=(valor);

Ejemplo: int mi_numero=5;

Asignación:

Sintáxis: (nombre)=(valor);

Ejemplo: mi_numero=5;
         Alpha='a';
```

4.4 Definición de uniones

Declaración

```
Sintáxis: union(etiqueta)
          {(type)(nom_elemento);
          (type)(nom_elemento);
          ...
          }(variable nombre);

Ejemplo: union demoetiquetanombre
         {int a;
         float b;
                                                    }nombre_var;
```

Asignación

```
Sintáxis: (etiqueta).(nom_elemento)=(valor);
          nombre_var.a=1;
          nombre_var.b=4.6;
```

4.5 Definición de estructuras

Declaración

```
Sintáxis: struct(etiqueta)
          {(type)(variable);
          (type)(variable);
          ...
          }(variable list);

Ejemplo: struct estudiante
         {int idnum;
         int grado;
         char mnemotecnico;
         } primero,segundo,tercero,cuarto;
```

Asignación

```
Sintáxis: (variable nombre).(member)=(valor);

Ejemplo: primero.idnum=123;
         segundo.grado=456;
```

4.6 Operadores

```
+,-,*,/   operadores aritméticos
++        incremento                ++ a;
--        decremento                -- a;
%         mod                       a = b % c;
>         mayor que                 if (a > b)
>=        mayor o igual que         if (a >= b)
<         menor que                 if (a < b)
<=        menor o igual que         if (a <= b)
==        exactamente igual a       if (a == b)
=         Asignación                a=25;
!=        no igual                  if (a != b)
!         no                        if (!a)
&&        AND                       if (a) && (b)
||        OR                        if (a) || (b)
&         bitwise and               a = b & c;
|         bitwise or                a = b - c;
>>        shiftar-derecha           a = b >> 2;
<<        shiftar-izquierda         a = b << 2;
~         complemento a 1           a = ~b
```

4.7 Tratamiento de strings (#include <string.h>)

```
strlen – devuelve la longitud de la cedena
```

```
strcpy – copia cadenas de caracteres
strcat – concatena cadenas
strcmp – compara cadenas de caracteres
sprintf – printf con salida a otro string
```

4.8 Tratamiento de caracteres (#include <ctype.h>)

```
isalnum -
isalpha -
isascii -
isblank -
iscntrl -
isdigit -
isgraph -
islower -
isprint -
ispunct -
isspace -
isupper -
isxdigit -
```

4.9 Entrada y Salida (#include <stdio.h>)

Entrada

```
Sintáxis: scanf("(formateadores)",(variables));

Ejemplo: scanf("%d %c %s",varuno,vardos,vartres);

Formateadores:

%d decimal integer
%o octal integer
%x hex integer
%h short integer
%c character
%s string
%r real valor
%e exponential notation
```

Salida

```
Sintáxis: printf("(formateadores)",(variables));

Ejemplo: printf("%d %c", numero1, numero2);

Formateadores:

%d decimal
%u unsigned decimal
%o octal
∧  cba ∧=
%h hex
%e exponential
%f float
```

```
%g shorter of %e or %f
%c char
%s string

Secuencias de escape:

\n salto de línea
\t tabulación
\r CR
\f FF
\b spacio atrás
```

4.10 Estructuras de control

FOR

```
for ((expr_inicio);(expr_fin);(expr_increento))
    {
    (sentencias);
    }
```

WHILE

```
while ((condición))
    {
    (sentencias);
    }
```

DO WHILE

```
do {
    (sentencias);
    }
while ((condición));
```

IF

```
if ((condición))
    {
    (sentencias);
    }
```

IF... ELSE

```
if ((condición))
    (sentencia 1);
else
    (sentencia 2);
```

SWITCH

```
switch ((expression))
    {case (valor 1):(sentencia 1);
     case (valor 2):(sentencia 2);
     ...
     default:(sentencia n);  }
```

5. Programas C de ejemplo

amalgama.c

```
/*****************************************************************************/
/*** Programa : amalgama.c                                                 ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Este programa genera un proceso Bash, que a su vez genera    ***/
/***          un fichero con el contenido de todos los programas C, del    ***/
/***          directorio en curso                                          ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>
#include <time.h>
#include <unistd.h>

#define MAXLONG 99

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
int RetCode;
char var0[30];
char fecha[11];
char hora[7];
char cadena[MAXLONG];
char sentencia1[MAXLONG];
char sentencia8[MAXLONG];
char sentencia9[MAXLONG];

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int amalgama(void);

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

   system("screenset");
   strcpy(sentencia1, "clear \n");
   strcpy(sentencia8, "RC=$? ; writelog \"$0 RC=$RC $USER $UID $HOSTNAME $(date +%d-%m-%Y)\" \
n");
   // strcpy(sentencia9, "echo -e \"\e[96m \" ; read -p \" *** End of Job $0 *** RC=$RC *** \"
dummy \n");

   amalgama();

   system("writelog 'amalgama script generation'");
   printf("\n\n");
   printf("\e[96m End of Program %s \n", Param[0]);
   printf("\n\n");
   return 0;

}

/*****************************************************************/
/*** Generar job amalgama.sh                                   ***/
```

```
/************************************************************/
int amalgama()
{

    printf("\t\e[93m generando amalgama.sh ... \n");

    FILE* fichero;
    fichero = fopen("./amalgama.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }

    fputs("#!/bin/bash\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs("#* Proceso: amalgama.sh                                              *\n", fichero);
    fputs("#* Sistema: GNU/Linux                                                *\n", fichero);
    fputs("#* Autor:   JCB                                                      *\n", fichero);
    fputs("#*                                                                   *\n", fichero);
    fputs("#* Remarks: Proceso que copia el contenido de los programas C en  *\n", fichero);
    fputs("#* un fichero de texto, intercalando un salto de página.          *\n", fichero);
    fputs("#*                                                                   *\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#\n", fichero);
    fputs("fileOUT=\"./amalgama.odt\"\n", fichero);
    fputs("let \"contador = 0\" \n", fichero);
    fputs("sep=\" \" \n", fichero);
    fputs("lin=\" ---marca---  \"\n", fichero);
    fputs("echo $sep > $fileOUT\n", fichero);
    fputs("#\n", fichero);
    fputs("for fileINP in *.c \n", fichero);
    fputs("   do\n", fichero);
    fputs("     let \"contador = contador + 1\"\n", fichero);
    fputs("     echo \" $contador. $fileINP \" >> $fileOUT\n", fichero);
    fputs("     echo $lin >> $fileOUT\n", fichero);
    fputs("     echo $sep >> $fileOUT\n", fichero);
    fputs("     cat $fileINP >> $fileOUT\n", fichero);
    fputs("     echo $sep >> $fileOUT\n", fichero);
    fputs("     echo $sep >> $fileOUT\n", fichero);
    fputs("     echo $sep >> $fileOUT\n", fichero);
    fputs("     echo $sep >> $fileOUT\n", fichero);
    fputs("     echo $sep >> $fileOUT\n", fichero);
    fputs("     echo -e '\014' >> $fileOUT \n", fichero);
    fputs("     echo $fileINP\n", fichero);
    fputs("   done\n", fichero);
    fputs("#\n", fichero);

    fputs(sentencia8, fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./amalgama.sh");
    RetCode = system("./amalgama.sh");
    RetCode = system("rm ./amalgama.sh");
    return 0;
}

/************************************************************************/
/***                        End Of Program                            ***/
/************************************************************************/
```

ascii.c

```
/*****************************************************************************/
/*** Programa: ascii.c                                                     ***/
/*** Sistema:  GNU/Linux - Debian                                          ***/
/*** Autor:    JCB                                                         ***/
/***                                                                       ***/
/*** Sinópsis: Visualiza el código Ascii                                   ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

/*****************************************************************************/
/*** Prototipos de las funciones                                           ***/
/*****************************************************************************/
int show_ascii(void);

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{
       system("clear; date");
       show_ascii();
       printf ("\n\n\e[96m End of program %s \e[0m\n\n", Param[0]);
       return 0;
}

/*****************************************************************************/
/*** Código ASCII                                                          ***/
/*****************************************************************************/
int show_ascii()
{
   int j;
   // char lit1[50]="\n\n \e[93m -enter- to continue...\e[0m";

   printf("\n");
   printf(" \e[96m ASCII code \e[0m \n\n");
   for (j=0; j<=127; j++)
       {
       if ( j%10 == 0)

         {
         printf("\n");
         }
        printf("\t %d %c", j, j);
       }
   return 0;
}

/*****************************************************************************/
/***                         End of Program                                ***/
/*****************************************************************************/
```

backup.c

```
/*****************************************************************************/
/*** Programa : backup.c                                                   ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Backup de directorios de usuario                             ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int f_copysec(void);

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
char cadena[99];

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{
   system("screenset");
   f_copysec();
   strcpy(cadena,"writelog 'Pgm: ");
   strcat(cadena,Param[0]);
   strcat(cadena,"'");
   system(cadena);
   printf ("\n\n\e[96m *** End of Program %s ***\e[m\n\n", Param[0]);
   return 0;
}

/*****************************************************************************/
/*** Copia de seguridad standard                                           ***/
/*****************************************************************************/
int f_copysec()
{
   // printf(" \e[92m Backup standard \e[0m \n\n");
   // system("cp -r -v -f /home/jordi/Documentos/* /media/jordi/HDDUSB");

   printf(" \e[96m Backup compressed \e[0m \n\n");
   system("tar -cpvzf \"/home/nuria/Familia_Backup_`date +%Y%m%d%H%M`.tgz\"
/home/nuria/Familia/*");
   system("tar -cpvzf \"/home/nuria/TIC_Backup_`date +%Y%m%d%H%M`.tgz\" /home/nuria/TIC/*");
   // system("tar -cpvzf \"/home/nuria/GR_Backup_`date +%Y%m%d%H%M`.tgz\" /home/nuria/GR/*");
}

/*****************************************************************************/
/***                          End Of Program                               ***/
/*****************************************************************************/
```

calendar.c

```
/*****************************************************************************/
/*** Programa : calendar.c                                                 ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Calendario anual                                             ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>
#include <time.h>
#include <locale.h>

/*****************************************************************************/
/*** Prototipos de las Funciones                                           ***/
/*****************************************************************************/
int Calendario_Anual(void);
int Obtener_Fecha_Hora(void);

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
int RetCode, i, j, dia, mes, any;
int contador;
char DiaSem[4];
char HH[3];
char MM[3];
char SS[3];
char Wmes[12][4];

/*****************************************************************************/
/*** Programa principal                                                    ***/
/*****************************************************************************/
int main()
{
  system("screenset");
  setlocale(LC_ALL, "spanish");
  Calendario_Anual();
  return 0;
}

/*****************************************************************************/
/*** Calendario Anual                                                      ***/
/*****************************************************************************/
int Calendario_Anual(void)
{

// Varibales Locales
       int dias_de_febrero, dias_del_mes, i, j;

       Obtener_Fecha_Hora();

       if (any==2022)
       { contador = 5;  }
       else if (any==2023)
       { contador = 6;  }
       else if (any==2024)
       { contador = 0;  }
       else
       { return 0;  }

   printf("\n\n");
   printf("*** %d *** \n", any);

   if ((any%4==0) && (any%100!=0) || any%400==0)
      dias_de_febrero=29;
   else
      dias_de_febrero=28;
```

```
   for(j=0;j<=11;j++)
      {

      dias_del_mes = 0;

      switch (j)
          {
          case 3: case 5: case 8: case 10:
          dias_del_mes=30;
          break;
          case 1:
          dias_del_mes=dias_de_febrero;
          break;
          case 0: case 2: case 4: case 6: case 7: case 9: case 11:
          dias_del_mes=31;
          break;
          }

        printf("\n\n");
        printf("\e[96m%s ", Wmes[j]);


        for(i=1;i<=dias_del_mes;i++)
           {

                    if (contador ==6)
               {
                           printf("\e[91m");                   // Domingos
                        contador = 0;
                        }
             else
                {
                        printf("\e[97m");       // Laborables
                        contador++;
                        }

              if ( (i==1 && j==0) || (i==6 && j==0) || \
                    (i==1 && j==4) || (i==24 && j==4) || (i==24 && j==5) || (i==15 && j==7)
|| \
                    (i==11 && j==8) || (i==24 && j==8)  || (i==12 && j==9) || (i==1 && j==10)
|| (i==6 && j==11) || \
                    (i==8 && j==11) || (i==25 && j==11) || (i==26 && j==11) )

                    {
                       printf("\e[95m");                        // Festivos anuales
                 }

          printf("%d ",i);
                 }

   }

    printf("\n\n\n");
    printf("\e[91m Domingos  - ");
    printf("\e[95m Festivos  - ");
    printf("\e[97m Laborables \n");
       printf("\n\n");

       return 0;

}

/*****************************************************************************/
/*** Obtener la fecha y la hora                                            ***/
/*****************************************************************************/
int Obtener_Fecha_Hora(void)
{

   strcpy(Wmes[0], "Ene");
   strcpy(Wmes[1], "Feb");
   strcpy(Wmes[2], "Mar");
   strcpy(Wmes[3], "Abr");
   strcpy(Wmes[4], "May");
   strcpy(Wmes[5], "Jun");
```

```
   strcpy(Wmes[6], "Jul");
   strcpy(Wmes[7], "Ago");
   strcpy(Wmes[8], "Sep");
   strcpy(Wmes[9], "Oct");
   strcpy(Wmes[10], "Nov");
   strcpy(Wmes[11], "Dic");

   time_t sisTime;
   struct tm *tiempo;
   time(&sisTime);
   tiempo=localtime(&sisTime);

// Fecha
   dia=tiempo->tm_mday;
   mes=tiempo->tm_mon;
   any=(tiempo->tm_year)+1900;

// Hora
   char Work[5];
   strftime(Work,3,"%H",tiempo);
   strcpy(HH,Work);
   strftime(Work,3,"%M",tiempo);
   strcpy(MM, Work);
   strftime(Work,3,"%S",tiempo);
   strcpy(SS, Work);
   strftime(Work,4,"%a",tiempo);
   strcpy(DiaSem, Work);

   return 0;

}
/*****************************************************************************/
/*** Final de programa                                                     ***/
/*****************************************************************************/
```

colors.c

```
/*****************************************************************************/
/*** programa : colors.c                                                   ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Muestra los posibles colores del terminal                    ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>

/*****************************************************************************/
/*** Prototipos de las funciones                                           ***/
/*****************************************************************************/
int show_colors(void);

/*****************************************************************************/
/***                           Main Program                                ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{
   system("screenset");
   show_colors();
   printf ("\n\n\e[96m End of program %s \e[0m\n\n", Param[0]);
   return 0;
}

/*****************************************************************************/
/*** Colores del terminal                                                  ***/
/*****************************************************************************/
int show_colors()
{
   int j;

   printf("\n");
   printf(" \e[96m Terminal colors \e[0m \n\n");
   for (j=0; j<=127; j++)
       {
       printf("\e[%dm color %d \e[0m ", j, j);
       }
   return 0;
}
/*****************************************************************************/
/***                          End of Program                               ***/
/*****************************************************************************/
```

compile.c

```
/*****************************************************************************/
/*** Programa: compile.c                                                   ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Compila los programas .c del directorio en curso.            ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <dirent.h>

#define MAX_LONG 99

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int f_cmp();

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
char cadena[99];

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

   system("screenset");
   printf(" \e[96m Compile C programs  \e[0m \n\n");
   f_cmp();
   strcpy(cadena,"writelog 'Pgm: ");
   strcat(cadena,Param[0]);
   strcat(cadena,"'");
   system(cadena);
   printf ("\n\n\e[96m *** End of Program %s ***\e[m\n\n", Param[0]);
   return 0;
}


/*****************************************************************************/
/*** Compila todos los programas .c del directorio en curso                ***/
/*****************************************************************************/
int f_cmp()
{
   char cadena[MAX_LONG];

   struct stat Estructura;
   struct dirent *Registro;
   DIR *WorkDir;
   int longitud;

   WorkDir=opendir(".");
   while ((Registro=readdir(WorkDir)) != NULL)
         {
         stat(Registro->d_name, &Estructura);
         if (strstr(Registro->d_name, ".c") != NULL)
            {

            strcpy(cadena, "gcc -o /usr/local/bin/");
            // strcpy(cadena, "gcc -o /home/jcb/bin/");

            longitud = strlen(Registro->d_name);
            strncat(cadena, Registro->d_name, longitud-2);
            strcat(cadena, " ");
```

```
            strcat(cadena, Registro->d_name);
            // strcat(cadena," $(mysql_config --libs)");       // para MariaDB
            printf("%s\n", cadena);
            system(cadena);
            }
         }
   closedir(WorkDir);

}

/*****************************************************************************/
/***                        End Of Program                                 ***/
/*****************************************************************************/
```

ConsultaIndices.c

```
/*****************************************************************************/
/*** Programa : ConsultaIndices.c                                          ***/
/*** Sistema  : Dev-C++ / W10                                              ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Consulta índice de una BD (SQL server)                       ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <time.h>
#include <string.h>

/*****************************************************************************/
/*** Definición de variables globales                                      ***/
/*****************************************************************************/
//                    ....
+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+....9
char sentencia[82] = "start sqlcmd -S MiServidorSQL -i ConsultaIndices.SQL -o
ConsultaIndices.TXT";
char Filename[26];
char comando1[16] = "USE MiBD;";
char comando2[3] = "GO";
char comando3[50] = "SELECT  OBJECT_NAME(IDX.OBJECT_ID) AS Table_Name,";
char comando4[24] = "IDX.name AS Index_Name,";
char comando5[37] = "IDXPS.index_type_desc AS Index_Type,";
char comando6[60] = "IDXPS.avg_fragmentation_in_percent  Fragmentation_Percentage";
char comando7[75] = "FROM sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL, NULL)
IDXPS";
char comando8[64] = "INNER JOIN sys.indexes IDX  ON IDX.object_id = IDXPS.object_id";
char comando9[34] = "AND IDX.index_id = IDXPS.index_id";
char comando10[39] = "ORDER BY Fragmentation_Percentage DESC";

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main()
{

   strncpy(Filename, "ConsultaIndices.SQL", 20);

   FILE* fichero;
   fichero = fopen(Filename, "w");
   fprintf(fichero,"%s \n",comando1);
   fprintf(fichero,"%s \n",comando2);
   fprintf(fichero,"%s \n",comando3);
   fprintf(fichero,"%s \n",comando4);
   fprintf(fichero,"%s \n",comando5);
   fprintf(fichero,"%s \n",comando6);
   fprintf(fichero,"%s \n",comando7);
   fprintf(fichero,"%s \n",comando8);
   fprintf(fichero,"%s \n",comando9);
   fprintf(fichero,"%s \n",comando10);
   fclose(fichero);

   system(sentencia);

   return 0;
}

/*****************************************************************************/
/***                         End Of Program                                ***/
/*****************************************************************************/
```

DBdump.c

```
/*****************************************************************************/
/*** Programa: DBdump.c                                                    ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Volcado de la base de datos que se le indica como            ***/
/***          argumento[1] de la función main, sobre un fichero de         ***/
/***          salida cuyo nombre será BKPaaaammddhhmmss.SQL                ***/
/***                                                                       ***/
/***          (aaaa=año mm=mes dd=dia hh=horas mm=minutos ss=segundos)      ***/
/***                                                                       ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

/*****************************************************************************/
/*** Constante globales                                                    ***/
/*****************************************************************************/
#define LONGIMAX 99

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
int RetCode;

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int DBdump(char[LONGIMAX]);
int help();

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

   system("screenset");

   if (CantParams <= 1)
      { help();
        return 4;
      }
   else
      {
      if (CantParams == 2)
         {
         DBdump(Param[1]);
         }
      }

   printf ("\n\n\e[96m End of program %s \e[m\n\n", Param[0]);
   return 0;
}

/*****************************************************************************/
/*** Pantalla de ayuda al operador                                         ***/
/*****************************************************************************/
int help()
{
   system("clear");
   printf("\n");
   printf ("\e[91m Error: Es preciso informar el nombre de una BD \e[m \n");
   printf("\n");
   return 0;
}
```

```
/*************************************************************************/
/*** Volcado de una base de datos                                      ***/
/*************************************************************************/
int DBdump(char tabla[LONGIMAX])
{
   char comando[150];

   strcpy(comando, "mysqldump -u root -p'xxxxxx' ");
   strcat(comando, tabla);
   strcat(comando, " > ./files/BKP");

   time_t tiempo = time(0);
   struct tm *tlocal = localtime(&tiempo);
   char TimeStamp[32];
   strftime(TimeStamp,32,"%Y%m%d%H%M%S",tlocal);
   strcat(comando, TimeStamp);
   strcat(comando, ".SQL");

   RetCode=system(comando);

   if (RetCode==0)
      {
       printf ("\n\e[92m %s%s%s \e[m", "Volcado finalizado sobre el fichero -->
./files/BKP",TimeStamp , ".SQL");
       system("writelog 'DBdump OK'");
      }
   else
      {
       printf ("\n\e[91m Se ha producido un error durante el volcado; revisar mensajes.\e[m");
       system("writelog 'DBdump finalizado con ERROR'");
      }

   return 0;
}

/*************************************************************************/
/***                        End Of Program                             ***/
/*************************************************************************/
```

DBload.c

```
/*****************************************************************************/
/*** Nombre Programa   : DBload.c                                          ***/
/*** Sistema Operativo : GNU/Linux                                         ***/
/*** Autor             : Jordi Calopa Bosch                                ***/
/***                                                                       ***/
/*** Remarks: Cargar registros para pruebas                                ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#include <ctype.h>
#include <unistd.h>

#define MAX_LONG 99999

/*****************************************************************************/
/*** Definición de variables globales                                      ***/
/*****************************************************************************/
int Contador, Contador2;
char cadena[MAX_LONG];

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int LoadRecord(void);

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

   system("resize -s 45 90; clear; date");

   while (Contador < 50000)
         {
         LoadRecord();
         Contador++;
         system("date");
         printf("\e[92mRegistros grabados: \e[m %d \n", Contador*Contador2);
         }

   printf ("\n\n\e[96m *** End of Program %s ***\e[m\n\n", Param[0]);
   return 0;
}



/*****************************************************************************/
/*** Inicializacion BD DMSTO (Index & Registro de Operaciones)             ***/
/*****************************************************************************/
int LoadRecord(void)
{
    Contador2 = 0;

    strcpy(cadena, "mysql -u root -e \"
");
    strcat(cadena, "USE DMSTO;
");

    while (Contador2 < 100)
    {
       strcat(cadena, " INSERT INTO T500 (T500_opertype, T500_user, T500_origen,
");
       strcat(cadena, "                   T500_languaje, T500_country, T500_region, T500_area,
T500_freesort1)    ");
       strcat(cadena, " VALUES ('SAT', 'root', 'DBload', 'ca', 'ES', '11', '12', '....
+....1....+....2....+');   ");
```

```
       Contador2++;
    }
    strcat(cadena, "\"");
    system(cadena);

    return 0;
}

/*****************************************************************************/
/***                          End Of Program                               ***/
/*****************************************************************************/
```

dmsto.c

```
/*****************************************************************************/
/*** Programa : dmsto.c                                                    ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Menú de trabajo para desarrollo                              ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#include <ctype.h>
#include <unistd.h>

#define MAX_LONG 99999

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
char resp2[2];
char cadena[MAX_LONG];
int RetCode;
char linea[125] =
"-------------------------------------------------------------------------------";


/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int FechaHora(void);
int f_endofpage(void);
int DBschema(void);
int DBrestore(void);

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

   while (1)
      {

         system("screenset");

      printf("\n\n\n");
      printf("\t\t\t \e[96m CONFIGURACION                      \e[0m\n");
      printf("\n");
      printf("\t\t\t \e[93m td\e[97m - Tipos de documento      \e[0m\n");
      printf("\t\t\t \e[93m ts\e[97m - Soportes                \e[0m\n");
      printf("\t\t\t \e[93m tl\e[97m - Niveles de seguridad    \e[0m\n");
      printf("\t\t\t \e[93m ti\e[97m - Idiomas                 \e[0m\n");
      printf("\t\t\t \e[93m te\e[97m - Empresas                \e[0m\n");
      printf("\t\t\t \e[93m tg\e[97m - Grupos                  \e[0m\n");
      printf("\t\t\t \e[93m tu\e[97m - Usuarios                \e[0m\n");
      printf("\n\n\n");
      printf("\t\t\t \e[96m MANTENIMIENTO                      \e[0m\n");
      printf("\n");
      printf("\t\t\t \e[93m is\e[97m - Information Schema      \e[0m\n");
      printf("\t\t\t \e[93m db\e[97m - Backup de la BD         \e[0m\n");
      printf("\t\t\t \e[93m dr\e[97m - Restore de la BD        \e[0m\n");
      printf("\t\t\t \e[93m lf\e[97m - Visualizar logfile      \e[0m\n");

      printf("\n\n");
      printf("\t\t\t \e[93m zz\e[97m - Exit                    \e[0m\n");
      printf("\n\n" );
      printf("\t\t \e[92m     ==> \e[0m" );
      // resp = getchar();
      scanf("%s",resp2);
```

```
      // f_topofpage();

      if (strcmp(resp2,"is")==0)
         { DBschema(); }

      else if (strcmp(resp2,"db")==0 )
         { system("DMSTODMP DMSTO; DMSTOLOG 'Backup BD DMSTO'"); }

      else if (strcmp(resp2,"dr")==0)
         { DBrestore(); }

      else if (strcmp(resp2,"lf")==0 )
         { system("printlog"); }

      else if (strcmp(resp2,"td")==0 )
         { system("triggerSQL 'SELECT * FROM TIPSD'; triggerSQL 'select count(*) from TIPSD;'
"); }

      else if (strcmp(resp2,"ts")==0 )
         { system("triggerSQL 'SELECT * FROM TDEVS'; triggerSQL 'select count(*) from TDEVS;'
"); }

      else if (strcmp(resp2,"tl")==0 )
         { system("triggerSQL 'SELECT * FROM TLVLS'; triggerSQL 'select count(*) from TLVLS;'
"); }

      else if (strcmp(resp2,"ti")==0 )
         { system("triggerSQL 'SELECT * FROM TLANG'; triggerSQL 'select count(*) from TLANG;'
"); }

      else if (strcmp(resp2,"te")==0 )
         { system("triggerSQL 'SELECT * FROM TEMPS'; triggerSQL 'select count(*) from TEMPS;'
"); }

      else if (strcmp(resp2,"tg")==0 )
         { system("triggerSQL 'SELECT * FROM TGRUP'; triggerSQL 'select count(*) from TGRUP;'
"); }

      else if (strcmp(resp2,"tu")==0 )
         { system("triggerSQL 'SELECT * FROM TUSER'; triggerSQL 'select count(*) from TUSER;'
"); }

      else if (strcmp(resp2,"zz")==0 )
         { break; }

      else
         {
         // break;
         }

      f_endofpage();
     }

   system("clear");
   printf ("\n\n\e[96m *** End of Program %s ***\e[m\n\n", Param[0]);
   return 0;
}


/*****************************************************************************/
/*** Final de pantalla                                                     ***/
/*****************************************************************************/
int f_endofpage(void)
{
   printf("\n\n");
   printf("%s\n", linea);
   printf("\e[96m  Press -enter- to continue ...\e[0m  \n");
   getchar();
   getchar();
   return 0;
}

/*****************************************************************************/
/*** Obtener Fecha y hora del sistema                                      ***/
/*****************************************************************************/
```

```
int FechaHora(void)
{
   time_t t;
   struct tm *tm;
   char fechayhora[100];
   char *meses[12]={"Ene", "Feb", "Mar", "Abr", "May", "Jun", "Jul", "Ago", "Sep", "Oct",
"Nov", "Dic"};
   t=time(NULL);
   tm=localtime(&t);
   printf (" %02d/%s/%d  %d:%02d:%02d \n", tm->tm_mday, meses[tm->tm_mon], 1900+tm->tm_year,
tm->tm_hour, tm->tm_min, tm->tm_sec);
   return 0;
}


/*****************************************************************************/
/*** Restore de la base de datos                                           ***/
/*****************************************************************************/
int DBrestore(void)
{

char respuesta[9];

   printf ("\n\e[92m   Si desea restaurar la BD DMSTO, desde el fichero RESTORE.SQL, teclee la
palabra \e[93mRESTORE \e[m\n\n");
   scanf("%s",respuesta);
   if (strcmp(respuesta,"RESTORE")==0)
      {
      system("clear");
      printf ("\n\n\e[92m Iniciando restore de la base de datos DMSTO ... \e[m\n\n");
      system("mysql -u root -p'xxxxxx' DMSTO < ./files/RESTORE.SQL");
      printf ("\n\e[92m Restore Finalizado.\e[m");
      system("writelog 'Restore base de datos DMSTO'");
      return 0;
      }
   else
      {
      system("clear");
      printf ("\n\n\e[91m Process cancelled due to operator intervetion \e[m\n\n");
      return 4;
      }
}

/*****************************************************************************/
/*** Visualizar esquema de la base de datos                                ***/
/*****************************************************************************/
int DBschema(void)
{
    strcpy(cadena, "mysql -u root -p'xxxxxx' -e \"                          ");
    strcat(cadena, " USE INFORMATION_SCHEMA ;                    ");
    strcat(cadena, " SELECT TABLE_NAME,                          ");
    strcat(cadena, "        COLUMN_NAME,                         ");
    strcat(cadena, "        COLUMN_KEY,                          ");
    strcat(cadena, "        DATA_TYPE,                           ");
    strcat(cadena, "        IS_NULLABLE,                         ");
    strcat(cadena, "        CHARACTER_MAXIMUM_LENGTH             ");
    strcat(cadena, " FROM INFORMATION_SCHEMA.COLUMNS             ");
    strcat(cadena, " WHERE TABLE_SCHEMA='DMSTO'                  ");
    strcat(cadena, " ORDER BY TABLE_NAME, ORDINAL_POSITION       ");
    strcat(cadena, "\"");
    system(cadena);
    system("writelog 'Information Schema base de datos DMSTO'");
    return 0;
}


/*****************************************************************************/
/***                        End Of Program                                 ***/
/*****************************************************************************/
```

dmstodmp.c

```
/*****************************************************************************/
/*** Programa: DMSTODMP.c                                                  ***/
/*** Sistema:  GNU/Linux - Debian                                          ***/
/*** Autor:    JCB                                                         ***/
/***                                                                       ***/
/*** Sinopsis: Volcado de la base de datos que se le indica como           ***/
/***           argumento[1] de la función main, sobre un fichero de        ***/
/***           salida cuyo nombre será BKPaaaammddhhmmss.SQL                ***/
/***                                                                       ***/
/***           (aaaa=año mm=mes dd=dia hh=horas mm=minutos ss=segundos)     ***/
/***                                                                       ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

/*****************************************************************************/
/*** Definición de constante globales                                      ***/
/*****************************************************************************/
#define LONGIMAX 99

/*****************************************************************************/
/*** Definición de variable globales                                       ***/
/*****************************************************************************/
int RetCode;

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int DMSTODMP(char[LONGIMAX]);
int help();

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

   system("clear; date");

   if (CantParams <= 1)
      { help();
        return 4;
      }
   else
      {
      if (CantParams == 2)
         {
         DMSTODMP(Param[1]);
         }
      }

   printf ("\n\n\e[96m End of program %s \e[m\n\n", Param[0]);
   return 0;
}

/*****************************************************************************/
/*** Pantalla de ayuda al operador                                         ***/
/*****************************************************************************/
int help()
{
   system("clear");
   printf("\n");
   printf ("\e[91m Error: Es preciso informar el nombre de una BD \e[m \n");
   printf("\n");
   return 0;
}
```

```
/*****************************************************************************/
/*** Volcado de una base de datos                                          ***/
/*****************************************************************************/
int DMSTODMP(char tabla[LONGIMAX])
{
   char comando[150];

   strcpy(comando, "mysqldump -u root ");
   strcat(comando, tabla);
   strcat(comando, " > ./files/BKP");

   time_t tiempo = time(0);
   struct tm *tlocal = localtime(&tiempo);
   char TimeStamp[32];
   strftime(TimeStamp,32,"%Y%m%d%H%M%S",tlocal);
   strcat(comando, TimeStamp);
   strcat(comando, ".SQL");

   RetCode=system(comando);

   if (RetCode==0)
      {
       printf ("\n\e[92m %s%s%s \e[m", "Volcado finalizado sobre el fichero -->
./files/BKP",TimeStamp , ".SQL");
       system("writelog 'DMSTODMP OK'");
      }
   else
      {
       printf ("\n\e[91m Se ha producido un error durante el volcado; revisar mensajes.\e[m");
       system("writelog 'DMSTODMP finalizado con ERROR'");
      }

   return 0;
}

/*****************************************************************************/
/***                          End Of Program                               ***/
/*****************************************************************************/
```

dmstoini.c

```
/*****************************************************************************/
/*** Nombre Programa   : DMSTOINI.c                                        ***/
/*** Sistema Operativo : GNU/Linux                                         ***/
/*** Autor             : JCB                                               ***/
/***                                                                       ***/
/*** Remarks: Inicializacion de la base de datos DMSTO                     ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#include <ctype.h>
#include <unistd.h>

#define MAX_LONG 99999

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
char cadena[MAX_LONG];
int RetCode;

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int Inicializar_Tablas(void);
int DBschema(void);

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

    if (CantParams > 1)
       {
       strcpy(cadena, "DMSTOSQL 'TRUNCATE TABLE ");
       strcat(cadena, Param[1]);
       strcat(cadena, "'");
       system(cadena);
       }

    else
       {
       Inicializar_Tablas();
       DBschema();
       system("DMSTOLOG 'Inicializacion base de datos DMSTO'");
       printf ("\n\n\e[96m *** End of Program %s ***\e[m\n\n", Param[0]);
       }

   return 0;
}


/*****************************************************************************/
/*** Carga inicial de las tablas                                           ***/
/*****************************************************************************/
int Inicializar_Tablas(void)
{
    printf ("\n\e[93m Inicializando la base de datos DMSTO. Por favor, espere ...\e[0m\n");
    usleep(1);

    strcpy(cadena, "mysql -u root -p'xxxxxx' -e \"                                  ");
    // strcat(cadena, "DROP DATABASE DMSTO;                                         ");
    // strcat(cadena, "CREATE DATABASE DMSTO;                                       ");
    strcat(cadena, "USE DMSTO;                                                     ");

    strcat(cadena, "/* ------------------------------------------------------- */; ");
    strcat(cadena, "/* TDOCS                                                    */; ");
```

```
strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "   DROP TABLE IF EXISTS TDOCS;                                   ");
strcat(cadena, "   CREATE TABLE TDOCS (                                          ");
strcat(cadena, "      TDOCS_numdoc CHAR(15) PRIMARY KEY,                         ");
strcat(cadena, "      TDOCS_tipodoc INT,                                         ");
strcat(cadena, "      TDOCS_soporte CHAR(3),                                     ");
strcat(cadena, "      TDOCS_nivelseg INT,                                        ");
strcat(cadena, "      TDOCS_estado INT,                                          ");
strcat(cadena, "      TDOCS_idioma CHAR(2),                                      ");
strcat(cadena, "      TDOCS_empresa CHAR(4),                                     ");
strcat(cadena, "      TDOCS_grupo CHAR(3),                                       ");
strcat(cadena, "      TDOCS_usercode INT,                                        ");
strcat(cadena, "      TDOCS_flow INT,                                            ");
strcat(cadena, "      TDOCS_sinopsis CHAR(250),                                  ");
strcat(cadena, "      TDOCS_ubicacion CHAR(250)                                  ");
strcat(cadena, "      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                   ");

strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "/* TKEYS                                                      */; ");
strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "   DROP TABLE IF EXISTS TKEYS;                                   ");
strcat(cadena, "   CREATE TABLE TKEYS (                                          ");
strcat(cadena, "      TKEYS_numdoc CHAR(15),                                     ");
strcat(cadena, "      TKEYS_clave CHAR(20)                                       ");
strcat(cadena, "      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                   ");

strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "/* TLOGS                                                      */; ");
strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "   DROP TABLE IF EXISTS TLOGS;                                   ");
strcat(cadena, "   CREATE TABLE TLOGS (                                          ");
strcat(cadena, "      TLOGS_table CHAR(5),                                       ");
strcat(cadena, "      TLOGS_key CHAR(15),                                        ");
strcat(cadena, "      TLOGS_timestamp CHAR(14),                                  ");
strcat(cadena, "      TLOGS_user CHAR(25),                                       ");
strcat(cadena, "      TLOGS_oldvalue TEXT(512) DEFAULT NULL,                     ");
strcat(cadena, "      TLOGS_newvalue TEXT(512) DEFAULT NULL                      ");
strcat(cadena, "      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                   ");

strcat(cadena, "/* ---------------------------------------------------------- */;    ");
strcat(cadena, "/* TIPSD                                                      */;    ");
strcat(cadena, "/* ---------------------------------------------------------- */;    ");
strcat(cadena, "   DROP TABLE IF EXISTS TIPSD;                                      ");
strcat(cadena, "   CREATE TABLE TIPSD (                                             ");
strcat(cadena, "      TIPSD_tipodoc INT PRIMARY KEY AUTO_INCREMENT,                 ");
strcat(cadena, "      TIPSD_descrip char(40) DEFAULT NULL                           ");
strcat(cadena, "      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                      ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Albaran de entrada') ;   ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Albaran de salida') ;    ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Analisis funcional') ;   ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Analisis organico') ;    ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Certificacion ISO') ;    ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Condiciones de compra'); ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Condiciones de venta');  ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Curriculum vitae') ;     ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Diagrama') ;             ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Esquema') ;              ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Factura emitida') ;      ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Factura recibida') ;     ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Guia de instalacion');   ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Guia de usuario') ;      ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Libro') ;                ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Licencia') ;             ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Mapa') ;                 ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Manifiesto aereo') ;     ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Manifiesto maritimo');   ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Manual de referencia');  ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Memorandum') ;           ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Orden de compra') ;      ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Orden de fabricacion');  ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Orden de trabajo');      ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Parte asistencia tec.'); ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Patente');               ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Pedido de cliente');     ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Permiso de circulacion');");
```

```
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Permiso de conduccion'); ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Permiso de obra') ;      ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Plano');                 ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Presupuesto') ;          ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Repositorio Digital');   ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Reserva') ;              ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Revista cientifica');    ");
strcat(cadena, " insert into TIPSD (TIPSD_descrip) values ('Titulo academico');      ");

strcat(cadena, "/* ------------------------------------------------------- */; ");
strcat(cadena, "/* TDEVS                                                   */; ");
strcat(cadena, "/* ------------------------------------------------------- */; ");
strcat(cadena, "   DROP TABLE IF EXISTS TDEVS;                                 ");
strcat(cadena, "   CREATE TABLE TDEVS (                                        ");
strcat(cadena, "      TDEVS_soporte CHAR(3) PRIMARY KEY,                       ");
strcat(cadena, "      TDEVS_digital CHAR(1),                                   ");
strcat(cadena, "      TDEVS_descrip char(40) DEFAULT NULL                      ");
strcat(cadena, "      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                 ");
strcat(cadena, " insert into TDEVS values ('AVD','Y','Audiovisual digital') ;    ");
strcat(cadena, " insert into TDEVS values ('PDF','Y','Documento PDF') ;          ");
strcat(cadena, " insert into TDEVS values ('TXT','Y','Documento de texto') ;     ");

strcat(cadena, " insert into TDEVS values ('URL','Y','Web site') ;               ");
strcat(cadena, " insert into TDEVS values ('WEB','Y','Documento web') ;          ");
strcat(cadena, " insert into TDEVS values ('AVA','N','Audiovisual analogico') ;  ");
strcat(cadena, " insert into TDEVS values ('FOT','N','Fotografia') ;             ");
strcat(cadena, " insert into TDEVS values ('PAP','N','Papel') ;                  ");
strcat(cadena, " insert into TDEVS values ('TAP','N','Cinta magnetica') ;        ");
strcat(cadena, " insert into TDEVS values ('DDE','Y','Docs externos, no red') ;  ");

strcat(cadena, "/* ------------------------------------------------------- */; ");
strcat(cadena, "/* TLVLS                                                   */; ");
strcat(cadena, "/* ------------------------------------------------------- */; ");
strcat(cadena, "   DROP TABLE IF EXISTS TLVLS;                                 ");
strcat(cadena, "   CREATE TABLE TLVLS (                                        ");
strcat(cadena, "      TLVLS_nivelseg INT PRIMARY KEY,                          ");
strcat(cadena, "      TLVLS_descrip char(40) DEFAULT NULL                      ");
strcat(cadena, "      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                 ");
strcat(cadena, " insert into TLVLS values ('000','Acceso publico') ;             ");
strcat(cadena, " insert into TLVLS values ('100','Usuario standard') ;           ");
strcat(cadena, " insert into TLVLS values ('200','Supervisor responsable') ;     ");
strcat(cadena, " insert into TLVLS values ('300','Director de area') ;           ");

strcat(cadena, "/* ------------------------------------------------------- */; ");
strcat(cadena, "/* TLANG                                                   */; ");
strcat(cadena, "/* ------------------------------------------------------- */; ");
strcat(cadena, "   DROP TABLE IF EXISTS TLANG;                                 ");
strcat(cadena, "   CREATE TABLE TLANG (                                        ");
strcat(cadena, "      TLANG_idioma CHAR(2) PRIMARY KEY,                        ");
strcat(cadena, "      TLANG_descrip char(40) DEFAULT NULL                      ");
strcat(cadena, "      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                 ");

strcat(cadena, "   insert into TLANG values ('aa' , 'afar                    '); ");
strcat(cadena, "   insert into TLANG values ('ab' , 'abjasio o abjasiano     '); ");
strcat(cadena, "   insert into TLANG values ('ae' , 'avestico                '); ");
strcat(cadena, "   insert into TLANG values ('af' , 'afrikaans               '); ");
strcat(cadena, "   insert into TLANG values ('ak' , 'akano                   '); ");
strcat(cadena, "   insert into TLANG values ('am' , 'amharico                '); ");
strcat(cadena, "   insert into TLANG values ('an' , 'aragones                '); ");
strcat(cadena, "   insert into TLANG values ('ar' , 'arabe                   '); ");
strcat(cadena, "   insert into TLANG values ('as' , 'asames                  '); ");
strcat(cadena, "   insert into TLANG values ('av' , 'avar o avaro            '); ");
strcat(cadena, "   insert into TLANG values ('ay' , 'aimara                  '); ");
strcat(cadena, "   insert into TLANG values ('az' , 'azeri                   '); ");
strcat(cadena, "   insert into TLANG values ('ba' , 'baskir                  '); ");
strcat(cadena, "   insert into TLANG values ('be' , 'bielorruso              '); ");
strcat(cadena, "   insert into TLANG values ('bg' , 'bulgaro                 '); ");
strcat(cadena, "   insert into TLANG values ('bh' , 'bhoyapuri               '); ");
strcat(cadena, "   insert into TLANG values ('bi' , 'bislama                 '); ");
strcat(cadena, "   insert into TLANG values ('bm' , 'bambara                 '); ");
strcat(cadena, "   insert into TLANG values ('bn' , 'bengali                 '); ");
strcat(cadena, "   insert into TLANG values ('bo' , 'tibetano                '); ");
strcat(cadena, "   insert into TLANG values ('br' , 'breton                  '); ");
strcat(cadena, "   insert into TLANG values ('bs' , 'bosnio                  '); ");
strcat(cadena, "   insert into TLANG values ('ca' , 'catalan                 '); ");
```

```
strcat(cadena, "   insert into TLANG values ('ce' , 'checheno                 '); ");
strcat(cadena, "   insert into TLANG values ('ch' , 'chamorro                 '); ");
strcat(cadena, "   insert into TLANG values ('co' , 'corso                    '); ");
strcat(cadena, "   insert into TLANG values ('cr' , 'cree                     '); ");
strcat(cadena, "   insert into TLANG values ('cs' , 'checo                    '); ");
strcat(cadena, "   insert into TLANG values ('cu' , 'eslavo eclesiastico      '); ");
strcat(cadena, "   insert into TLANG values ('cv' , 'chuvasio                 '); ");
strcat(cadena, "   insert into TLANG values ('cy' , 'gales                    '); ");
strcat(cadena, "   insert into TLANG values ('da' , 'danes                    '); ");
strcat(cadena, "   insert into TLANG values ('de' , 'aleman                   '); ");
strcat(cadena, "   insert into TLANG values ('dv' , 'maldivo o dhivehi        '); ");
strcat(cadena, "   insert into TLANG values ('dz' , 'dzongkha                 '); ");
strcat(cadena, "   insert into TLANG values ('ee' , 'ewe                      '); ");
strcat(cadena, "   insert into TLANG values ('el' , 'griego moderno           '); ");
strcat(cadena, "   insert into TLANG values ('en' , 'ingles                   '); ");
strcat(cadena, "   insert into TLANG values ('eo' , 'esperanto                '); ");
strcat(cadena, "   insert into TLANG values ('es' , 'espanol                  '); ");
strcat(cadena, "   insert into TLANG values ('et' , 'estonio                  '); ");
strcat(cadena, "   insert into TLANG values ('eu' , 'euskera                  '); ");
strcat(cadena, "   insert into TLANG values ('fa' , 'persa                    '); ");
strcat(cadena, "   insert into TLANG values ('ff' , 'fula                     '); ");
strcat(cadena, "   insert into TLANG values ('fi' , 'fines o finlandes        '); ");
strcat(cadena, "   insert into TLANG values ('fj' , 'fiyiano o fiyi           '); ");
strcat(cadena, "   insert into TLANG values ('fo' , 'feroes                   '); ");
strcat(cadena, "   insert into TLANG values ('fr' , 'frances                  '); ");
strcat(cadena, "   insert into TLANG values ('fy' , 'frison o frisio          '); ");
strcat(cadena, "   insert into TLANG values ('ga' , 'irlandes o gaelico       '); ");
strcat(cadena, "   insert into TLANG values ('gd' , 'gaelico escoces          '); ");
strcat(cadena, "   insert into TLANG values ('gl' , 'gallego                  '); ");
strcat(cadena, "   insert into TLANG values ('gn' , 'guarani                  '); ");
strcat(cadena, "   insert into TLANG values ('gu' , 'guyarati                 '); ");
strcat(cadena, "   insert into TLANG values ('gv' , 'manes o gaelico manes     '); ");
strcat(cadena, "   insert into TLANG values ('ha' , 'hausa                    '); ");
strcat(cadena, "   insert into TLANG values ('he' , 'hebreo                   '); ");
strcat(cadena, "   insert into TLANG values ('hi' , 'hindi o hindu            '); ");
strcat(cadena, "   insert into TLANG values ('ho' , 'hiri motu                '); ");
strcat(cadena, "   insert into TLANG values ('hr' , 'croata                   '); ");
strcat(cadena, "   insert into TLANG values ('ht' , 'haitiano                 '); ");
strcat(cadena, "   insert into TLANG values ('hu' , 'hungaro                  '); ");
strcat(cadena, "   insert into TLANG values ('hy' , 'armenio                  '); ");
strcat(cadena, "   insert into TLANG values ('ia' , 'interlingua              '); ");
strcat(cadena, "   insert into TLANG values ('id' , 'indonesio                '); ");
strcat(cadena, "   insert into TLANG values ('ie' , 'occidental               '); ");
strcat(cadena, "   insert into TLANG values ('ig' , 'igbo                     '); ");
strcat(cadena, "   insert into TLANG values ('ii' , 'yi de Sichuan            '); ");
strcat(cadena, "   insert into TLANG values ('ik' , 'inupiaq                  '); ");
strcat(cadena, "   insert into TLANG values ('io' , 'ido                      '); ");
strcat(cadena, "   insert into TLANG values ('is' , 'islandes                 '); ");

strcat(cadena, "   insert into TLANG values ('it' , 'italiano                 '); ");
strcat(cadena, "   insert into TLANG values ('iu' , 'inuktitut o inuit        '); ");
strcat(cadena, "   insert into TLANG values ('ja' , 'japones                  '); ");
strcat(cadena, "   insert into TLANG values ('jv' , 'javanes                  '); ");
strcat(cadena, "   insert into TLANG values ('ka' , 'georgiano                '); ");
strcat(cadena, "   insert into TLANG values ('kg' , 'kongo o kikongo          '); ");
strcat(cadena, "   insert into TLANG values ('ki' , 'kikuyu                   '); ");
strcat(cadena, "   insert into TLANG values ('kj' , 'kuanyama                 '); ");
strcat(cadena, "   insert into TLANG values ('kk' , 'kazajo o kazajio         '); ");
strcat(cadena, "   insert into TLANG values ('kl' , 'groenlandes o kalaallisut '); ");
strcat(cadena, "   insert into TLANG values ('km' , 'camboyano o jemer        '); ");
strcat(cadena, "   insert into TLANG values ('kn' , 'canares                  '); ");
strcat(cadena, "   insert into TLANG values ('ko' , 'coreano                  '); ");
strcat(cadena, "   insert into TLANG values ('kr' , 'kanuri                   '); ");
strcat(cadena, "   insert into TLANG values ('ks' , 'cachemiro o cachemir     '); ");
strcat(cadena, "   insert into TLANG values ('ku' , 'kurdo                    '); ");
strcat(cadena, "   insert into TLANG values ('kv' , 'komi                     '); ");
strcat(cadena, "   insert into TLANG values ('kw' , 'cornico                  '); ");
strcat(cadena, "   insert into TLANG values ('ky' , 'kirguis                  '); ");
strcat(cadena, "   insert into TLANG values ('la' , 'latin                    '); ");
strcat(cadena, "   insert into TLANG values ('lb' , 'luxemburgues             '); ");
strcat(cadena, "   insert into TLANG values ('lg' , 'luganda                  '); ");
strcat(cadena, "   insert into TLANG values ('li' , 'limburgues               '); ");
strcat(cadena, "   insert into TLANG values ('ln' , 'lingala                  '); ");
strcat(cadena, "   insert into TLANG values ('lo' , 'lao                      '); ");
strcat(cadena, "   insert into TLANG values ('lt' , 'lituano                  '); ");
```

```
    strcat(cadena, "   insert into TLANG values ('lu' , 'luba-katanga              '); ");
    strcat(cadena, "   insert into TLANG values ('lv' , 'leton                    '); ");
    strcat(cadena, "   insert into TLANG values ('mg' , 'malgache o malagasy      '); ");
    strcat(cadena, "   insert into TLANG values ('mh' , 'marshales                '); ");
    strcat(cadena, "   insert into TLANG values ('mi' , 'maori                    '); ");
    strcat(cadena, "   insert into TLANG values ('mk' , 'macedonio                '); ");
    strcat(cadena, "   insert into TLANG values ('ml' , 'malayalam                '); ");
    strcat(cadena, "   insert into TLANG values ('mn' , 'mongol                   '); ");
    strcat(cadena, "   insert into TLANG values ('mr' , 'marati                   '); ");
    strcat(cadena, "   insert into TLANG values ('ms' , 'malayo                   '); ");
    strcat(cadena, "   insert into TLANG values ('mt' , 'maltes                   '); ");
    strcat(cadena, "   insert into TLANG values ('my' , 'birmano                  '); ");
    strcat(cadena, "   insert into TLANG values ('na' , 'nauruano                 '); ");
    strcat(cadena, "   insert into TLANG values ('nb' , 'noruego bokmal           '); ");
    strcat(cadena, "   insert into TLANG values ('nd' , 'ndebele del norte        '); ");
    strcat(cadena, "   insert into TLANG values ('ne' , 'nepali                   '); ");
    strcat(cadena, "   insert into TLANG values ('ng' , 'ndonga                   '); ");
    strcat(cadena, "   insert into TLANG values ('nl' , 'neerlandes u holandes    '); ");
    strcat(cadena, "   insert into TLANG values ('nn' , 'nynorsk                  '); ");
    strcat(cadena, "   insert into TLANG values ('no' , 'noruego                  '); ");
    strcat(cadena, "   insert into TLANG values ('nr' , 'ndebele del sur          '); ");
    strcat(cadena, "   insert into TLANG values ('nv' , 'navajo                   '); ");
    strcat(cadena, "   insert into TLANG values ('ny' , 'chichewa                 '); ");
    strcat(cadena, "   insert into TLANG values ('oc' , 'occitano                 '); ");
    strcat(cadena, "   insert into TLANG values ('oj' , 'ojibwa                   '); ");
    strcat(cadena, "   insert into TLANG values ('om' , 'oromo                    '); ");
    strcat(cadena, "   insert into TLANG values ('or' , 'oriya                    '); ");
    strcat(cadena, "   insert into TLANG values ('os' , 'osetico                  '); ");
    strcat(cadena, "   insert into TLANG values ('pa' , 'panyabi o penyabi        '); ");
    strcat(cadena, "   insert into TLANG values ('pi' , 'pali                     '); ");
    strcat(cadena, "   insert into TLANG values ('pl' , 'polaco                   '); ");
    strcat(cadena, "   insert into TLANG values ('ps' , 'pastun                   '); ");
    strcat(cadena, "   insert into TLANG values ('pt' , 'portugues                '); ");
    strcat(cadena, "   insert into TLANG values ('qu' , 'quechua                  '); ");
    strcat(cadena, "   insert into TLANG values ('rm' , 'romanche                 '); ");
    strcat(cadena, "   insert into TLANG values ('rn' , 'kirundi                  '); ");
    strcat(cadena, "   insert into TLANG values ('ro' , 'rumano                   '); ");
    strcat(cadena, "   insert into TLANG values ('ru' , 'ruso                     '); ");
    strcat(cadena, "   insert into TLANG values ('rw' , 'ruandes                  '); ");
    strcat(cadena, "   insert into TLANG values ('sa' , 'sanscrito                '); ");
    strcat(cadena, "   insert into TLANG values ('sc' , 'sardo                    '); ");
    strcat(cadena, "   insert into TLANG values ('sd' , 'sindhi                   '); ");
    strcat(cadena, "   insert into TLANG values ('se' , 'sami septentrional       '); ");
    strcat(cadena, "   insert into TLANG values ('sg' , 'sango                    '); ");
    strcat(cadena, "   insert into TLANG values ('si' , 'cingales                 '); ");
    strcat(cadena, "   insert into TLANG values ('sk' , 'eslovaco                 '); ");

    strcat(cadena, "   insert into TLANG values ('sl' , 'esloveno                 '); ");
    strcat(cadena, "   insert into TLANG values ('sm' , 'samoano                  '); ");
    strcat(cadena, "   insert into TLANG values ('sn' , 'shona                    '); ");
    strcat(cadena, "   insert into TLANG values ('so' , 'somali                   '); ");
    strcat(cadena, "   insert into TLANG values ('sq' , 'albanes                  '); ");
    strcat(cadena, "   insert into TLANG values ('sr' , 'serbio                   '); ");
    strcat(cadena, "   insert into TLANG values ('ss' , 'suazi o swati            '); ");
    strcat(cadena, "   insert into TLANG values ('st' , 'sesotho                  '); ");
    strcat(cadena, "   insert into TLANG values ('su' , 'sundanes                 '); ");
    strcat(cadena, "   insert into TLANG values ('sv' , 'sueco                    '); ");
    strcat(cadena, "   insert into TLANG values ('sw' , 'suajili                  '); ");
    strcat(cadena, "   insert into TLANG values ('ta' , 'tamil                    '); ");
    strcat(cadena, "   insert into TLANG values ('te' , 'telugu                   '); ");
    strcat(cadena, "   insert into TLANG values ('tg' , 'tayiko                   '); ");
    strcat(cadena, "   insert into TLANG values ('th' , 'tailandes                '); ");
    strcat(cadena, "   insert into TLANG values ('ti' , 'tigrina                  '); ");
    strcat(cadena, "   insert into TLANG values ('tk' , 'turcomano                '); ");
    strcat(cadena, "   insert into TLANG values ('tl' , 'tagalo                   '); ");
    strcat(cadena, "   insert into TLANG values ('tn' , 'setsuana                 '); ");
    strcat(cadena, "   insert into TLANG values ('to' , 'tongano                  '); ");
    strcat(cadena, "   insert into TLANG values ('tr' , 'turco                    '); ");
    strcat(cadena, "   insert into TLANG values ('ts' , 'tsonga                   '); ");
    strcat(cadena, "   insert into TLANG values ('tt' , 'tartaro                  '); ");
    strcat(cadena, "   insert into TLANG values ('tw' , 'twi                      '); ");
    strcat(cadena, "   insert into TLANG values ('ty' , 'tahitiano                '); ");
    strcat(cadena, "   insert into TLANG values ('ug' , 'uigur                    '); ");
    strcat(cadena, "   insert into TLANG values ('uk' , 'ucraniano                '); ");
    strcat(cadena, "   insert into TLANG values ('ur' , 'urdu                     '); ");
```

```
strcat(cadena, "   insert into TLANG values ('uz' , 'uzbeko                    '); ");
strcat(cadena, "   insert into TLANG values ('ve' , 'venda                     '); ");
strcat(cadena, "   insert into TLANG values ('vi' , 'vietnamita                '); ");
strcat(cadena, "   insert into TLANG values ('vo' , 'volapuk                   '); ");
strcat(cadena, "   insert into TLANG values ('wa' , 'valon                     '); ");
strcat(cadena, "   insert into TLANG values ('wo' , 'wolof                     '); ");
strcat(cadena, "   insert into TLANG values ('xh' , 'xhosa                     '); ");
strcat(cadena, "   insert into TLANG values ('yi' , 'yidish                    '); ");
strcat(cadena, "   insert into TLANG values ('yo' , 'yoruba                    '); ");
strcat(cadena, "   insert into TLANG values ('za' , 'chuang o zhuang           '); ");
strcat(cadena, "   insert into TLANG values ('zh' , 'chino                     '); ");
strcat(cadena, "   insert into TLANG values ('zu' , 'zulu                      '); ");


strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "/* TEMPS                                                      */; ");
strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "   DROP TABLE IF EXISTS TEMPS;                                  ");
strcat(cadena, "   CREATE TABLE TEMPS (                                         ");
strcat(cadena, "      TEMPS_empresa CHAR(4) PRIMARY KEY,                        ");
strcat(cadena, "      TEMPS_descrip char(40) DEFAULT NULL                       ");
strcat(cadena, "      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                  ");
strcat(cadena, " insert into TEMPS values ('0001','Empresa 1') ;                ");
strcat(cadena, " insert into TEMPS values ('0002','Empresa 2') ;                ");
strcat(cadena, " insert into TEMPS values ('0003','Empresa 3') ;                ");
strcat(cadena, " insert into TEMPS values ('0004','Empresa 4') ;                ");
strcat(cadena, " insert into TEMPS values ('0005','Empresa 5') ;                ");

strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "/* TGRUP                                                       */; ");
strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "   DROP TABLE IF EXISTS TGRUP;                                  ");
strcat(cadena, "   CREATE TABLE TGRUP (                                         ");
strcat(cadena, "      TGRUP_grupo CHAR(3) PRIMARY KEY,                          ");
strcat(cadena, "      TGRUP_descrip char(40) DEFAULT NULL                       ");
strcat(cadena, "      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                  ");
strcat(cadena, " insert into TGRUP values ('ARQ','Arquitectura') ;              ");
strcat(cadena, " insert into TGRUP values ('BIO','Biologia') ;                  ");
strcat(cadena, " insert into TGRUP values ('EDF','Educacion Fisica') ;          ");
strcat(cadena, " insert into TGRUP values ('FIL','Filosofia') ;                 ");

strcat(cadena, " insert into TGRUP values ('FIS','Fisica') ;                    ");
strcat(cadena, " insert into TGRUP values ('GEO','Geografia') ;                 ");
strcat(cadena, " insert into TGRUP values ('GLG','Geologia') ;                  ");
strcat(cadena, " insert into TGRUP values ('HIS','Historia') ;                  ");
strcat(cadena, " insert into TGRUP values ('IDO','Informacion y Documentacion') ; ");
strcat(cadena, " insert into TGRUP values ('LIT','Literatura') ;                ");
strcat(cadena, " insert into TGRUP values ('MAT','Matematicas') ;               ");
strcat(cadena, " insert into TGRUP values ('MED','Medicina') ;                  ");
strcat(cadena, " insert into TGRUP values ('MUS','Musica') ;                    ");
strcat(cadena, " insert into TGRUP values ('QUI','Quimica') ;                   ");
strcat(cadena, " insert into TGRUP values ('RLG','Religion') ;                  ");
strcat(cadena, " insert into TGRUP values ('TIC','Informatica y Comunicaciones'); ");
strcat(cadena, " insert into TGRUP values ('***','De cualquier disciplina') ;  ");

strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "/* TUSER                                                       */; ");
strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "   DROP TABLE IF EXISTS TUSER;                                  ");
strcat(cadena, "   CREATE TABLE TUSER (                                         ");
strcat(cadena, "      TUSER_usercode INT PRIMARY KEY,                           ");
strcat(cadena, "      TUSER_nombre CHAR(50),                                    ");
strcat(cadena, "      TUSER_password CHAR(30),                                  ");
strcat(cadena, "      TUSER_nivelseg INT,                                       ");
strcat(cadena, "      TUSER_empresa CHAR(4),                                    ");
strcat(cadena, "      TUSER_grupo CHAR(3),                                      ");
strcat(cadena, "      TUSER_permiso_add BOOLEAN,                                ");
strcat(cadena, "      TUSER_permiso_shw BOOLEAN,                                ");
strcat(cadena, "      TUSER_permiso_upd BOOLEAN,                                ");
strcat(cadena, "      TUSER_permiso_chg BOOLEAN,                                ");
strcat(cadena, "      TUSER_permiso_del BOOLEAN                                 ");
strcat(cadena, "      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                  ");

strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "/* TFLOW                                                       */; ");
```

```
strcat(cadena, "/* ---------------------------------------------------------- */; ");
strcat(cadena, "   DROP TABLE IF EXISTS TFLOW;                                   ");
strcat(cadena, "   CREATE TABLE TFLOW (                                          ");
strcat(cadena, "      TFLOW_tipodoc INT PRIMARY KEY,                             ");
strcat(cadena, "      TFLOW_flow INT,                                            ");
strcat(cadena, "      TFLOW_descrip char(40) DEFAULT NULL                        ");
strcat(cadena, "      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                   ");

strcat(cadena, "/* ---------------------------------------------------------- */;   ");
strcat(cadena, "/* TPAIS                                                      */;   ");
strcat(cadena, "/* ---------------------------------------------------------- */;   ");
strcat(cadena, "   DROP TABLE IF EXISTS TPAIS;                                    ");
strcat(cadena, "   CREATE TABLE TPAIS (                                           ");
strcat(cadena, "      TPAIS_country char(2) PRIMARY KEY,                          ");
strcat(cadena, "      TPAIS_descrip char(24) DEFAULT NULL,                        ");
strcat(cadena, "      TPAIS_contry3 char(3) DEFAULT NULL                          ");
strcat(cadena, "      ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                   ");
strcat(cadena, "   insert into TPAIS values ('AD' , 'Andorra'             , 'AND'); ");
strcat(cadena, "   insert into TPAIS values ('AR' , 'Argentina'           , 'ARG'); ");
strcat(cadena, "   insert into TPAIS values ('BO' , 'Bolivia'             , 'BOL'); ");
strcat(cadena, "   insert into TPAIS values ('BR' , 'Brasil'              , 'BRA'); ");
strcat(cadena, "   insert into TPAIS values ('CL' , 'Chile'               , 'CHL'); ");
strcat(cadena, "   insert into TPAIS values ('CO' , 'Colombia'            , 'COL'); ");
strcat(cadena, "   insert into TPAIS values ('CR' , 'Costa Rica'          , 'CRI'); ");
strcat(cadena, "   insert into TPAIS values ('CU' , 'Cuba'                , 'CUB'); ");
strcat(cadena, "   insert into TPAIS values ('DO' , 'Republica Dominicana' , 'DOM'); ");
strcat(cadena, "   insert into TPAIS values ('EC' , 'Ecuador'             , 'ECU'); ");
strcat(cadena, "   insert into TPAIS values ('ES' , 'Espana'              , 'ESP'); ");
strcat(cadena, "   insert into TPAIS values ('FR' , 'Francia'             , 'FRA'); ");
strcat(cadena, "   insert into TPAIS values ('GR' , 'Grecia'              , 'GRC'); ");
strcat(cadena, "   insert into TPAIS values ('GT' , 'Guatemala'           , 'GTM'); ");
strcat(cadena, "   insert into TPAIS values ('HN' , 'Honduras'            , 'HND'); ");
strcat(cadena, "   insert into TPAIS values ('IL' , 'Israel'              , 'ISR'); ");
strcat(cadena, "   insert into TPAIS values ('IT' , 'Italia'              , 'ITA'); ");
strcat(cadena, "   insert into TPAIS values ('JP' , 'Japon'               , 'JPN'); ");
strcat(cadena, "   insert into TPAIS values ('MX' , 'Mexico'              , 'MEX'); ");
strcat(cadena, "   insert into TPAIS values ('NI' , 'Nicaragua'           , 'NIC'); ");
strcat(cadena, "   insert into TPAIS values ('PA' , 'Panama'              , 'PAN'); ");
strcat(cadena, "   insert into TPAIS values ('PE' , 'Peru'                , 'PER'); ");
strcat(cadena, "   insert into TPAIS values ('PS' , 'Palestina'           , 'PSE'); ");
strcat(cadena, "   insert into TPAIS values ('PT' , 'Portugal'            , 'PRT'); ");
strcat(cadena, "   insert into TPAIS values ('PY' , 'Paraguay'            , 'PRY'); ");
strcat(cadena, "   insert into TPAIS values ('RU' , 'Rusia'               , 'RUS'); ");
strcat(cadena, "   insert into TPAIS values ('SV' , 'El Salvador'         , 'SLV'); ");
strcat(cadena, "   insert into TPAIS values ('UA' , 'Ucrania'             , 'UKR'); ");
strcat(cadena, "   insert into TPAIS values ('UY' , 'Uruguay'             , 'URY'); ");
strcat(cadena, "   insert into TPAIS values ('VA' , 'Ciudad del Vaticano' , 'VAT'); ");
strcat(cadena, "   insert into TPAIS values ('VE' , 'Venezuela'           , 'VEN'); ");
strcat(cadena, "   insert into TPAIS values ('ZW' , 'Zimbawe'             , 'ZWE'); ");


strcat(cadena, "   /* ---------------------------------------------------------- */;");
strcat(cadena, "   /* TZONE                                                      */;");
strcat(cadena, "   /* ---------------------------------------------------------- */;");
strcat(cadena, "      DROP TABLE IF EXISTS TZONE;                                 ");
strcat(cadena, "      CREATE TABLE TZONE (                                        ");
strcat(cadena, "         TZONE_region char(2) PRIMARY KEY,                        ");
strcat(cadena, "         TZONE_descrip char(23) DEFAULT NULL                      ");
strcat(cadena, "         ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                ");
strcat(cadena, "      insert into TZONE values ('01' , 'ANDALUCIA');              ");
strcat(cadena, "      insert into TZONE values ('02' , 'ARAGON');                 ");
strcat(cadena, "      insert into TZONE values ('03' , 'PRINCIPADO DE ASTURIAS');  ");
strcat(cadena, "      insert into TZONE values ('04' , 'ISLAS BALEARES');         ");
strcat(cadena, "      insert into TZONE values ('05' , 'CANARIAS');               ");
strcat(cadena, "      insert into TZONE values ('06' , 'CANTABRIA');              ");
strcat(cadena, "      insert into TZONE values ('07' , 'CASTILLA-LA MANCHA');     ");
strcat(cadena, "      insert into TZONE values ('08' , 'CASTILLA y LEoN');        ");
strcat(cadena, "      insert into TZONE values ('09' , 'CATALUNYA');              ");
strcat(cadena, "      insert into TZONE values ('10' , 'EXTREMADURA');            ");
strcat(cadena, "      insert into TZONE values ('11' , 'GALICIA');                ");
strcat(cadena, "      insert into TZONE values ('12' , 'MADRID');                 ");
strcat(cadena, "      insert into TZONE values ('13' , 'REGION DE MURCIA');       ");
strcat(cadena, "      insert into TZONE values ('14' , 'NAVARRA');                ");
strcat(cadena, "      insert into TZONE values ('15' , 'PAIS VASCO');             ");
strcat(cadena, "      insert into TZONE values ('16' , 'LA RIOJA');               ");
```

```
strcat(cadena, "      insert into TZONE values ('17' , 'COMUNIDAD VALENCIANA');      ");
strcat(cadena, "      insert into TZONE values ('18' , 'CIUDAD DE CEUTA');           ");
strcat(cadena, "      insert into TZONE values ('19' , 'CIUDAD DE MELILLA');         ");

strcat(cadena, "   /* ---------------------------------------------------------- */;");
strcat(cadena, "   /* Areas                                                      */;");
strcat(cadena, "   /* ---------------------------------------------------------- */;");
strcat(cadena, "      DROP TABLE IF EXISTS TAREA;                                  ");
strcat(cadena, "      CREATE TABLE TAREA (                                         ");
strcat(cadena, "         TAREA_area char(2) PRIMARY KEY,                           ");
strcat(cadena, "         TAREA_descrip char(23) DEFAULT NULL                       ");
strcat(cadena, "         ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;                 ");
strcat(cadena, "      insert into TAREA values ('01' , 'Alava');                   ");
strcat(cadena, "      insert into TAREA values ('02' , 'Albacete');                ");
strcat(cadena, "      insert into TAREA values ('03' , 'Alicante');                ");
strcat(cadena, "      insert into TAREA values ('04' , 'Almeria');                 ");
strcat(cadena, "      insert into TAREA values ('05' , 'Avila');                   ");
strcat(cadena, "      insert into TAREA values ('06' , 'Badajoz');                 ");
strcat(cadena, "      insert into TAREA values ('07' , 'Baleares');                ");
strcat(cadena, "      insert into TAREA values ('08' , 'Barcelona');               ");
strcat(cadena, "      insert into TAREA values ('09' , 'Burgos');                  ");
strcat(cadena, "      insert into TAREA values ('10' , 'Caceres');                 ");
strcat(cadena, "      insert into TAREA values ('11' , 'Cadiz');                   ");
strcat(cadena, "      insert into TAREA values ('12' , 'Castellon');               ");
strcat(cadena, "      insert into TAREA values ('13' , 'Ciudad Real');             ");
strcat(cadena, "      insert into TAREA values ('14' , 'Cordoba');                 ");
strcat(cadena, "      insert into TAREA values ('15' , 'La Coruna');               ");
strcat(cadena, "      insert into TAREA values ('16' , 'Cuenca');                  ");
strcat(cadena, "      insert into TAREA values ('17' , 'Girona');                  ");
strcat(cadena, "      insert into TAREA values ('18' , 'Granada');                 ");
strcat(cadena, "      insert into TAREA values ('19' , 'Guadalajara');             ");
strcat(cadena, "      insert into TAREA values ('20' , 'Guipuzcoa');               ");
strcat(cadena, "      insert into TAREA values ('21' , 'Huelva');                  ");
strcat(cadena, "      insert into TAREA values ('22' , 'Huesca');                  ");
strcat(cadena, "      insert into TAREA values ('23' , 'Jaen');                    ");
strcat(cadena, "      insert into TAREA values ('24' , 'Leon');                    ");
strcat(cadena, "      insert into TAREA values ('25' , 'Lleida');                  ");
strcat(cadena, "      insert into TAREA values ('26' , 'La Rioja');                ");
strcat(cadena, "      insert into TAREA values ('27' , 'Lugo');                    ");
strcat(cadena, "      insert into TAREA values ('28' , 'Madrid');                  ");
strcat(cadena, "      insert into TAREA values ('29' , 'Malaga');                  ");
strcat(cadena, "      insert into TAREA values ('30' , 'Murcia');                  ");
strcat(cadena, "      insert into TAREA values ('31' , 'Navarra');                 ");
strcat(cadena, "      insert into TAREA values ('32' , 'Orense');                  ");
strcat(cadena, "      insert into TAREA values ('33' , 'Asturias');                ");
strcat(cadena, "      insert into TAREA values ('34' , 'Palencia');                ");
strcat(cadena, "      insert into TAREA values ('35' , 'Las Palmas');              ");
strcat(cadena, "      insert into TAREA values ('36' , 'Pontevedra');              ");
strcat(cadena, "      insert into TAREA values ('37' , 'Salamanca');               ");
strcat(cadena, "      insert into TAREA values ('38' , 'Santa Cruz de Tenerife');  ");
strcat(cadena, "      insert into TAREA values ('39' , 'Cantabria');               ");
strcat(cadena, "      insert into TAREA values ('40' , 'Segovia');                 ");
strcat(cadena, "      insert into TAREA values ('41' , 'Sevilla');                 ");
strcat(cadena, "      insert into TAREA values ('42' , 'Soria');                   ");
strcat(cadena, "      insert into TAREA values ('43' , 'Tarragona');               ");
strcat(cadena, "      insert into TAREA values ('44' , 'Teruel');                  ");
strcat(cadena, "      insert into TAREA values ('45' , 'Toledo');                  ");
strcat(cadena, "      insert into TAREA values ('46' , 'Valencia');                ");
strcat(cadena, "      insert into TAREA values ('47' , 'Valladolid');              ");
strcat(cadena, "      insert into TAREA values ('48' , 'Vizcaya');                 ");
strcat(cadena, "      insert into TAREA values ('49' , 'Zamora');                  ");
strcat(cadena, "      insert into TAREA values ('50' , 'Zaragoza');                ");
strcat(cadena, "      insert into TAREA values ('51' , 'Ceuta');                   ");
strcat(cadena, "      insert into TAREA values ('52' , 'Melilla');                 ");


strcat(cadena, "\"");

RetCode=system(cadena);

if (RetCode==0)
  {
   printf ("\n\e[93m Inicialización de la base de datos DMSTO finalizada OK.\e[m");
   system("writelog 'DMSTO init OK'");
  }
```

```
   else
      {
       printf ("\n\e[91m ERROR durante la inicialización; revisar mensajes.\e[m");
       system("writelog 'Carga TFLOW finalizada con ERROR'");
      }

    return 0;
}


/*****************************************************************************/
/*** Visualizar esquema de la base de datos                                ***/
/*****************************************************************************/
int DBschema(void)
{
    strcpy(cadena, "mysql -u root -e \"                          ");
    strcat(cadena, " USE INFORMATION_SCHEMA ;                    ");
    strcat(cadena, " SELECT TABLE_NAME,                          ");
    strcat(cadena, "        COLUMN_NAME,                         ");
    strcat(cadena, "        COLUMN_KEY,                          ");
    strcat(cadena, "        DATA_TYPE,                           ");
    strcat(cadena, "        IS_NULLABLE,                         ");
    strcat(cadena, "        CHARACTER_MAXIMUM_LENGTH             ");
    strcat(cadena, " FROM INFORMATION_SCHEMA.COLUMNS             ");
    strcat(cadena, " WHERE TABLE_SCHEMA='DMSTO'                  ");
    strcat(cadena, " ORDER BY TABLE_NAME, ORDINAL_POSITION       ");
    strcat(cadena, "\"");
    system(cadena);
    system("writelog 'Information Schema base de datos DMSTO'");
    return 0;
}

/*****************************************************************************/
/***                        End Of Program                                 ***/
/*****************************************************************************/
```

environ.c

```
/*****************************************************************************/
/*** Programa: environment.c                                               ***/
/*** Sistema:  GNU/Linux - Debian                                          ***/
/*** Autor:    JCB                                                         ***/
/***                                                                       ***/
/*** Remarks: Lista el contenido de alguna variables de entorno del        ***/
/***          sistema operativo, los limites de algunos tipos de datos,    ***/
/***          el código Ascii y los colores disponibles en el terminal.    ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>

/*****************************************************************************/
/*** Prototipos de las funciones                                           ***/
/*****************************************************************************/
int show_environment(void);
int show_limits(void);
int show_ascii(void);
int show_colors(void);

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{
   system("resize -s 45 90; clear; date");
   show_environment();
   show_limits();
   show_ascii();
   show_colors();
   printf ("\n\n\e[96m End of program %s \e[0m\n\n", Param[0]);
   return 0;
}

/*****************************************************************************/
/*** Algunas variables de entorno                                          ***/
/*****************************************************************************/
int show_environment()
{
   printf("\n");
   printf(" \e[93m Environment \e[0m ");
   printf("\t User ............. \e[92m%s\e[m\n", getenv("USER"));
   printf("\t\t Username ......... \e[92m%s\e[m\n", getenv("USERNAME"));
   printf("\t\t Home ............. \e[92m%s\e[m\n", getenv("HOME"));
   printf("\t\t Shell ............ \e[92m%s\e[m\n", getenv("SHELL"));
   printf("\t\t Pwd .............. \e[92m%s\e[m\n", getenv("PWD"));
   printf("\t\t Languaje ......... \e[92m%s\e[m\n", getenv("LANGUAGE"));
   printf("\t\t Lang ............. \e[92m%s\e[m\n", getenv("LANG"));
   return 0;
}

/*****************************************************************************/
/*** Límites de algunos tipos de datos                                     ***/
/*****************************************************************************/
int show_limits()
{
   printf("\n");
   printf(" \e[93m Limits \e[0m ");
   printf("\t Cantidad de bits del tipo CHAR ....... \e[92m%d\e[m\n", CHAR_BIT);
   printf("\t\t Valor min del tipo CHAR.............. \e[92m%d\e[m\n", CHAR_MIN);
   printf("\t\t Valor max del tipo CHAR .............. \e[92m%d\e[m\n", CHAR_MAX);
   printf("\t\t Valor min del tipo CHAR con signo..... \e[92m%d\e[m\n", SCHAR_MIN);
   printf("\t\t Valor max del tipo CHAR con signo..... \e[92m%d\e[m\n", SCHAR_MAX);
   printf("\t\t Valor max dei tipo CHAR sin signo..... \e[92m%d\e[m\n", UCHAR_MAX);
   printf("\t\t Valor min del tipo INT ............... \e[92m%d\e[m\n", INT_MIN);
   printf("\t\t Valor max del tipo INT ............... \e[92m%d\e[m\n", INT_MAX);
   printf("\t\t Valor min del tipo LONG .............. \e[92m%u\e[m\n", LONG_MIN);
```

```
   printf("\t\t Valor max del tipo LONG ..... \e[92m%u\e[m\n", LONG_MAX);
   printf("\t\t Valor max del tipo LONG sin signo..... \e[92m%u\e[m\n", ULONG_MAX);
   printf("\t\t Valor min del tipo SHORT ..... \e[92m%d\e[m\n", SHRT_MIN);
   printf("\t\t Valor max del tipo SHORT ..... \e[92m%d\e[m\n", SHRT_MAX);
   printf("\t\t Valor max del tipo SHORT sin signo.... \e[92m%d\e[m\n", USHRT_MAX);
   return 0;
}

/*****************************************************************************/
/*** Visualizar ASCII                                                      ***/
/*****************************************************************************/
int show_ascii()
{
   int j;
   // char lit1[50]="\n\n \e[93m -enter- to continue...\e[0m";

   printf("\n");
   printf(" \e[93m ASCII code \e[0m \n");
   for (j=33; j<=255; j++)
       {
        if ( j%8 == 0)

          {
          printf("\n");
          }
         printf("\t %d %c", j, j);
       }
   return 0;
}

/*****************************************************************************/
/*** Colores del terminal                                                  ***/
/*****************************************************************************/
int show_colors()
{
   int j;

   printf("\n\n");
   printf(" \e[93m Terminal colors \e[0m \n");
   for (j=0; j<=107; j++)
       {
       if ( j%5 == 0)
         {
         printf("\n");
         }
       printf("\e[%dm color %d \e[0m ", j, j);
       }
   return 0;
}
/*****************************************************************************/
/***                         End of Program                                ***/
/*****************************************************************************/
```

hddtest.c

```
/*****************************************************************************/
/*** Programa : hddtest.c                                                  ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Test de grabación sobre disco duro y lectura de un fichero   ***/
/***          de 10 GB aproximadamente.                                    ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>

/*****************************************************************************/
/*** Constantes globales                                                   ***/
/*****************************************************************************/
#define CANTREGS 100000000
#define DIVISOR 10000000
#define WORKFILE "./workfile.txt"
#define LONGIMAX 99

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
char resp[4];
char Registro[LONGIMAX];
char workcmd[LONGIMAX];
char cadena[99];

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int FechaHora();
int f_hdt();

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{
   system("screenset");
   f_hdt();
   strcpy(cadena,"writelog 'Pgm: ");
   strcat(cadena,Param[0]);
   strcat(cadena,"'");
   system(cadena);
   printf ("\n\n\e[96m *** End of Program %s ***\e[m\n\n", Param[0]);
   return 0;
}

/*****************************************************************************/
/*** Write/Read test                                                       ***/
/*****************************************************************************/
int f_hdt()
{

   system("screenset");

// write
   int j;

   printf("\n\e[93m");
   FechaHora();
   printf("\e[0m \n");

    FILE* FileOUT;
    FileOUT = fopen(WORKFILE, "w");
    if (FileOUT==NULL)
       {
       fputs ("*** File Open Error ***",stderr);
```

```
       return 4;
       }

    for (j=0; j<=CANTREGS; j++)
        {
        if ((j%DIVISOR)==0)
           {
           printf("\e[92mRegistros grabados: \e[m %d \n", j);
           }
        fprintf (FileOUT, "%s\n", "....
+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+....9....+.."
);
        }

    fclose(FileOUT);

// read
   int k;

   printf("\n\e[93m");
   FechaHora();
   printf("\e[0m \n");

   FILE *FileINP;
   FileINP = fopen(WORKFILE, "r");

   while(fgets(Registro, LONGIMAX, (FILE*) FileINP))
        {
        k = k+1;
        if ((k%DIVISOR)==0)
           {
           printf("\e[92mRegistros leidos: \e[m %d \n", k);
           }
        }

   fclose(FileINP);

// delete
   strcpy(workcmd,"rm ");
   strcat(workcmd, WORKFILE);
   system(workcmd);

   printf("\n\e[93m");
   FechaHora();
   printf("\e[0m \n");

   return 0;
}

/*****************************************************************************/
/*** Obtener Fecha y hora del sistema                                      ***/
/*****************************************************************************/
int FechaHora()
{
   time_t t;
   struct tm *tm;
   char fechayhora[100];
   char *meses[12]={"Ene", "Feb", "Mar", "Abr", "May", "Jun", "Jul", "Ago", "Sep", "Oct",
"Nov", "Dic"};
   t=time(NULL);
   tm=localtime(&t);
   printf (" %02d/%s/%d  %d:%02d:%02d \n", tm->tm_mday, meses[tm->tm_mon], 1900+tm->tm_year,
tm->tm_hour, tm->tm_min, tm->tm_sec);
   return 0;
}
/*****************************************************************************/
/***                          End Of Program                               ***/
/*****************************************************************************/
```

LimpiarLog.c

```
/*****************************************************************************/
/*** Programa : LimpiarLog.c                                               ***/
/*** Sistema  : Dev-C++ / W10                                              ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Limpiar LOG de una BD (SQL server)                           ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <time.h>
#include <string.h>

/*****************************************************************************/
/*** Definición de variables globales                                      ***/
/*****************************************************************************/
char sentencia[72] = "start sqlcmd -S MiServidorSQL -i LimpiarLog.SQL -o LimpiarLog.TXT";
char Filename[26];
char comando1[16] = "USE MiBD;";
char comando2[3] = "GO";
char comando3[47] = "ALTER DATABASE MiBD SET RECOVERY SIMPLE;";
char comando4[35] = "DBCC SHRINKFILE(MiBD_LOG,1);";
char comando5[45] = "ALTER DATABASE MiBD SET RECOVERY FULL;";
char comando6[22] = "sp_helpdb MiBD;";

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main()
{

   strncpy(Filename, "LimpiarLog.SQL", 15);

   FILE* fichero;
   fichero = fopen(Filename, "w");
   fprintf(fichero,"%s \n",comando1);
   fprintf(fichero,"%s \n",comando2);
   fprintf(fichero,"%s \n",comando3);
   fprintf(fichero,"%s \n",comando2);
   fprintf(fichero,"%s \n",comando4);
   fprintf(fichero,"%s \n",comando2);
   fprintf(fichero,"%s \n",comando5);
   fprintf(fichero,"%s \n",comando2);
   fprintf(fichero,"%s \n",comando6);
   fprintf(fichero,"%s \n",comando2);
   fclose(fichero);

   system(sentencia);

   return 0;
}

/*****************************************************************************/
/***                         End Of Program                                ***/
/*****************************************************************************/
```

loadtab2.c

```
/*****************************************************************************/
/*** Nombre Programa   : loadtab2.c                                        ***/
/*** Sistema Operativo : GNU/Linux                                         ***/
/*** Autor             : Jordi Calopa Bosch                                ***/
/***                                                                       ***/
/*** Remarks: Cargar registros de pruebas en la tabla de operaciones,      ***/
/***          a través de sentencias de sistema operativo.                 ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

#define MAX_LONG 99999

/*****************************************************************************/
/*** Definición de variables globales                                      ***/
/*****************************************************************************/
int Contador, j;
char cadena[MAX_LONG];

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int LoadRecord(void);

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

   while (Contador < 500000)
         {
         LoadRecord();
         Contador++;
         if (Contador%1==0)
            {
             printf("\e[92mRegistros grabados: \e[m %d \n", Contador*100);
            }
         }

   printf ("\n\n\e[96m *** End of Program %s ***\e[m\n\n", Param[0]);
   return 0;
}

/*****************************************************************************/
/*** Cargar registros de pruebas en la tabla de operaciones                ***/
/*****************************************************************************/
int LoadRecord(void)
{

    strcpy(cadena, "mysql -u root -e \"               ");
    strcat(cadena, "USE DMSTO;                        ");

    for (j=0; j<=100; j++)
        {
        strcat(cadena, " INSERT INTO T500 (T500_opertype,  ");
        strcat(cadena, "                   T500_user,      ");
        strcat(cadena, "                   T500_origen,    ");
        strcat(cadena, "                   T500_languaje,  ");
        strcat(cadena, "                   T500_country,   ");
        strcat(cadena, "                   T500_region,    ");
        strcat(cadena, "                   T500_area,      ");
        strcat(cadena, "                   T500_subarea,   ");
        strcat(cadena, "                   T500_freesort1, ");
        strcat(cadena, "                   T500_freesort2, ");
        strcat(cadena, "                   T500_desc)      ");
```

```
        strcat(cadena, " VALUES ('RSV',                         ");
        strcat(cadena, "         'root',                        ");
        strcat(cadena, "         'loadtab2',                    ");
        strcat(cadena, "         'ca',                          ");
        strcat(cadena, "         'ES',                          ");
        strcat(cadena, "         '09',                          ");
        strcat(cadena, "         '08',                          ");
        strcat(cadena, "         '08001',                       ");
        strcat(cadena, "         '....+....1....+....2....+',   ");
        strcat(cadena, "         '<<<<<<<<<< fs2 >>>>>>>>>>',   ");
        strcat(cadena, "         'registro de pruebas');        ");
        }

    strcat(cadena, "\"");
    system(cadena);

    return 0;
}

/*****************************************************************************/
/***                        End Of Program                                 ***/
/*****************************************************************************/
```

modfile.c

```
/*****************************************************************************/
/*** Programa : modfile.c                                                  ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Copia el fichero indicado en el argumento[1] sobre el        ***/
/***          fichero indicado en el argumento[2], añadiéndole un          ***/
/***          prefijo y un sufijo.                                         ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>

/*****************************************************************************/
/*** Constantes globales                                                   ***/
/*****************************************************************************/
#define MAX_LONGI 9999
#define PREFIJO "    strcat(cadena, \"      insert into TVIAS values ('"
#define SUFIJO "');    \"); "

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
char RegINP[MAX_LONGI];
char RegOUT[MAX_LONGI];

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

   system("screenset");

// Verificar cantidad de argumentos

   if (CantParams <= 2)
      {
      printf("\n ERROR: Verifique argumentos de entrada \n\n");
      return 4;
      }


// Copiar Fichero

   FILE *FENTRADA;
   FENTRADA = fopen(Param[1], "r");

   FILE *FSALIDA;
   FSALIDA = fopen(Param[2], "w");

   while (!feof(FENTRADA))
     {
       fgets(RegINP, MAX_LONGI, (FILE*) FENTRADA);

       if (strlen(RegINP)>1)
          {
          strcpy(RegOUT, PREFIJO);
          RegINP[strlen(RegINP)-1] = '\0';
          strcat(RegOUT, RegINP);
          strcat(RegOUT, SUFIJO);
          fprintf (FSALIDA, "%s\n", RegOUT);
          }
     }

   fclose(FENTRADA);
   fclose(FSALIDA);
```

```
// Final

   printf ("\n\n\e[96m *** End of Program %s ***\e[m\n\n", Param[0]);
   return 0;
}

/*****************************************************************************/
/***                          End Of Program                               ***/
/*****************************************************************************/
```

power2.c

```
/*****************************************************************************/
/*** Programa : power2.c                                                   ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Visualiza potencias de base 2                                ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

/*****************************************************************************/
/*** Prototipos de las funciones                                           ***/
/*****************************************************************************/
int show_power2(void);

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{
   system("screenset");
   show_power2();
   printf ("\n\n\e[96m End of program %s \e[0m\n\n", Param[0]);
   return 0;
}

/*****************************************************************************/
/*** Visualiza potencias de base 2                                         ***/
/*****************************************************************************/
int show_power2()
{
   int j;
   float total;

   printf("\n \e[96m Powers of 2 \e[0m \n");
   total = 1;
   printf("\n");

   for (j=1; j<=128; j++)
     {
     total = (total * 2);
     printf("\t\e[93m2 elevado a \e[m%i \e[93m=\e[m \e[92m%.09E\e[m \n", j , total);
      }

   return 0;
}
/*****************************************************************************/
/***                         End of Program                                ***/
/*****************************************************************************/
```

printlog.c

```
/*****************************************************************************/
/*** Programa : printlog.c                                                 ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Imprimir el log file                                         ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>

#define MAX_LONG 99

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
char Filename[26];
char sentencia[MAX_LONG];

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

   system("screenset");
   printf("\n \e[96m Print Log File \e[0m \n\n");
   time_t tiempo = time(0);
   struct tm *tlocal = localtime(&tiempo);
   char anymes[6];
   strftime(anymes,7,"%Y%m",tlocal);
   strcpy(Filename, "/home/jcb/logfile");
   strncat(Filename, anymes, 6);
   strncat(Filename, ".txt", 5);
   strcpy(sentencia, "cat ");
   strcat(sentencia, Filename);
   system(sentencia);
   printf ("\n\n\e[96m *** End of Program %s ***\e[m\n\n", Param[0]);
   return 0;

}

/*****************************************************************************/
/***                          End Of Program                               ***/
/*****************************************************************************/
```

Rebuild.c

```
/*****************************************************************************/
/*** Programa : Rebuild.c                                                  ***/
/*** Sistema  : Dev-C++ / W10                                              ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Reconstruir un índice de una tabla (SQL server)              ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <time.h>
#include <string.h>


/*****************************************************************************/
/*** Definición de variables globales                                      ***/
/*****************************************************************************/
char sentencia[65] = "start sqlcmd -S MiServidorSQL -i Rebuild.SQL -o Rebuild.TXT";
char Filename[26];
char comando1[16] = "USE MiBD;";
char comando2[3] = "GO";
char comando3[53] = "ALTER INDEX ALL ON [dbo].[COMPRAS] REBUILD";

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main()
{

   strncpy(Filename, "Rebuild.SQL", 12);

   FILE* fichero;
   fichero = fopen(Filename, "w");
   fprintf(fichero,"%s \n",comando1);
   fprintf(fichero,"%s \n",comando2);
   fprintf(fichero,"%s \n",comando3);
   fprintf(fichero,"%s \n",comando2);
   fclose(fichero);

   system(sentencia);

   return 0;
}

/*****************************************************************************/
/***                         End Of Program                                ***/
/*****************************************************************************/
```

screenset.c

```
/*****************************************************************************/
/*** Programa : screenset.c                                                ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Programa de visualización de la pantalla por defecto,        ***/
/***          que será lanzado por todos los programas de la aplicación.   ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int FechaHora(void);

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
char linea[125] =
"-------------------------------------------------------------------------------";

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{
        system("resize -s 50 90; clear");

        printf("\e[94m \n");
        printf(" %s\n", linea);
        printf("\e[93m  Data Management System & Testing Operations              ");
        FechaHora();
        printf("\e[94m");
        printf(" %s\n", linea);
        printf("\e[0m \n");
        return 0;
}

/*****************************************************************************/
/*** Obtener Fecha y hora del sistema                                      ***/
/*****************************************************************************/
int FechaHora(void)
{
   time_t t;
   struct tm *tm;
   char fechayhora[100];
   char *meses[12]={"Ene", "Feb", "Mar", "Abr", "May", "Jun", "Jul", "Ago", "Sep", "Oct",
"Nov", "Dic"};
   t=time(NULL);
   tm=localtime(&t);
   printf (" %02d/%s/%d  %d:%02d:%02d \n", tm->tm_mday, meses[tm->tm_mon], 1900+tm->tm_year,
tm->tm_hour, tm->tm_min, tm->tm_sec);
   return 0;
}
/*****************************************************************************/
/***                         End of Program                                ***/
/*****************************************************************************/
```

showdir.c

```
/*****************************************************************************/
/*** Programa: showdir.c                                                   ***/
/*** Sistema:  GNU/Linux - Debian                                          ***/
/*** Autor:    JCB                                                         ***/
/***                                                                       ***/
/*** Remarks: Lista el contenido del directorio actual y algunas de las    ***/
/***          variables de entorno.                                        ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>

/*****************************************************************************/
/*** Prototipos de las funciones                                           ***/
/*****************************************************************************/
int show_currentdir(void);
int show_environment(void);

/*****************************************************************************/
/*** Definición de variables globales                                      ***/
/*****************************************************************************/
char cmd1[16]="ls -l -t --color";

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{
   system("clear; date");
   show_currentdir();
   show_environment();
   printf ("\n\n\e[96m End of program %s \e[0m\n\n", Param[0]);
   return 0;
}

/*****************************************************************************/
/*** Directorio actual                                                     ***/
/*****************************************************************************/
int show_currentdir()
{
   printf("\n \e[93m Current Directory \e[0m \n\n");
   system(cmd1);

   return 0;
}

/*****************************************************************************/
/*** Algunas variables de entorno                                          ***/
/*****************************************************************************/
int show_environment()
{
   printf("\n");
   printf(" \e[93m Environment \e[0m ");
   printf("\t User ............. \e[92m%s\e[m\n", getenv("USER"));
   printf("\t\t Username ......... \e[92m%s\e[m\n", getenv("USERNAME"));
   printf("\t\t Home ............. \e[92m%s\e[m\n", getenv("HOME"));
   printf("\t\t Shell ............ \e[92m%s\e[m\n", getenv("SHELL"));
   printf("\t\t Pwd .............. \e[92m%s\e[m\n", getenv("PWD"));
   printf("\t\t Languaje ......... \e[92m%s\e[m\n", getenv("LANGUAGE"));
   printf("\t\t Lang ............. \e[92m%s\e[m\n", getenv("LANG"));
   return 0;
}
/*****************************************************************************/
/***                        End of Program                                 ***/
/*****************************************************************************/
```

showdoc.c

```
/*****************************************************************************/
/*** Programa : showdoc.c                                                  ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Programa que visualiza el documento que se le indica         ***/
/***          como argumento de la función main.                           ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <locale.h>

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
char sentencia[132];

/*****************************************************************************/
/***                            Main program                               ***/
/*****************************************************************************/
int main(int CantidadArg, char *argumento[])
 {

   setlocale(LC_ALL, "spanish");

   if (CantidadArg <2)
      {
          printf("Error: No se ha informado el nombre del documento gráfico. Pulse -enter-
para finalizar.");
          return 4;
          }

   FILE *file;
   if (file = fopen(argumento[1], "r"))
      {
      fclose(file);


      if ( strstr(argumento[1], ".jpg") == NULL && strstr(argumento[1], ".JPG") == NULL )
         {
         // en blanco
                }
         else
         {
         strcpy(sentencia, "ristretto ");   // visor de imágenes
         strcat(sentencia, argumento[1]);
         system(sentencia);
         }


      if ( strstr(argumento[1], ".pdf") == NULL && strstr(argumento[1], ".PDF") == NULL )
         {
         // en blanco
                }
         else
         {
         strcpy(sentencia, "evince ");                    // visor de documentos
         strcat(sentencia, argumento[1]);
         system(sentencia);
         }
      }

   else
      {
          printf("Error: No se encuentra el documento %s \n\n", argumento[1]);
          return 4;
      }
```

```
       return 0;
 }
/*****************************************************************************/
/***                          End of program                               ***/
/*****************************************************************************/
```

sysinfo.c

```
/*****************************************************************************/
/*** Programa : sysinfo.c                                                  ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Visualiza información del sistema                            ***/
/***                                                                       ***/
/***   sin argumentos --> Ayuda / Help                                     ***/
/***               -a --> Código ASCII                                     ***/
/***               -c --> Colores del terminal                             ***/
/***               -d --> Directorio actual                                ***/
/***               -e --> Variables de entorno                             ***/
/***               -f --> Números de coma flotante                         ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

/*****************************************************************************/
/*** Prototipos de las funciones                                           ***/
/*****************************************************************************/
int help(void);
int lsta(void);
int lstc(void);
int lstd(void);
int lste(void);
int lstf(void);

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
int j;
float total;
char cmd[16]="ls -l -t --color";

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

   system("screenset");

   if (CantParams <= 1)
      {
      help();
      return 0;
      }

   if (strcmp(Param[1],"-a")==0)
      { lsta(); }
   else if (strcmp(Param[1],"-c")==0)
      { lstc(); }
   else if (strcmp(Param[1],"-d")==0)
      { lstd(); }
   else if (strcmp(Param[1],"-e")==0)
      { lste(); }
   else if (strcmp(Param[1],"-f")==0)
      { lstf(); }
   else
      {
      help();
      }

   printf ("\n\n\e[96m *** End of Program %s ***\e[0m\n\n", Param[0]);

// Write Log
   system("writelog 'sysinfo'");
```

```
   return 0;

}

/*****************************************************************************/
/*** Ayuda al operador                                                     ***/
/*****************************************************************************/
int help()
{
   printf("\n");
   printf(" \e[92m Argumentos de entrada al programa -sysinfo- \e[0m \n\n");
   printf("          --> Ayuda / Help                        \n");
   printf("       -a --> Código ASCII                        \n");
   printf("       -c --> Colores del terminal                \n");
   printf("       -d --> Directorio actual                   \n");
   printf("       -e --> Variables de entorno                \n");
   printf("       -f --> Números de coma flotante            \n");
   printf("\e[0m\n");
   return 0;
}


/*****************************************************************************/
/*** Código ASCII                                                          ***/
/*****************************************************************************/
int lsta()
{
   printf("\n");
   printf(" \e[92m ASCII code \e[0m \n\n");
   for (j=32; j<=127; j++)             // De 0 a 31 caracteres de control
       {
       if ( j%10 == 0)

         {
         printf("\n");
         }
        printf("\t %d %c", j, j);
       }
   return 0;
}

/*****************************************************************************/
/*** Colores del terminal                                                  ***/
/*****************************************************************************/
int lstc()
{
   printf("\n");
   printf(" \e[92m Terminal colors \e[0m \n\n");
   for (j=1; j<=107; j++)
       {
       printf("\e[%dm color %d \e[0m ", j, j);
       }
   printf("\e[0m\n");
   return 0;
}

/*****************************************************************************/
/*** Directorio actual                                                     ***/
/*****************************************************************************/
int lstd()
{
   printf("\n \e[92m Current Directory \e[0m \n\n");
   system(cmd);
   return 0;
}

/*****************************************************************************/
/*** Algunas variables de entorno                                          ***/
/*****************************************************************************/
int lste()
{
   printf("\n");
   printf(" \e[93m Environment \e[0m ");
   printf("\t User ............. \e[92m%s\e[m\n", getenv("USER"));
```

```
   printf("\t\t Username ......... \e[92m%s\e[m\n", getenv("USERNAME"));
   printf("\t\t Home ............. \e[92m%s\e[m\n", getenv("HOME"));
   printf("\t\t Shell ............ \e[92m%s\e[m\n", getenv("SHELL"));
   printf("\t\t Pwd ............. \e[92m%s\e[m\n", getenv("PWD"));
   printf("\t\t Languaje ......... \e[92m%s\e[m\n", getenv("LANGUAGE"));
   printf("\t\t Lang ............. \e[92m%s\e[m\n", getenv("LANG"));
   printf("\n\e[93mwhoami\n");
   printf("\e[93m------\n");
   system("whoami");
   return 0;
}

/*****************************************************************************/
/***Número de coma flotante e impresión con notación exponencial          ***/
/*****************************************************************************/
int lstf()
{
   printf("\n \e[92m Powers of 2 \e[0m \n");
   total = 1;
   printf("\n");

   for (j=1; j<=128; j++)
     {
     total = (total * 2);
     printf("\t\e[93m2 elevado a \e[m%i \e[93m=\e[m \e[92m%.38E\e[m \n", j , total);
      }
   return 0;
}
/*****************************************************************************/
/***                         End of Program                                ***/
/*****************************************************************************/
```

triggerSQL.c

```
/*****************************************************************************/
/*** Programa : triggerSQL.c                                               ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Disparador de sentencias SQL indicadas en argumento[1]       ***/
/***          sobre la base de datos DMSTO, vía comandos del sistema.      ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

/*****************************************************************************/
/*** Constante globales                                                    ***/
/*****************************************************************************/
#define LONGIMAX 999

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
char cadena[LONGIMAX];
int RetCode;

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int disparador(char[LONGIMAX]);

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

   printf ("\n");

   if (CantParams <= 1)
      { return 4; }
   else
      { disparador(Param[1]); }

   // printf ("\n\e[96m End of program %s \e[m\n\n", Param[0]);

   return 0;
}

/*****************************************************************************/
/*** Visualizar esquema de la base de datos                                ***/
/*****************************************************************************/
int disparador(char sentencia[LONGIMAX])
{
    // Dispara sentencia
    strcpy(cadena, "mysql -u root -p'xxxxxx' -e \"    ");
    strcat(cadena, " USE DMSTO; ");
    strcat(cadena, sentencia);
    strcat(cadena, "\"");
    RetCode=system(cadena);

    if (RetCode==0)
      {
      // Registrar sentencia en log file
      strcpy(cadena, "writelog '");
      strcat(cadena, "triggerSQL --> ");
      strcat(cadena, sentencia);
      strcat(cadena, "'");
      system(cadena);
      }
   else
      {
```

```
       printf ("\n\e[91m SQL ERROR\e[m");
       system("writelog 'triggerSQL finalizado con ERROR'");
      }

    return 0;
}

/*****************************************************************************/
/***                          End Of Program                               ***/
/*****************************************************************************/
```

writejobs.c

```
/*****************************************************************************/
/*** Programa : writejobs.c                                                ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Generación de los scripts Bash                               ***/
/***                                                                       ***/
/***            chmod xyz                                                  ***/
/***                                                                       ***/
/***                  x = Propietario                                      ***/
/***                  y = Grupo                                            ***/
/***                  z = Resto                                            ***/
/***                                                                       ***/
/***            0 = --- = sin acceso                                       ***/
/***            1 = --x = ejecución                                        ***/
/***            2 = -w- = escritura                                        ***/
/***            3 = -wx = escritura y ejecución                            ***/
/***            4 = r-- = lectura                                          ***/
/***            5 = r-x = lectura y ejecución                              ***/
/***            6 = rw- = lectura y escritura                              ***/
/***            7 = rwx = lectura, escritura y ejecución                   ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>
#include <time.h>
#include <unistd.h>

#define MAXLONG 99

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
int RetCode;
char var0[30];
char fecha[11];
char hora[7];
char cadena[MAXLONG];
char sentencia1[MAXLONG];
char sentencia8[MAXLONG];
char sentencia9[MAXLONG];

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int Fecha(void);
int Hora(void);
int amlgma(void);
int bckpgm(void);
int coltrm(void);
int compil(void);
int copsec(void);
int dbback(void);
int dbrest(void);
int dsktop(void);
int histor(void);
int lanpgs(void);
int shwdir(void);
int shwdoc(void);

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

   system("screenset");
   Fecha();
```

```
   Hora();
   printf("\t\e[96m ------------------------------------------------------\n");
   printf("\t\e[96m  Jobs generator  \n");
   printf("\t\e[96m ------------------------------------------------------\e[m\n");
   printf("\n\n");

   strcpy(sentencia1, "clear \n");
   strcpy(sentencia8, "RC=$? ; writelog \"$0 RC=$RC $USER $UID $HOSTNAME $(date +%d-%m-%Y)\" \
n");
   strcpy(sentencia9, "echo -e \"\e[96m \" ; read -p \" *** End of Job $0 *** RC=$RC *** \"
dummy \n");

   amlgma();
   bckpgm();
   coltrm();
   compil();
   copsec();
   dbback();
   dbrest();
   dsktop();
   histor();
   lanpgs();
   shwdir();
   shwdoc();

   system("writelog 'genjobs - Scripts generation'");
   printf("\n\n");
   printf("\e[96m End of Program %s \n", Param[0]);
   printf("\n\n");
   return 0;

}

/*************************************************************/
/*** Genera job amlgma.sh                                  ***/
/*************************************************************/
int amlgma()
{

    printf("\t\e[93m generando j_amlgma.sh ... \n");

    FILE* fichero;
    fichero = fopen("./files/j_amlgma.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }

    fputs("#!/bin/bash\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs("#* Proceso: j_amlgma.sh                                          *\n", fichero);
    fputs("#* Sistema: GNU/Linux                                           *\n", fichero);
    fputs("#* Autor:   JCB                                                 *\n", fichero);
    fputs("#*                                                             *\n", fichero);
    fputs("#* Remarks: Proceso que copia el contenido de los scripts en un   *\n", fichero);
    fputs("#* fichero de texto, intercalando varias líneas de separación.    *\n", fichero);
    fputs("#*                                                             *\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#\n", fichero);
    fputs("fileOUT=\"./files/scripts.txt\"\n", fichero);
    fputs("let \"contador = 0\" \n", fichero);
    fputs("sep=\" \" \n", fichero);
    fputs("lin=\"-----------------------------------------\"\n", fichero);
    fputs("echo $sep > $fileOUT\n", fichero);
    fputs("#\n", fichero);
    fputs("for fileINP in *.sh \n", fichero);
    fputs("   do\n", fichero);
    fputs("     let \"contador = contador + 1\"\n", fichero);
    fputs("     echo \" $contador. $fileINP \" >> $fileOUT\n", fichero);
    fputs("     echo $lin >> $fileOUT\n", fichero);
    fputs("     echo $sep >> $fileOUT\n", fichero);
```

```
    fputs("     cat $fileINP >> $fileOUT\n", fichero);
    fputs("     echo $sep >> $fileOUT\n", fichero);
    fputs("     echo $sep >> $fileOUT\n", fichero);
    fputs("     echo $sep >> $fileOUT\n", fichero);
    fputs("     echo $sep >> $fileOUT\n", fichero);
    fputs("     echo $sep >> $fileOUT\n", fichero);
    fputs("     echo $fileINP\n", fichero);
    fputs("   done\n", fichero);
    fputs("#\n", fichero);

    fputs(sentencia8, fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./files/j_amlgma.sh");
    return 0;
}

/*************************************************************/
/*** Genera job dbback.sh                                  ***/
/*************************************************************/
int dbback()
{

    printf("\t\e[93m generando j_dbback.sh ... \n");

    FILE* fichero;
    fichero = fopen("./files/j_dbback.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }

    fputs("#!/bin/bash\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs("#* Proceso: j_dbback.sh                                              *\n", fichero);
    fputs("#* Sistema: GNU/Linux                                                *\n", fichero);
    fputs("#* Autor:   JCB                                                      *\n", fichero);
    fputs("#*                                                                  *\n", fichero);
    fputs("#* Remarks: Volcado de la base de datos JCB sobre fichero plano     *\n", fichero);
    fputs("#*                                                                  *\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#\n", fichero);
    fputs("f1=\"./files/DumpDMS_\" \n", fichero);
    fputs("f2=$(date +%Y%m%d) \n", fichero);
    fputs("f3=\".SQL\" \n", fichero);
    fputs("file=$f1$f2$f3 \n", fichero);
    fputs("echo \"Dump de la BD sobre el fichero $file\" \n", fichero);
    fputs("mysqldump -u root -pxxxxxx DMSTO > $file\n", fichero);
    fputs("if [ $? -eq 0 ]\n", fichero);
    fputs("then\n", fichero);
    fputs("   echo \"Volcado OK.\" \n", fichero);
    fputs("else\n", fichero);
    fputs("   echo \"Error durante el backup de la base de datos\" \n", fichero);
    fputs(   "read dummy\n", fichero);
    fputs("fi\n", fichero);
    fputs("#\n", fichero);

    fputs(sentencia8, fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./files/j_dbback.sh");
    return 0;
}

/*************************************************************/
/*** Genera job bckpgm.sh                                  ***/
/*************************************************************/
int bckpgm()
{
```

```
    printf("\t\e[93m generando j_bckpgm.sh ... \n");

    FILE* fichero;
    fichero = fopen("./files/j_bckpgm.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }

    fputs("#!/bin/bash\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs("#* Proceso: j_bckpgm.sh                                          *\n", fichero);
    fputs("#* Sistema: GNU/Linux                                           *\n", fichero);
    fputs("#* Autor:   JCB                                                 *\n", fichero);
    fputs("#*                                                              *\n", fichero);
    fputs("#* Remarks: Copia de respaldo de programas y scripts            *\n", fichero);
    fputs("#*                                                              *\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#\n", fichero);
    fputs("# Copia de programas\n", fichero);
    fputs("#\n", fichero);
    fputs("if test -d /home/jordi/Documentos/PgmsC\n", fichero);
    fputs("then\n", fichero);
    fputs("  echo -e \"\n\e[92mEl directorio de programas ya existe \e[m\n\"\n", fichero);
    fputs("else\n", fichero);
    fputs("  echo -e \"\n\e[92mGenerando directorio ./PgmsC \e[m\n\"\n", fichero);
    fputs("  mkdir /home/jordi/Documentos/PgmsC\n", fichero);
    fputs("fi\n", fichero);
    fputs("#\n", fichero);
    fputs("cp -r -v ./*.c /home/jordi/Documentos/PgmsC\n", fichero);
    fputs("#\n", fichero);
    fputs("# Copia de scripts\n", fichero);
    fputs("#\n", fichero);
    fputs("if test -d /home/jordi/Documentos/scripts\n", fichero);
    fputs("then\n", fichero);
    fputs("  echo -e \"\n\e[92mEl directorio de scripts ya existe \e[m\n\"\n", fichero);
    fputs("else\n", fichero);
    fputs("  echo -e \"\n\e[92mGenerando directorio ./scripts \e[m\n\"\n", fichero);
    fputs("  mkdir /home/jordi/Documentos/scripts\n", fichero);
    fputs("fi\n", fichero);
    fputs("#\n", fichero);
    fputs("cp -r -v ./*.sh /home/jordi/Documentos/scripts\n", fichero);
    fputs("#\n", fichero);

    fputs(sentencia8, fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./files/j_bckpgm.sh");
    return 0;
}



/*************************************************************/
/*** Genera job coltrm.sh                                  ***/
/*************************************************************/
int coltrm()
{

    printf("\t\e[93m generando j_coltrm.sh ... \n");

    FILE* fichero;
    fichero = fopen("./files/j_coltrm.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }
```

```
    fputs("#!/bin/bash\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs("#* Proceso: j_coltrm.sh                                          *\n", fichero);
    fputs("#* Sistema: GNU/Linux                                           *\n", fichero);
    fputs("#* Autor:   JCB                                                 *\n", fichero);
    fputs("#*                                                             *\n", fichero);
    fputs("#* Remarks: Visualiza los posibles colores del terminal         *\n", fichero);
    fputs("#*                                                             *\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#\n", fichero);
    fputs("for((contador=1; contador<=107; contador++))\n", fichero);
    fputs("   do\n", fichero);
    fputs("     if [ ${contador} -gt 29 ] && [ ${contador} -lt 38 ]; then\n", fichero);
    fputs("      echo -e \"\e[${contador}mColor ${contador}\e[0m\"\n", fichero);
    fputs("     fi\n", fichero);
    fputs("     if [ ${contador} -gt 39 ] && [ ${contador} -lt 48 ]; then\n", fichero);
    fputs("      echo -e \"\e[${contador}mColor ${contador}\e[0m\"\n", fichero);
    fputs("     fi\n", fichero);
    fputs("     if [ ${contador} -gt 89 ] && [ ${contador} -lt 98 ]; then\n", fichero);
    fputs("      echo -e \"\e[${contador}mColor ${contador}\e[0m\"\n", fichero);
    fputs("     fi\n", fichero);
    fputs("     if [ ${contador} -gt 99 ] && [ ${contador} -lt 108 ]; then\n", fichero);
    fputs("      echo -e \"\e[${contador}mColor ${contador}\e[0m\"\n", fichero);
    fputs("     fi\n", fichero);
    fputs("   done   \n", fichero);

    fputs(sentencia8, fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./files/j_coltrm.sh");
    return 0;
}



/************************************************************/
/*** Genera el job compil.sh                              ***/
/************************************************************/
int compil()
{

    printf("\t\e[93m generando j_compil.sh ... \n");

    FILE* fichero;
    fichero = fopen("./files/j_compil.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }

    fputs("#!/bin/bash\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs("#* Proceso: j_compil.sh                                          *\n", fichero);
    fputs("#* Sistema: GNU/Linux                                           *\n", fichero);
    fputs("#* Autor:   JCB                                                 *\n", fichero);
    fputs("#*                                                             *\n", fichero);
    fputs("#* Remarks: Compila todos los programas .c que encuentra en      *\n", fichero);
    fputs("#* el directorio y guarda los ejecutables en ./files/           *\n", fichero);
    fputs("#*                                                             *\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#\n", fichero);
    fputs("echo $0\n", fichero);
    fputs("prefijo=\"gcc -o ./files/\"\n", fichero);
    fputs("for filename in *.c\n", fichero);
    fputs("   do\n", fichero);
    fputs("     objeto=${filename%.c}\n", fichero);
    fputs("     comando=\"$prefijo$objeto $filename\"\n", fichero);
```

```
    fputs("     echo $comando\n", fichero);
    fputs("     $comando\n", fichero);
    fputs("   done\n", fichero);

    fputs(sentencia8, fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./files/j_compil.sh");
    return 0;
}



/*************************************************************/
/*** Genera job copsec.sh                                  ***/
/*************************************************************/
int copsec()
{

    printf("\t\e[93m generando j_copsec.sh ... \n");

    FILE* fichero;
    fichero = fopen("./files/j_copsec.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }

    fputs("#!/bin/bash\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs("#* Proceso: j_copsec.sh                                              *\n", fichero);
    fputs("#* Sistema: GNU/Linux                                                *\n", fichero);
    fputs("#* Autor:   JCB                                                      *\n", fichero);
    fputs("#*                                                                  *\n", fichero);
    fputs("#* Remarks: Copia de seguridad sobre discos HDD2 y HDD3             *\n", fichero);
    fputs("#*                                                                  *\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#\n", fichero);
    fputs("cp -r -v -f /home/jordi/Documentos/* /media/jordi/HDD2/JCB\n", fichero);
    fputs("cp -r -v -f /home/jordi/Documentos/* /media/jordi/HDD3/JCB\n", fichero);
    fputs("tar -cpvzf \"/media/jordi/HDD2/Backup_`date +%d%m%Y%H%M`.tgz\"
/home/jordi/Documentos/*\n", fichero);
    fputs("#\n", fichero);

    fputs(sentencia8, fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./files/j_copsec.sh");
    return 0;
}


/*************************************************************/
/*** Genera job dbrest.sh                                  ***/
/*************************************************************/
int dbrest()
{

    printf("\t\e[93m generando j_dbrest.sh ... \n");

    FILE* fichero;
    fichero = fopen("./files/j_dbrest.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }
```

```
    fputs("#!/bin/bash\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs("#* Proceso: j_dbrest.sh                                          *\n", fichero);
    fputs("#* Sistema: GNU/Linux                                           *\n", fichero);
    fputs("#* Autor:   JCB                                                 *\n", fichero);
    fputs("#*                                                             *\n", fichero);
    fputs("#* Remarks: Restore de la base de datos DMSTO                   *\n", fichero);
    fputs("#*                                                             *\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#                                                                    \n",
fichero);
    fputs("mysql -u root --password=xxxxxx DMSTO < ./files/RESTORE.SQL          \n",
fichero);
    fputs("#                                                                    \n",
fichero);

    fputs(sentencia8, fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./files/j_dbrest.sh");
    return 0;
}


/*************************************************************/
/*** Genera job histor.sh                                  ***/
/*************************************************************/
int histor ()
{

    printf("\t\e[93m generando j_histor.sh ... \n");

    FILE* fichero;
    fichero = fopen("./files/j_histor.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }

    fputs("#!/bin/bash\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs("#* Proceso: j_histor.sh                                          *\n", fichero);
    fputs("#* Sistema: GNU/Linux                                           *\n", fichero);
    fputs("#* Autor:   JCB                                                 *\n", fichero);
    fputs("#*                                                             *\n", fichero);
    fputs("#* Remarks: Proceso que vuelca el hist?rico bash en un fichero  *\n", fichero);
    fputs("#* que despu?s visualiza por consola.                           *\n", fichero);
    fputs("#*                                                             *\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#\n", fichero);
    fputs("cat ~/.bash_histor  > ./files/histbash.txt\n", fichero);
    fputs("cat ./files/histbash.txt\n", fichero);
    fputs("#\n", fichero);

    fputs(sentencia8, fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./files/j_histor.sh");
    return 0;
}

/*************************************************************/
/*** Genera job lanpgs.sh                                  ***/
/*************************************************************/
int lanpgs()
{
```

```
    printf("\t\e[93m generando j_lanpgs.sh ... \n");

    FILE* fichero;
    fichero = fopen("./files/j_lanpgs.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }

    fputs("#!/bin/bash\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs("#* Proceso: j_lanpgs.sh                                              *\n", fichero);
    fputs("#* Sistema: GNU/Linux                                                *\n", fichero);
    fputs("#* Autor:   JCB                                                      *\n", fichero);
    fputs("#*                                                                   *\n", fichero);
    fputs("#* Remarks: Búsqueda lenta de direcciones activas dentro de          *\n", fichero);
    fputs("#*           una red local (si no es posible utilizar nmap).         *\n", fichero);
    fputs("#*                                                                   *\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#\n", fichero);

    fputs("for i in {0..255}                    \n", fichero);
    fputs("do                                   \n", fichero);
    fputs("   h=\"192.168.1.$i\"                \n", fichero);
    fputs("   ping -c 1 -q \"$h\" &>/dev/null    \n", fichero);
    fputs("   if [ $? -eq 0 ]                   \n", fichero);
    fputs("   then                              \n", fichero);
    fputs("      echo \"$h OK\"                 \n", fichero);
    fputs("   fi                                \n", fichero);
    fputs("done                                 \n", fichero);

    fputs(sentencia8, fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./files/j_lanpgs.sh");
    return 0;
}


/*************************************************************/
/*** Genera job shwdir.sh                                  ***/
/*************************************************************/
int shwdir()
{

    printf("\t\e[93m generando j_shwdir.sh ... \n");

    FILE* fichero;
    fichero = fopen("./files/j_shwdir.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }

    fputs("#!/bin/bash\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs("#* Proceso: j_shwdir.sh                                              *\n", fichero);
    fputs("#* Sistema: GNU/Linux                                                *\n", fichero);
    fputs("#* Autor:   JCB                                                      *\n", fichero);
    fputs("#*                                                                   *\n", fichero);
    fputs("#* Remarks: Visualizar directorio de trabajo actual                  *\n", fichero);
    fputs("#*                                                                   *\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#\n", fichero);

    fputs("ls -l --color\n", fichero);
```

```
    fputs(sentencia8, fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./files/j_shwdir.sh");
    return 0;
}


/*************************************************************/
/*** Genera job shwdoc.sh                                  ***/
/*************************************************************/
int shwdoc()
{

    printf("\t\e[93m generando j_shwdoc.sh ... \n");

    FILE* fichero;
    fichero = fopen("./files/j_shwdoc.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }

    fputs("#!/bin/bash\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs("#* Proceso: j_shwdoc.sh                                              *\n", fichero);
    fputs("#* Sistema: GNU/Linux                                                *\n", fichero);
    fputs("#* Autor:   JCB                                                      *\n", fichero);
    fputs("#*                                                                  *\n", fichero);
    fputs("#* Remarks: Visualizar documento de información PDF                  *\n", fichero);
    fputs("#*                                                                  *\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#                                                                          \n",
fichero);
    fputs("evince ./docum/$1                                                          \n",
fichero);
    fputs("#                                                                          \n",
fichero);

    fputs("RC=$? ; writelog \"$0 RC=$RC $USER $UID $HOSTNAME $(date +%d-%m-%Y) $1\" \n",
fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./files/j_shwdoc.sh");
    return 0;
}


/*************************************************************/
/*** Genera job dsktop.sh                                  ***/
/*************************************************************/
int dsktop()
{

    printf("\t\e[93m generando j_dsktop.sh ... \n");

    FILE* fichero;
    fichero = fopen("./files/j_dsktop.sh", "w");

    if (fichero==NULL)
       {
       fputs ("File error",stderr);
       return 4;
       }

    fputs("#!/bin/bash\n", fichero);
    fputs("#*********************************************************************\n", fichero);
    fputs("#* Proceso: j_dsktop.sh                                              *\n", fichero);
```

```
    fputs("#* Sistema: GNU/Linux                                              *\n", fichero);
    fputs("#* Autor:   JCB                                                    *\n", fichero);
    fputs("#*                                                                 *\n", fichero);
    fputs("#* Remarks: Modificar configuración escritorio                     *\n", fichero);
    fputs("#*                                                                 *\n", fichero);
    fputs("#*****************************************************************\n", fichero);
    fputs(sentencia1, fichero);

    fputs("#                                                                    \n",
fichero);
    fputs("gsettings set org.gnome.dsktop.background primary-color '#0000FF'          \n",
fichero);
    fputs("gsettings set org.gnome.dsktop.background picture-options 'centered'       \n",
fichero);
    fputs("#                                                                    \n",
fichero);

    fputs(sentencia8, fichero);
    fputs(sentencia9, fichero);
    fclose(fichero);

    RetCode = system("chmod 755 ./files/j_dsktop.sh");
    return 0;
}

/*****************************************************************************/
/*** Obtener fecha del sistema                                             ***/
/*****************************************************************************/
int Fecha()
{
   time_t tiempo = time(0);
   struct tm *tlocal = localtime(&tiempo);
   char Dia[2];
   strftime(Dia,3,"%d",tlocal);
   strncpy(fecha, Dia, 2);
   strncat(fecha, "-", 2);
   char Mes[2];
   strftime(Mes,3,"%m",tlocal);
   strncat(fecha, Mes, 2);
   strncat(fecha, "-", 2);
   char Any[4];
   strftime(Any,5,"%Y",tlocal);
   strncat(fecha, Any, 4);
   return 0;
}

/*****************************************************************************/
/*** Obtener hora del sistema                                              ***/
/*****************************************************************************/
int Hora()
{
   time_t tiempo = time(0);
   struct tm *tlocal = localtime(&tiempo);
   char Hor[2];
   strftime(Hor,3,"%H",tlocal);
   strncpy(hora, Hor, 2);
   strncat(hora, ":", 2);
   char Min[2];
   strftime(Min,3,"%M",tlocal);
   strncat(hora, Min, 2);
   return 0;
}


/*****************************************************************************/
/***                         End Of Program                                ***/
/*****************************************************************************/
```

writelog.c

```
/*****************************************************************************/
/*** Programa : writelog.c                                                 ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Programa que permite grabar en un -logfile- el contenido     ***/
/***          del argumento 1 de la función main. El nombre del fichero    ***/
/***          log es el siguiente: ./logfileAAAAMM.txt donde AAAAMM        ***/
/***          corresponde al año y mes de la fecha.                        ***/
/***                                                                       ***/
/***          Este programa puede ser lanzado desde un proceso batch       ***/
/***          como se muestra en el siguiente ejemplo:                     ***/
/***                                                                       ***/
/***          writelog "proceso: $0 - User: $USER - UID: $UID"             ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <time.h>
#include <string.h>
#include <dirent.h>
#include <unistd.h>
#include <sys/stat.h>

#define MAX_LONG 200
#define WORKDIR "/home/jcb"

/*****************************************************************************/
/*** Variables globales                                                    ***/
/*****************************************************************************/
DIR *directorio;
char cadena[200];

/*****************************************************************************/
/*** Prototipos de Funciones                                               ***/
/*****************************************************************************/
int WriteLOG(char dummy[MAX_LONG]);

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{

// Comprueba si existe el directorio de trabajo
   directorio = opendir(WORKDIR);
   if (directorio == NULL)
      {
      printf("generando directorio ...\n");
      mkdir(WORKDIR , 0777);
      }


// Verifica que recibe al menos 1 argumento
   if (CantParams > 1)
      {
      time_t tiempo = time(0);
      struct tm *tlocal = localtime(&tiempo);
      char TimeStamp[14];
      strftime(TimeStamp,15,"%Y%m%d%H%M%S",tlocal);
      strcpy(cadena, TimeStamp);
      strcat(cadena," - ");
      strcat(cadena,getenv("USER"));
      strcat(cadena," - ");
      // strcat(cadena,getenv("LOGNAME"));
      // strcat(cadena," - ");
      strcat(cadena,Param[1]);
      WriteLOG(cadena);
      }
```

```
   else
      {
      printf("No se encuentra parámetro de entrada\n");
      return 4;
      }

   return 0;
}

/*****************************************************************/
/*** Grabar registro en logfile con un texto recibido como   ***/
/*** parámetro de entrada a la función.                      ***/
/*****************************************************************/
int WriteLOG(char texto[MAX_LONG])
{
   char Filename[26];
   int j;
   int longi;

   longi = strlen(texto);

   time_t tiempo = time(0);
   struct tm *tlocal = localtime(&tiempo);
   char anymes[6];
   strftime(anymes,7,"%Y%m",tlocal);

   strcpy(Filename, WORKDIR);
   strncat(Filename, "/logfile", 9);
   strncat(Filename, anymes, 6);
   strncat(Filename, ".txt", 5);

   FILE* fichero;
   fichero = fopen(Filename, "a");
   // fprintf(fichero,"%s", ">>> ");
   for (j=0; j<=longi-1; j++)
       { fprintf(fichero,"%c", texto[j]); }
   fprintf(fichero,"%s", "\n");
   fclose(fichero);

   return 0;
}

/*************************************************************************/
/***                        End Of Program                             ***/
/*************************************************************************/
```

zhelp000.c

```
/*****************************************************************************/
/*** Programa : zhelp000.c                                                 ***/
/*** Sistema  : GNU/Linux - OpenSuse                                       ***/
/*** Autor    : JCB                                                        ***/
/***                                                                       ***/
/*** Remarks: Información de ayuda al operador en formato HTML             ***/
/***                                                                       ***/
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

/*****************************************************************************/
/*** Prototipos de las funciones                                           ***/
/*****************************************************************************/
int generar_help();

/*****************************************************************************/
/*** Variables globales                                                   ***/
/*****************************************************************************/
int RetCode;

/*****************************************************************************/
/***                          Main Program                                 ***/
/*****************************************************************************/
int main(int CantParams, char *Param[])
{
   system("screenset");
   generar_help();
   printf ("\n\n\e[96m *** End of Program %s ***\e[m\n\n", Param[0]);
   return 0;
}

/*****************************************************************************/
/*** Generar zhelp002.html                                                 ***/
/*****************************************************************************/
int generar_help()
{
       FILE* fichero;
       fichero = fopen("zhelp002.html", "w");

       if (fichero==NULL)
          {
          printf("*** File Open error ***\n");
          getchar();
          return 4;
                }

    fprintf(fichero ,"<html lang=\"es-ES\">\n");
    fprintf(fichero ,"<head>\n");
    fprintf(fichero ,"</head>\n");

    fprintf(fichero ,"   <style>\n");
    fprintf(fichero ,"  dummy { }\n");
    fprintf(fichero ,"  body {\n");
    fprintf(fichero ,"       color: white;\n");
    fprintf(fichero ,"       background-color: #000080;\n");
    fprintf(fichero ,"       }\n");
    fprintf(fichero ,"h4 {\n");
    fprintf(fichero ,"       color: yellow;\n");
    fprintf(fichero ,"        background-color: #000080;\n");
    fprintf(fichero ,"       }\n");
    fprintf(fichero ,"h5 {\n");
    fprintf(fichero ,"       color: orange;\n");
    fprintf(fichero ,"       background-color: #000080;\n");
    fprintf(fichero ,"       }\n");
    fprintf(fichero ,"   </style>\n");

    fprintf(fichero ,"<body>  \n");
```

```
       fprintf(fichero ,"<H3> help002 </h3>
<br> \n");
    fprintf(fichero ,"<H4> Sistema de numeración decimal </H4>
\n");
    fprintf(fichero ," Simbología -->       0    1    2    3    4    5    6    7    8    9
<br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," Notación:
<br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," En este sistema de numeración posicional, para obtener un valor cada
posición se multiplica por una      <br> \n");
    fprintf(fichero ," potencia de diez, empezando por la derecha en caso de números enteros.
La primera posición (Unidades)    <br> \n");
    fprintf(fichero ," se multiplica por diez elevado a cero; La segunda posición (Decenas) se
multiplica por diez elevado      <br> \n");
       fprintf(fichero ," a uno; la tercera cifra (centenas) por diez elevado a dos y así
sucesivamente.                          <br> \n");
    fprintf(fichero ,"
<br> \n");
       fprintf(fichero ," El número 1234 equivale a:
<br> \n");
       fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," (1 x 10^3) + (2 x 10^2) + (3 x 10^1) + (4 x 10^0)  ==>  (1000) + (200)
+ (30) + (4)                      <br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," En el caso de los decimales se utiliza el mismo método pero con
exponentes negativos, empezando por      <br> \n");
    fprintf(fichero ," la izquierda a partir de la coma. El mismo modo el número 0,5678
equivale a:                           <br> \n");
       fprintf(fichero ,"
<br> \n");
    fprintf(fichero ,"   (5 x 10^-1) + (6 x 10^-2) + (7 x 10^-3) + (8 x 10^-4) ==> (0,5) +
(0,06) + (0,007) + (0,0008)          <br> \n");
       fprintf(fichero ,"
<br> \n");

    fprintf(fichero ,"<H4> Clasificación de los números </H4>
\n");
    fprintf(fichero ," <pre> <font face=\"courier new\" size=\"2\">
<br> \n");
    fprintf(fichero ,"                                                | Naturales (>=0)
<br> \n");
    fprintf(fichero ,"                                                |    (N)
<br> \n");
    fprintf(fichero ,"                                       | Enteros |
<br> \n");
    fprintf(fichero ,"                                       |         |
<br> \n");
    fprintf(fichero ,"                        | Racionales |         | Negativos (<0)
<br> \n");
    fprintf(fichero ,"                        |    (Q)     |
<br> \n");
    fprintf(fichero ,"                        |            |               | Exactos
<br> \n");
    fprintf(fichero ,"                        |            |               |
<br> \n");
    fprintf(fichero ,"             | Reales  |            | Fraccionarios  |
<br> \n");
    fprintf(fichero ,"             |   (R)   |                             |            |
Periodicos puros                    <br> \n");
    fprintf(fichero ,"             |         |                             | Inexactos  |
<br> \n");
    fprintf(fichero ,"  Complejos  |         | Irracionales                                |
Periodicos mixtos                   <br> \n");
    fprintf(fichero ,"     (C)     |
<br> \n");
    fprintf(fichero ,"             |
<br> \n");
    fprintf(fichero ,"             | Imaginarios
<br> \n");
```

```
    fprintf(fichero ,"
<br> \n");
       fprintf(fichero ,"</font> </pre>
<br> \n");

    fprintf(fichero ,"<H4> Facciones </H4>
\n");
    fprintf(fichero ," <pre> <font face=\"courier new\" size=\"2\">
<br> \n");
    fprintf(fichero ,"    Suma                    Resta                       Multiplicación
División              <br> \n");
    fprintf(fichero ,"     a     c    ad + bc      e     g    eh - fg         a     c       ac
e     g       eh      <br> \n");
    fprintf(fichero ,"    --- + --- = -------     --- - --- = -------        --- * ---  =
----          --- / ---  =  ----     <br> \n");
       fprintf(fichero ,"     b     d      bd         f     h      fh            b     d
bd            f     h       fg      <br> \n");
       fprintf(fichero ,"
<br> \n");
    fprintf(fichero ,"   Ejemplos:
<br> \n");
    fprintf(fichero ,"     2     7    10 + 21      12    2    36 - 8          3     4       12
50    30      300      <br> \n");
    fprintf(fichero ,"    --- + --- = -------     --- - --- = -------        --- * ---  =
----          --- / ---  =  ---      <br> \n");
       fprintf(fichero ,"     3     5      15         4     3      12            8     7
56            5     6      150      <br> \n");
       fprintf(fichero ,"</font> </pre>
<br> \n");
    fprintf(fichero ," Fracciones equivalentes son aquellas que una se pueden obtener una a
partir de la otra, multiplicando o  <br> \n");
    fprintf(fichero ," dividiendo el numerador y el denominador por el mismo número:
<br> \n");
    fprintf(fichero ," <pre> <font face=\"courier new\" size=\"2\">
<br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ,"   1                7            1     7     7
<br> \n");
    fprintf(fichero ,"  --- equivalente a --- dado que --- * --- = ---
<br> \n");
       fprintf(fichero ,"   2                14           2     7     14
<br> \n");
       fprintf(fichero ,"</font> </pre>
<br> \n");
       fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," Para convertir fracciones en porcentajes multiplicaremos la fracción
por 100 y resolveremos el quebrado. <br> \n");
    fprintf(fichero ," Supongamos que tenemos 98 habitaciones ocupadas en un hotel que dispone
de 140 hab.:                      <br> \n");
       fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," <pre> <font face=\"courier new\" size=\"2\">
<br> \n");
    fprintf(fichero ,"   98   100     9800
<br> \n");
    fprintf(fichero ,"  --- * ---  = -----  = 70    --> es decir el indice de ocupación es del
70 %.                           <br> \n");
       fprintf(fichero ,"  140    1      140
<br> \n");
       fprintf(fichero ,"</font> </pre>
<br> \n");

    fprintf(fichero ,"<H4> Sistema de numeración Romano </H4>
\n");
    fprintf(fichero ," <pre> <font face=\"courier new\" size=\"2\">
<br> \n");
    fprintf(fichero ," Simbología -->   I         V         X          L         C          D
M                        <br> \n");
    fprintf(fichero ,"                  Uno      Cinco      Diez    Cincuenta    Cien
Quinientos    Mil                    <br> \n");
       fprintf(fichero ,"</font> </pre>
<br> \n");
```

```
    fprintf(fichero ," Notación:
<br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," - El valor de un número se obtiene por la suma de sus símbolos
<br> \n");
    fprintf(fichero ," - Los símbolos I, X, C y M pueden repetirse hasta 3 veces consecutivas
<br> \n");
    fprintf(fichero ," - Un símbolo a la izquierda de otro inmediatamente mayor, le resta
<br> \n");
    fprintf(fichero ," - Los símbolos V, L y D no pueden repetirse y no pueden ser utilizados
para restar                        <br> \n");
    fprintf(fichero ," - I solo puede restar de V o X
<br> \n");
    fprintf(fichero ," - X solo puede restar de L o C
<br> \n");
    fprintf(fichero ," - C solo puede restar de D o M
<br> \n");
    fprintf(fichero ," - Un guión (-) encima de otro símbolo, multiplica por 1000 su valor
<br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," Ejemplos:
<br> \n");
    fprintf(fichero ," <pre> <font face=\"courier new\" size=\"2\">
<br> \n");
    fprintf(fichero ," I=1        II=2        III=3        IV=4       V=5
<br> \n");
    fprintf(fichero ," VI=6       VII=7       VIII=8       IX=9       X=10
<br> \n");
    fprintf(fichero ," XI=11      XII=12      XIII=13      XIV=14     XV=15
<br> \n");
    fprintf(fichero ," XVI=16     XVII=17     XVIII=18     XIX=19     XX=20
<br> \n");
    fprintf(fichero ," XXI=21     XXII=22     XXIII=23     XXIV=24    XXV=25
<br> \n");
    fprintf(fichero ," XXVI=26    XXVII=27    XXVIII=28    XXIX=29    XXX=30
<br> \n");
    fprintf(fichero ," XXXI=31    XXXII=32    XXXIII=33    XXXIV=34   XXXV=35
<br> \n");
    fprintf(fichero ," XXXVI=36   XXXVII=37   XXXVIII=38   XXXIX=39   XL=40
<br> \n");
    fprintf(fichero ," XLI=41     XLII=42     XLIII=43     XLIV=44    XLV=45
<br> \n");
    fprintf(fichero ," XLVI=46    XLVII=47    XLVIII=48    XLIX=49    L=50
<br> \n");
    fprintf(fichero ," LI=51      LII=52      LIII=53      LIV=54     LV=55
<br> \n");
    fprintf(fichero ," LVI=56     LVII=57     LVIII=58     LIX=59     LX=6
<br> \n");
    fprintf(fichero ," LXI=61     LXII=62     LXIII=63     LXIV=64    LXV=65
<br> \n");
    fprintf(fichero ," LXVI=66    LXVII=67    LXVIII=68    LXIX=69    LXX=70
<br> \n");
    fprintf(fichero ," LXXI=71    LXXII=72    LXXIII=73    LXXIV=74   LXXV=75
<br> \n");
    fprintf(fichero ," LXXVI=76   LXXVII=77   LXXVIII=78   LXXIX=79   LXXX=80
<br> \n");
    fprintf(fichero ," LXXXI=81   LXXXII=82   LXXXIII=83   LXXXIV=84  LXXXV=85
<br> \n");
    fprintf(fichero ," LXXXVI=86  LXXXVII=87  LXXXVIII=88  LXXXIX=89  XC=90
<br> \n");
    fprintf(fichero ," XCI=91     XCII=92     XCIII=93     XCIV=94    XCV=95
<br> \n");
    fprintf(fichero ," XCVI=96    XCVII=97    XCVIII=98    XCXIX=99   C=100
<br> \n");
    fprintf(fichero ," C=100    CC= 200   CCC=300    CD=400    D=500
<br> \n");
    fprintf(fichero ," DC=600   DCC=700   DCCC=800   CM=900    M=1000
<br> \n");
    fprintf(fichero ,"                                  __         _
<br> \n");
    fprintf(fichero ," M=1000   MM= 2000   MMM=3000    IV=4000    V=5000
<br> \n");
```

```
    fprintf(fichero ," __       ___        ____        __         _
<br> \n");
    fprintf(fichero ," VI=6000  VII=7000   VIII=8000   IX=9000    X=10000
<br> \n");
    fprintf(fichero ," </font> </pre>
<br> \n");

       fprintf(fichero ,"<H4> Combinatoria </H4>
\n");
       fprintf(fichero ,"<H5> Variaciones </H5>
<br> \n");
       fprintf(fichero ,"   Variaciones de m elementos tomados de n en n son todos los
subconjuntos de n elementos que se pueden   <br> \n");
       fprintf(fichero ,"   formar a partir de un conjunto original, en los que
consideraremos subconjuntos distintos cuando difiera <br> \n");
       fprintf(fichero ,"   bien sea en alguno de los elementos, bien sea en el orden de los
mismos. (n <= m)                      <br> \n");
    fprintf(fichero ," <pre> <font face=\"courier new\" size=\"2\">
<br> \n");
       fprintf(fichero ,"         m!                                                n
<br> \n");
       fprintf(fichero ,"   V = ------   (Sin Repetición)                VR = m     (Con
repetición)                                <br> \n");
       fprintf(fichero ,"       (m-n)!
<br> \n");
    fprintf(fichero ,"</font> </pre>
<br> \n");
       fprintf(fichero ,"<H5> Permutaciones   </H5>
<br> \n");
       fprintf(fichero ,"   Permutaciones de m elementos tomados de n en n son todos los
subconjuntos de n elementos que se pueden <br> \n");
       fprintf(fichero ,"   formar, a partir de un conjunto original, con la condición de que
los subconjuntos serán distintos cuando <br> \n");
       fprintf(fichero ,"   difiera  el orden de los mismos. Es un caso particular de
variación en que n = m, por lo que también   <br> \n");
       fprintf(fichero ,"   nos referimos a ellas como permutaciones de n elementos (Pn).
<br> \n");
    fprintf(fichero ," <pre> <font face=\"courier new\" size=\"2\">
<br> \n");
       fprintf(fichero ,"   Pn = n!
<br> \n");
    fprintf(fichero ," </font> </pre>
<br> \n");
       fprintf(fichero ,"<H5> Combinaciones   </H5>
<br> \n");
       fprintf(fichero ,"   Combinaciones de m elementos tomados de n en n son todos los
subconjuntos de n elementos que se pueden <br> \n");
       fprintf(fichero ,"   formar, a partir de un conjunto original, con la condición de que
los subconjuntos serán distintos cuando <br> \n");
       fprintf(fichero ,"   difiera alguno de sus elementos.
<br> \n");
    fprintf(fichero ," <pre> <font face=\"courier new\" size=\"2\">
<br> \n");
       fprintf(fichero ,"             m!                                    (m+n-1)!
<br> \n");
       fprintf(fichero ,"   C =  -----------   (Sin Repetición)    CR = -----------   (Con
repetición)                              <br> \n");
       fprintf(fichero ,"        (m-n)! * n!                              (m-1)! * n!
<br> \n");
    fprintf(fichero ," </font> </pre>
<br> \n");

       fprintf(fichero ,"<H4> Estadística </H4>
\n");
    fprintf(fichero ," Es la rama de las matemáticas cuyo objeto es el estudio de una
determinada población, mediante el        <br> \n");
    fprintf(fichero ," análisis de los datos obtenidos de ella. Podemos dividirla en dos
grandes apartados, primeramente la     <br> \n");
    fprintf(fichero ," estadística descriptiva que comprende todas las técnicas de obtención,
clasificación y presentación      <br> \n");
    fprintf(fichero ," de los datos de la población sujeta a estudio y en segundo lugar la
estadística inferencial que abarca   <br> \n");
    fprintf(fichero ," la generación de modelos, patrones de comportamiento y deducciones  a
partir de los datos obtenidos.     <br> \n");
```

```
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," Fases del estudio estadístico :
<br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," - Definir el objeto del estudio
<br> \n");
    fprintf(fichero ," - Definir las variables a estudiar
<br> \n");
    fprintf(fichero ," - Recoger los datos de la muestra
<br> \n");
    fprintf(fichero ," - Describir detalladamente la información obtenida
<br> \n");
    fprintf(fichero ," - Realizar las inferencias pertinentes
<br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," Población: Es el conjunto de elementos sobre el cual se desea realizar
el estudio estadístico.           <br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," Muestra: Es un subconjunto de la Población, obtenido mediante las
técnicas de muestreo adecuadas         <br> \n");
    fprintf(fichero ," que permitan que dicha muestra sea representativa del conjunto de la
población estudiada.               <br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," Dato: Es el valor concreto de los que puede tener una variable.
<br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," Variable: Es una característica de los elementos de una población, que
según su nivel de medida          <br> \n");
    fprintf(fichero ," pueden clasificarse en:
<br> \n");
    fprintf(fichero ," <pre> <font face=\"courier new\" size=\"2\">
<br> \n");
    fprintf(fichero ,"     - Cualitativas  >>
\n" );
    fprintf(fichero ,"       Las que expresan características no numéricas (también
denominadas atributos).                        \n");
    fprintf(fichero ,"           - Dicotómicas -> Solamente pueden tomar dos valores (si/no,
h/m).                               \n");
    fprintf(fichero ,"           - Politómicas -> Pueden tomar más de dos valores (nombres,
colores, lugares).                   \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ,"     - Cuantitativas >>
\n");
    fprintf(fichero ,"       Las que contienen valores numéricos.
\n");
    fprintf(fichero ,"           - Discretas -> Pueden tomar determinados valores, dentro del
intervalo.                          \n");
    fprintf(fichero ,"           - Continuas -> Pueden tomar infinitos valores, dentro del
intervalo.                           \n");
    fprintf(fichero ," </font> </pre>
<br> \n");
    fprintf(fichero ," Frecuencia Absoluta: Cantidad de veces que aparece un determinado valor
dentro de la serie de datos.     <br> \n");
    fprintf(fichero ," Frecuencia Relativa: Cociente entre la Frecuencia Absoluta y la
cantidad de elementos de la muestra.     <br> \n");
    fprintf(fichero ," Frecuencia Absoluta Acumulada: Sumatorio de todas las Frecuencias
Absolutas de los valores <= al valor tomado como referencia. <br> \n");
    fprintf(fichero ," Frecuencia Relativa Acumulada: Cociente entre la Frecuencia Absoluta
Acumulada y la cantidad de elementos de la muestra. <br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," Moda: Es el valor que más se repite dentro de una serie de datos; el de
mayor frecuencia.               <br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ," Tabla de frecuencias
<br> \n");
```

```
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ,"  Supongamos que en el muestreo hemos obtenido la siguiente serie
ordenada:                              <br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ,"             A A A B B C C C C C C C D D E E E F F G G G G H H H I I I
I                                 <br> \n");
    fprintf(fichero ," <pre> <font face=\"courier new\" size=\"2\">
<br> \n");
    fprintf(fichero ,"                                          Tabla de Frecuencias
<br> \n");
    fprintf(fichero ,"  Valor   Frecuencia    Frecuencia       Frecuencia    Frecuencia
Porcentaje                          <br> \n");
    fprintf(fichero ,"    x     Absoluta      Abs. Acumulada   Relativa      Rel. Acumulada
(FR · 100)                          <br> \n");
    fprintf(fichero ,"
<br> \n");
    fprintf(fichero ,"    A        3               3             0,1             0,1
10,00 %                           <br> \n");
    fprintf(fichero ,"    B        2               5             0,0667          0,1667
6,67 %                           <br> \n");
    fprintf(fichero ,"    C        7              12             0,2333          0,4
23,33 % (Moda)                    <br> \n");
    fprintf(fichero ,"    D        2              14             0,0667          0,4667
6,67 %                           <br> \n");
    fprintf(fichero ,"    E        3              17             0,1             0,5667
10,00 %                           <br> \n");
    fprintf(fichero ,"    F        2              19             0,0667          0,6334
6,67 %                           <br> \n");
    fprintf(fichero ,"    G        4              23             0,1333          0,7667
13,33 %                           <br> \n");
    fprintf(fichero ,"    H        3              26             0,1             0,8667
10,00 %                           <br> \n");
    fprintf(fichero ,"    I        4              30             0,1333          1
13,33 %                           <br> \n");
    fprintf(fichero ,"Totales     30                                            1
100,00 %                           <br> \n");
    fprintf(fichero ," </font> </pre>
<br> \n");

    fprintf(fichero ,"</body> \n");
    fclose(fichero);
    RetCode = system("chmod 777 ./zhelp002.html");
    RetCode = system("/usr/bin/firefox ./zhelp002.html");

   return 0;
}

/*****************************************************************************/
/***                            End of Program                             ***/
/*****************************************************************************/
```

6. Algoritmos

6.1 Representación del algoritmos

En nuestra propia vida cotidiana, a veces de forma inconsiente, desarrollamos algoritmos para determinar en que orden y forma realizaremos la actividad diaria que debemos acometer. En el mundo de la programación de ordenadores es imprescindible desarrollar algoritmos para posteriormente diseñar programas informáticos eficaces.

Definición del algoritmo

El **algoritmo** es un conjunto de instrucciones precisas y ordenadas que permiten la resolución de un problema dado, en un tiempo finito.

Representación de un algoritmo

Existen diversas formas de representar un algoritmo, de las que seguidamente indicamos algunas de las más utilizadas, como son el lenguaje natural, los diagramas de flujo y el pseudocódigo. Si bien es cierto que existen otros tipos de representación de algoritmos que no son objeto de este estudio.

Lenguaje natural

Como su nombre indica, esta opción consiste en narrar los pasos a seguir para conseguir la resolución del problema dado, incluyendo en ocasiones ejemplos aclaratorios.

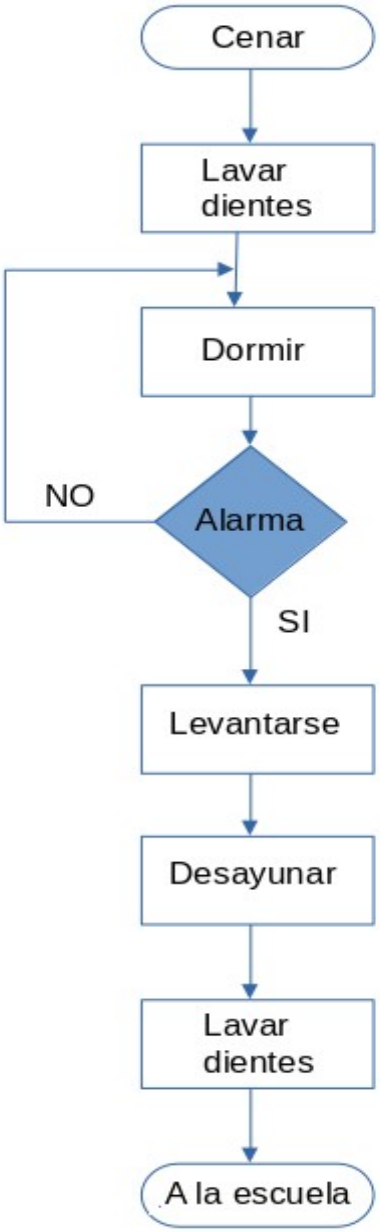
Veamos un ejemplo simple de como planificar la rutina de un hijo, en edad escolar, desde la hora de cenar por la noche hasta que va a la escuela al día siguiente:

- Cenar
- Lavarse los dientes
- Ir a dormir hasta que suene el despertador
- Levantarse
- Desayunar
- Lavarse los dientes
- Ir a la escuela

Diagramas de flujo

Una de las forma de representación de un algoritmo es a través de diagramas de flujo, los cuales, mediante una determinada simbología (ver anexo I), describen el proceso a realizar de forma inequívoca. Además de la simbología, pueden incluir breves textos explicativos.

Continuando con el ejemplo anterior, veremos seguidamente el diagrama de flujo correspondiente:



Pseudocódigo

El pseudocódigo es una forma narrativa, respetando determinadas convenciones, que permite especificar un algoritmo de forma parecida a los lenguajes de programación. También en este caso, veremos seguidamente el pseudocódigo del ejemplo anterior.

Inicio
    Cenar
    Lavarse los dientes
    mientras que no suene el despertador
        dormir
    fin mientras
    Levantarse
    Desayunar
    Lavarse los dientes
    Ir a la escuela
Fin

En nivel de detalle del pseudocódigo dependerá de los requerimiento técnicos, pudiendo llegar a un nivel de detalle tan extenso como el mismo código de implementación, si fuera preciso.

Obsérvese que el punto "*Levantarse*" del pseudocódigo anterior podría ser sustituido por el siguiente desglose:

    Despertar
    Parar la alarma
    Poner los pies en el suelo
    Bostezar
    Ir al servicio
    Vestirse
    ...

Precedencia de los operadores aritméticos

- Paréntesis **( )**
- Exponente **^**
- Multiplicación ***** y División **/**
- Módulo **%, Mod**
- Suma **+** y Resta **-**

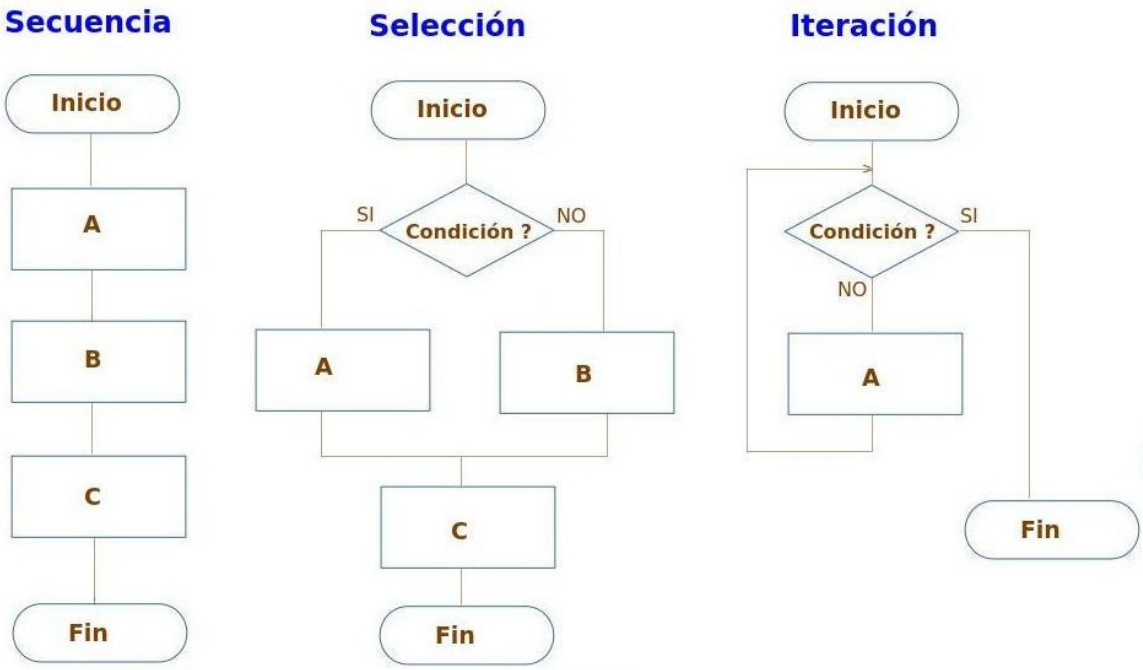
Nota: El orden de cálculo de operadores con igual prioridad es de izquierda a derecha. Si hay paréntesis anidados el orden es de dentro hacia fuera.

Programación estructurada

La programación estructurada, es un tipo de programación cuyo objetivo principal es el diseño de programas alta calidad y claridad, basada principalmente en la utilización de tres estructuras de control básicas (secuencia, selección e iteración) y de subrutinas, estas últimas en aras de aplicar la técnica de programación descendente.

En este tipo de programación también se procura delimitar con precisión los rangos de validez de las variables y estructuras de datos, evitando de este modo alteraciones no deseadas, sobre las mismas.

Estructuras de control



6.2 ejemplos de algoritmos

Lectura un fichero secuencial

Una tarea muy común en programación consiste en leer un fichero secuencial para realizar un determinado tratamiento de su información.

Diagrama de flujo

Implementación en C

Leemos un fichero secuencial, de gran volumen, y vamos informando del avance de la lectura cada N registros, cantidad definida como DIVISOR.

```
/*** Constantes  ***/
#define DIVISOR 1000000
#define WORKFILE "./workfile.txt"
#define LONGIMAX 99

/*** Variables  ***/
char Registro[LONGIMAX];
.
.
.
// Leer fichero
   int k;

   FILE *FileINP;
   FileINP = fopen(WORKFILE, "r");

   while(fgets(Registro, LONGIMAX, (FILE*) FileINP))
       {
       k = k+1;
       if ((k%DIVISOR)==0)
          {
          printf("\e[92mRegistros leidos: \e[m %d \n", k);
          }
       }

   fclose(FileINP);

   return 0;
}
```

Copiar ficheros de texto

Una tarea muy común es realizar copias de ficheros de texto, para lo cual todos los sistemas operativos disponen de sus correspondientes utilidades. En nuestro ejemplo desarrollamos una utilidad para copiar un fichero añadiéndole un prefijo y un sufijo a cada registro.

Pseudocódigo

Inicio
    abrir fichero de entrada
    abrir fichero de salida
    mientras que no es fin de fichero de entrada
        leer registro de entrada
        añadir prefijo y sufijo
        grabar registro de salida
    fin mientras
    cerrar fichero de entrada
    cerrar fichero de salida
Fin

Implementación en C

```
.
.
/*** Definición de constantes ***/
#define MAX_LONGI 9999
#define PREFIJO "xxxxxxxxxx"
#define SUFIJO "yyyyyyyyyy”

/*** Definición de variables ***/
char RegINP[MAX_LONGI];
char RegOUT[MAX_LONGI];

/***  Main Program ***/
int main(int CantParams, char *Param[])
{
.
.
.
// Copiar Fichero
   FILE *FENTRADA;
   FENTRADA = fopen(Param[1], "r");

   FILE *FSALIDA;
   FSALIDA = fopen(Param[2], "w");

   while (!feof(FENTRADA))
     {
       fgets(RegINP, MAX_LONGI, (FILE*) FENTRADA);

       if (strlen(RegINP)>1)
          {
          strcpy(RegOUT, PREFIJO);
```

```
          RegINP[strlen(RegINP)-2] = '\0';
          strcat(RegOUT, RegINP);
          strcat(RegOUT, SUFIJO);
          fprintf (FSALIDA, "%s\n", RegOUT);
          }
     }

   fclose(FENTRADA);
   fclose(FSALIDA);

// Final
   printf ("\n\n\e[96m *** End of Program %s ***\e[m\n\n", Param[0]);
   return 0;
}
```

Nota: Este programa esta diseñado para tratar registros cuya longitud sea inferior a :

(9999 – (longitud prefijo + longitud sufijo + 2))

Ordenar pequeños ficheros

En muchas ocasiones nos encontramos con la necesidad de ordenar la información, para posteriormente tratarla de una manera específica, como sucede con pequeños ficheros habitualmente utilizados en los programas de gestión (Países, regiones, provincias, idiomas...etc)

Pseudocódigo

Inicio
leer fichero de entrada y cargarlo en memoria
abrir fichero
mientras que no sea fin de fichero
leer registro y cargarlo en vector
fin mientras
cerrar fichero
ordenar el vector cargado en memoria
(Detalle del algoritmo de ordenación, si es preciso)
grabar el fichero de salida
abrir fichero
mientras que no sea fin de vector
grabar fila del vector en registro de salida
fin mientras
cerrar fichero
Fin

Implementación en VB6

La siguiente rutina permite ordenar un fichero con registros, tanto de longitud fija como variable, pero en cualquier caso que finalicen con CR.

```
Dim Vector(10000, 1) As String
Dim j, k, Tope_Vector As Double
Dim WorkField As String


    ' Leer fichero y cargarlo en memoria
    j = 0
    Open "entrada.txt" For Input As #1
    While Not EOF(1)
        Input #1, Vector(j, 1)
        j = j + 1
    Wend
    Close #1
    Tope_Vector = j
    ProgressBar1.Max = Tope_Vector

    ' Ordenar el vector en memoria
    For j = 0 To Tope_Vector
    ProgressBar1.Value = j
        For k = 1 To Tope_Vector
        If Vector(j, 1) < Vector(k, 1) Then
            WorkField = Vector(k, 1)
```

```
            Vector(k, 1) = Vector(j, 1)
            Vector(j, 1) = WorkField
        End If
        Next k
    Next j

    ' Grabar fichero de salida
    Open "salida.txt" For Output As #1
    For j = 0 To Tope_Vector
        Print #1, Vector(j, 1)
    Next j
    Close #1
```

Bloqueo de elementos de una BD y/o programas

Al objeto de evitar actualizaciones solapadas sobre un mismo elemento de la BD (registro, tabla, fichero temporal) o incluso la ejecución paralela de programas o transacciones, es preciso utilizar algún método que nos permita garantizar la integridad de la información.

Una de las posibles formas de conseguir este objetivo es trabajar con una tabla de bloqueos en la que se registren previamente los elementos que deseamos bloquear, de forma inmediata, y se eliminen una vez actualizados, según la secuencia que seguidamente se describe.

Lenguaje natural

1. Antes de realizar la lectura del objeto a tratar, leemos la tabla de bloqueos para comprobar si el objeto ya estuviera bloqueado.

2. Si está bloqueado, emitimos un mensaje informativo y opcionalmente lo mostramos en modo de solo lectura, si son registros o tablas.

3. Si no está bloqueado, grabamos el identificador del registros en la tabla de bloqueos, para bloquearlo.

4. Cuando finalizamos la actualización, borramos el registro correspondiente de la tabla de bloqueos, dejándolo libre para posteriores actualizaciones.

Normalmente utilizaremos los identificadores del nombre del elemento (Nombre_Objeto) y el valor de la clave de búsqueda (Valor_Objeto), identificando de este modo unívocamente que deseamos actualizar.

Descripción de la tabla de bloqueos

Nombre_Objeto	CHAR(99)	primary key
Valor_Objeto	CHAR(99)	primary key
Usuario	CHAR(50)	
Workstation	CHAR(50)	
Fecha	DATE	
Hora	TIME	

Tipo de bloqueo

El único tipo de bloqueo que ofrece una garantía absoluta de que no podrán ser realizadas actualizaciones solapadas es “adLockPessimistic”. No obstante con la utilización de un método como el descrito anteriormente es posible utilizar “adlockoptimistic”, puesto que es mediante código como conseguimos la garantía de la integridad de la información.

Función de Consulta, Bloqueo y Desbloqueo

Una única función que permita realizar las acciones de bloqueo/desbloqueo de elementos de la BD es suficiente para nuestro propósito. Dicha función debería recibir como parámetros de entrada el Nombre_Objeto, el Valor_Objeto y el tipo de acción (**B**loquear, **C**onsultar, **D**esbloquear) y mediante un código de retorno puede indicar si la función ha finalizado correctamente.

Ejemplos

Para bloquear un registro:
Nombre-Objeto: Nombre de la tabla que contiene el registro
Valor_Objeto: Valor de la clave del registro

Para bloquear una tabla:
Nombre-Objeto: “TBL”
Valor_Objeto: Nombre de la tabla

Para bloquear un programa:
Nombre-Objeto: “PGM”
Valor_Objeto: Nombre del programa

Para bloquear una transacción:
Nombre-Objeto: “TRN”
Valor_Objeto: Nombre de la transacción

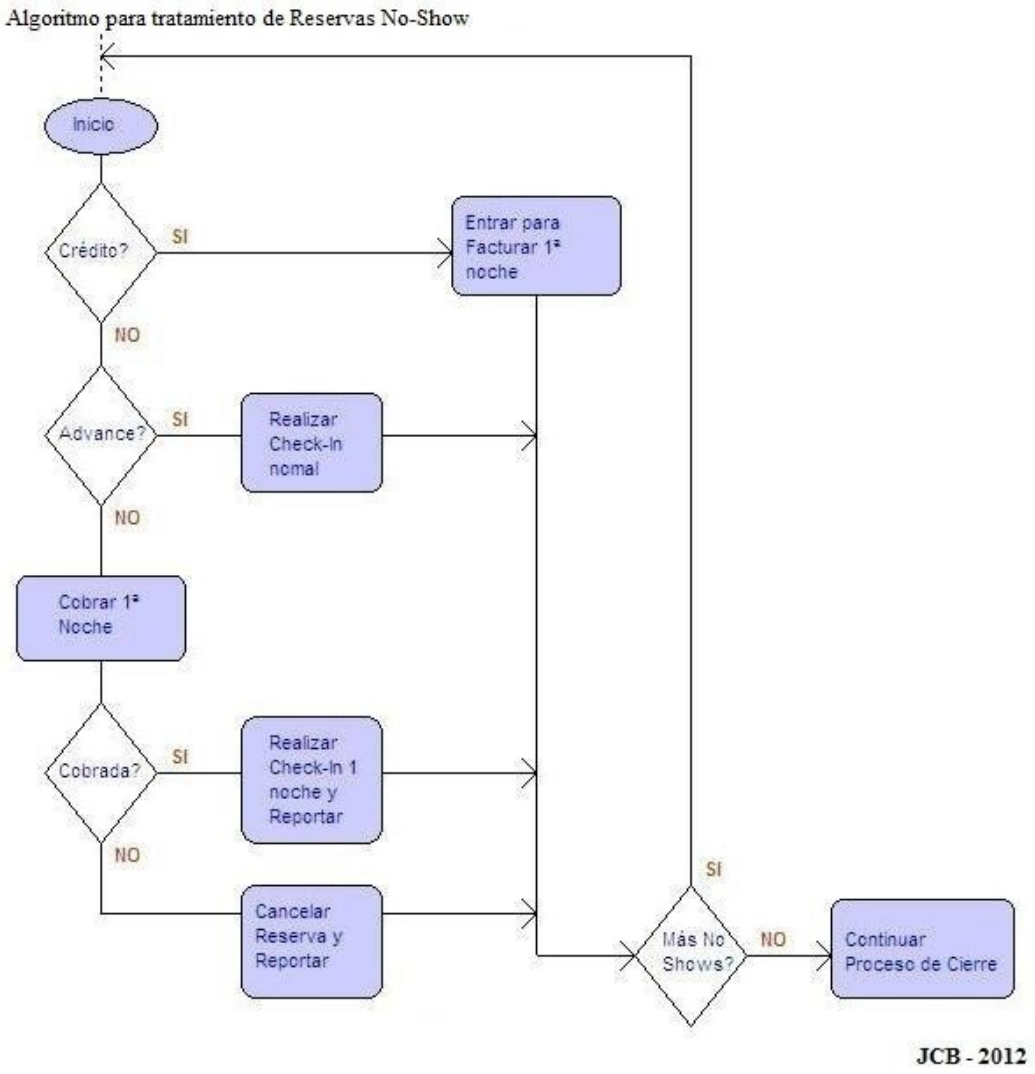
Observaciones

• La implementación de este método de bloqueo, no requiere obligatoriamente ser realizado en conjunto, en un corto espacio de tiempo; es posible ir implementándolo progresivamente en la medida que se realizan mantenimientos sobre programas de actualización.

Tratamiento de reservas No-Show

En el ámbito de la hostelería, aquellas reservas en la que no se presenta el cliente sin previo aviso reciben el tratamiento de no-show, que permite según la normativa cobrar la primera noche y guardar dicha reserva, a la espera del cliente, hasta el día siguiente que será cancelada.

Diagrama de flujo



Nota: una reserva advance es aquella que se paga por adelantado y el importe de la misma no es reembolsable, aunque no se presente el cliente.

Calcular el factorial de un número N

El factorial de un número N es el producto de todos los números enteros positivos desde el 1 hasta el número dado inclusive.

Pseudocódigo

Inicio
leer número N
asignar multiplicador = 1
asignar factorial = 1
si ((N = 0) OR (N = 1))
el factorial es 1
sino
mientras multiplicador <= N
factorial <-- factorial * multiplicador
incrementar 1 a multiplicador
fin mientras
fin si
mostrar factorial
Fin

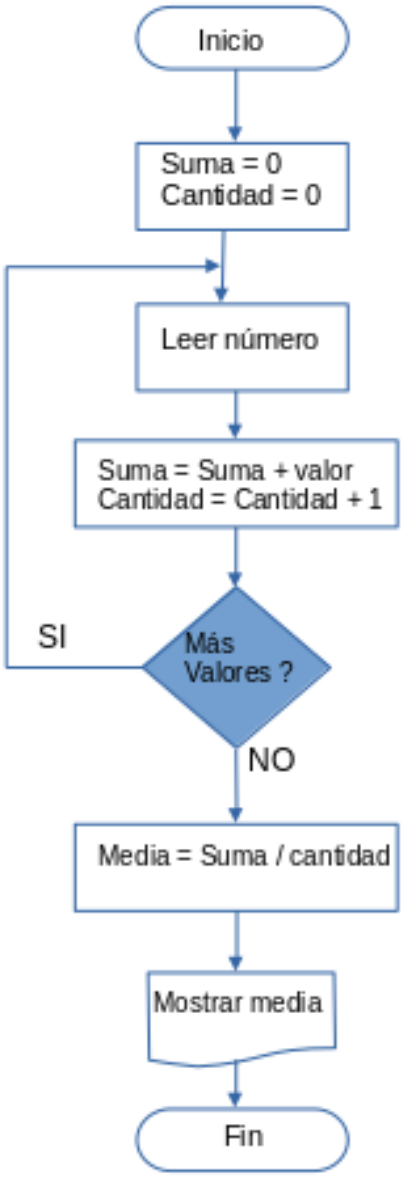
Implementación en R

```
numero <- scan("/home/server/TIC/PgmsR/datos.txt")
numero <- as.integer(numero)

factorial <- 1

if ((numero==0)|(numero==1))
   {
    print(1)
   } else {
   for( i in 1:numero)
      factorial <- factorial * i
      print (factorial)
   }
```

Calcular media aritmética de una serie

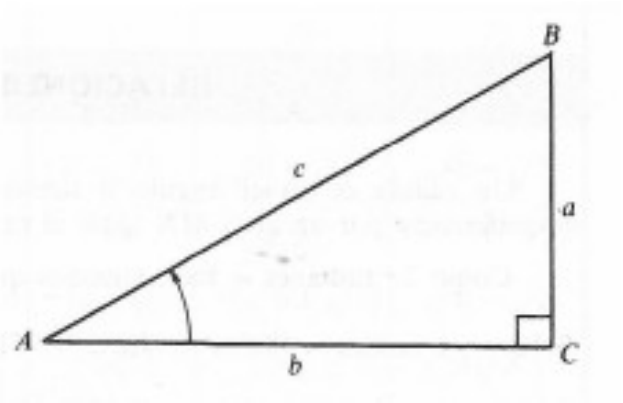
La media aritmética de una serie numérica es el valor obtenido al sumar todos los números, dividido por la cantidad de termino de la serie.

Diagrama de Flujo



Cálculo de razones trigonométricas

Lenguaje natural



Sen A = a/c
Cos A = b/c
Tg A = a/b
Cotg A = b/a
Sec A = c/b
Cosec A = c/a

Pi = 3,1415926535 1 Radián = 180º / Pi

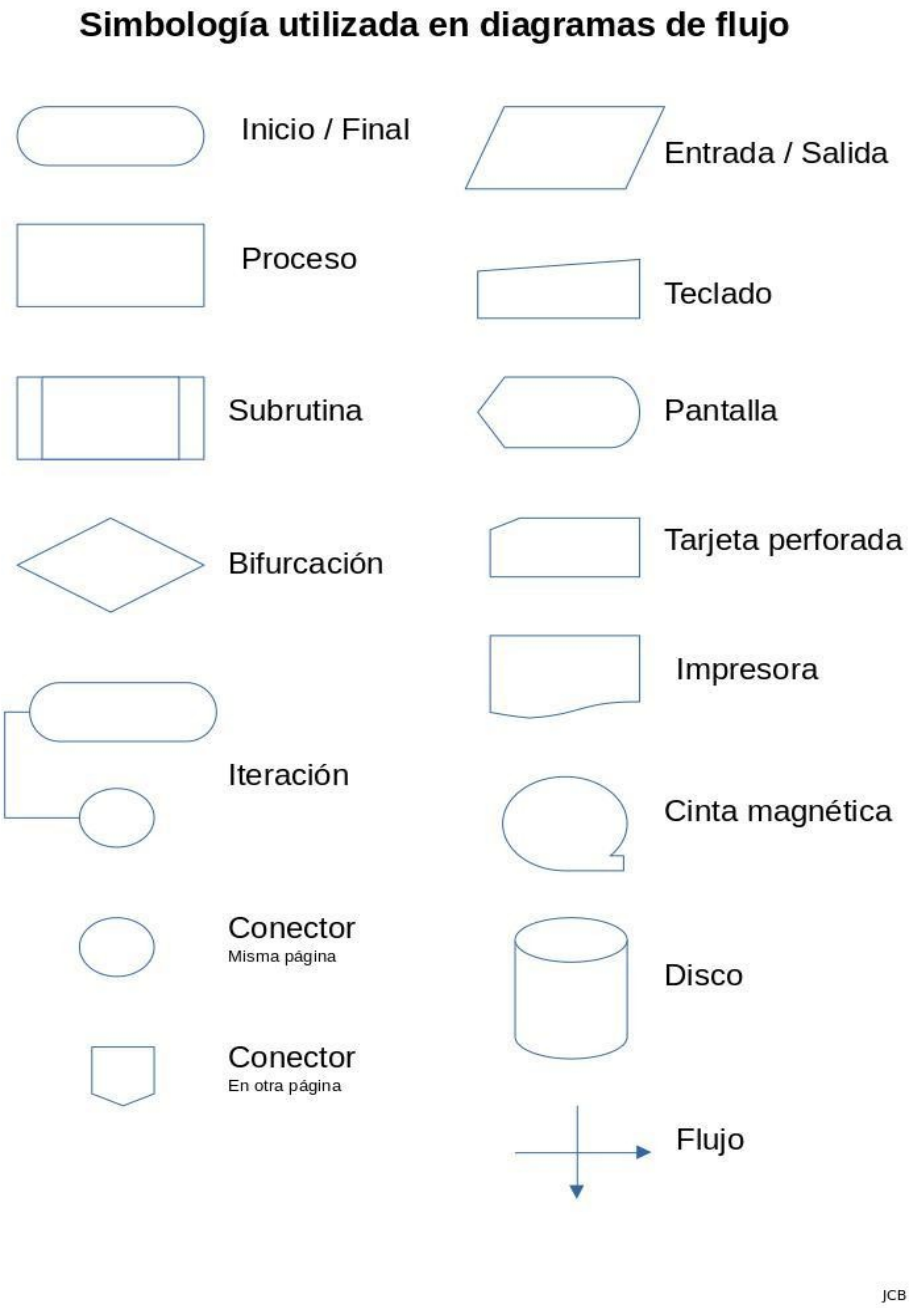
Ley de los Senos

(a / sen A) = (b / Sen B) = (c / Sen C)

Adición

Sen(A±B) = Sen A * Cos B ± Cos A * Sen B
Cos(A±B) = Cos A * Cos B ± Sen A * Sen B

Anexo I. Simbología de diagramas de flujo



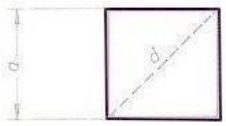
Anexo II. Superficies y volúmenes

SUPERFICIES

Cuadrado

$$A = a^2$$

$$d = a\sqrt{2}$$

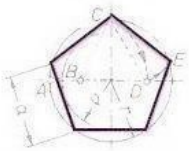


Pentágono Regular

$$A = \frac{5}{8} r^2 \sqrt{10 + 2\sqrt{5}}$$

$$a = \frac{1}{2} r \sqrt{10 - 2\sqrt{5}}$$

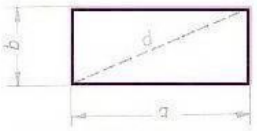
$$\rho = \frac{1}{4} r \sqrt{6 + 2\sqrt{5}}$$



Rectángulo

$$A = a\,b$$

$$d = \sqrt{a^2 + b^2}$$

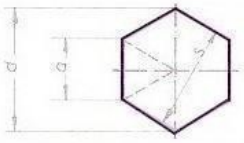


Hexágono Regular

$$A = \frac{3}{2} a^2 \sqrt{3}$$

$$d = 2a = \frac{2}{\sqrt{3}} s$$

$$s = \frac{\sqrt{3}}{2} d$$

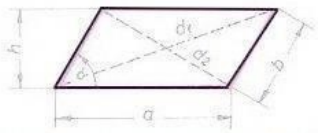


Paralelogramo

$$A = a\,h = a\,b \operatorname{sen} \alpha$$

$$d_1 = \sqrt{(a + h \cot \alpha)^2 + h^2}$$

$$d_2 = \sqrt{(a - h \cot \alpha)^2 + h^2}$$



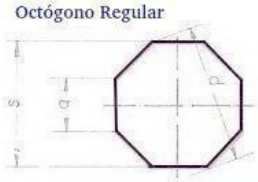
Octógono Regular

$$A = 2\,a\,s \cong 0.83\,s^2 = 2\,s\sqrt{d^2 - s^2}$$

$$a = s \tan 22.5°$$

$$s = d \cos 22.5°$$

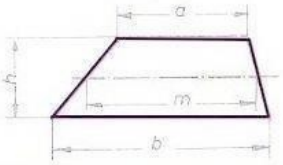
$$d = \frac{s}{\cos 22.5°}$$



Trapecio

$$A = \frac{a + b}{2} h = m\,h$$

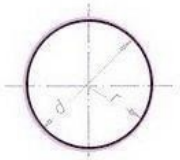
$$m = \frac{a + b}{2}$$



Círculo

$$A = \frac{\pi}{4} d^2 = \pi r^2$$

$$P = 2\pi r = \pi d$$

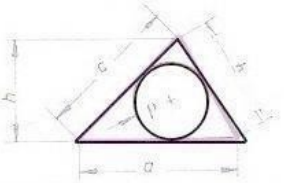


Triángulo

$$A = \frac{a\,h}{2} = \rho\,c$$

$$= \sqrt{s(s-a)(s-b)(s-c)}$$

$$s = \frac{a + b + c}{2}$$

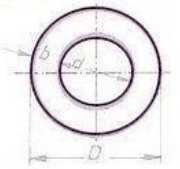


Corona Circular

$$A = \frac{\pi}{4}(D^2 - d^2)$$

$$= \pi(d + b)\,b$$

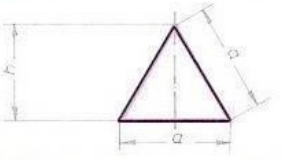
$$b = \frac{D - d}{2}$$



Triángulo Equilátero

$$A = \frac{a^2}{4}\sqrt{3}$$

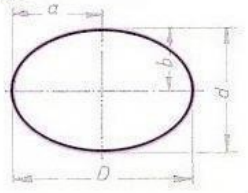
$$h = \frac{a}{2}\sqrt{3}$$



Elipse

$$A = \frac{\pi}{4} D\,d = \pi\,a\,b$$

$$P \cong \pi \frac{D + d}{2} = \frac{a - b}{a + b}$$



JCB - 1982

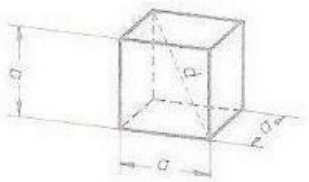
VOLÚMENES

Cubo

$V = a^3$

$A = 6\,a^2$

$d = \sqrt{3}\,a$

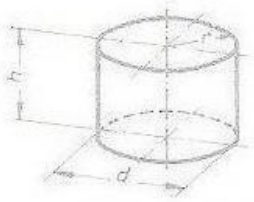


Cilindro Circular

$V = \frac{\pi}{4} d^2 h$

$A_l = 2\pi r h$

$A_t = 2\pi r(r + h)$

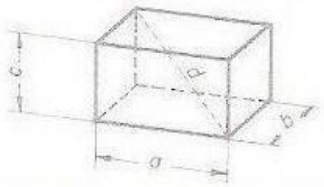


Prisma Rectangular

$V = a\ b\ c$

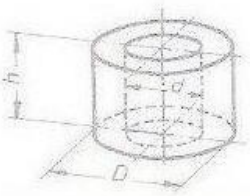
$A = 2(a\,b + a\,c + b\,c)$

$d = \sqrt{a^2 + b^2 + c^2}$



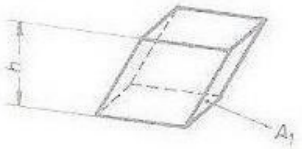
Cilindro Hueco

$V = \frac{\pi}{4} h\,(D^2 - d^2)$



Prisma Oblicuo

$V = A_l\ h$



Cono Circular

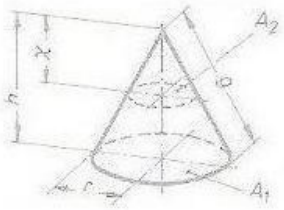
$V = \frac{\pi}{3} r^2 h$

$A_l = \pi r g$

$A_t = \pi r(r + g)$

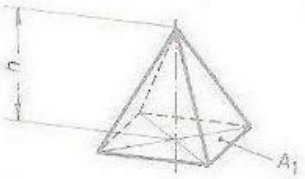
$g = \sqrt{h^2 + r^2}$

$A_2 : A_1 = x^2 : h^2$



Pirámide Rectangular

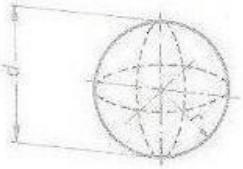
$V = \frac{A_l\ h}{3}$



Esfera

$V = \frac{4}{3}\pi r^3 = \frac{1}{6}\pi d^3$

$A = 4\pi r^2 = \pi d^2$



Leyenda

V = Volumen A = Area h = Altura r = Radio d = Diámetro / Diagonal a,b,c = Lados

JCB - 1982

Anexo III Glosario de términos y abreviaturas

ALFANUMÉRICO – Referido a cualquier combinación de letras y números.

ASCII - American Standard Code of Information Interchange.

BASE DE DATOS – Conjunto de información estructurada y organizada para su rápido acceso, habitualmente almacenada en sistemas informáticos.

BATCH – Referido normalmente a los procesos por lotes que se ejecutan en un ordenador y que no interaccionan con el usuario, o a lo sumo con el operador del sistema.

BIT – (Digito binario) es la unidad mínima de información utilizada en los ordenadores que puede representar dos estados: 0 ó 1

BYTE – Es un conjunto de 8 bits. Además es la cantidad de memoria necesaria para almacenar un concepto inteligible como una letra, un número o un símbolo especial.

CAMPO – (como parte de un registro) Es un conjunto de bytes consecutivos con significado propio. El tamaño mínimo de un campo es de un byte. Existe además un tipo especial de campo, denominado Campo Clave que es aquel que identifica unívocamente cada registro de un fichero indexado.

DATAGRID – Es un control, utilizada de forma común en los lenguajes de programación, que permite visualizar y modificar fácilmente (dentro de una matriz gráfica) el contenido de tablas de base de datos, dado que contiene las funciones de presentación y tratamiento habitualmente necesarias para ello.

DIRECTORIO - Contenedor de programas, archivos y subdirectorios, dentro de un sistema de almacenamiento.

EOF – (End Of File) Fin de fichero.

FICHERO - Conjunto de registros sistemáticamente almacenados en memoria auxiliar, para su posterior consulta y/o tratamiento.

FIFO - (First in first out) En lo referente a colas de trabajo, son aquellas en las que la primera tarea entrada es la primera en ser procesada y así sucesivamente.

FTP – Es un protocolo de transferencia de ficheros entre ordenadores.

GUI – Graphical User Interface

LIFO - (Last in first out) En lo referente a colas de trabajo, son aquellas en las que la última tarea registrada es la primera en ser procesada.

LRL - (Logical record lenght) Longitud de registro lógico.

MAINFRAME – (Unidad central) Término con el que se designa a grandes computadores utilizados para el procesamiento masivo de información de grandes bases de datos, procesamiento realizado habitualmente en batch.

PROGRAMA – Un programa informático es un conjunto de instrucciones que le indican a un ordenador como realizar una tarea concreta. Normalmente un programa se escribe en un lenguaje inteligible para el programador y posteriormente es convertido, mediante el proceso de compilación, al lenguaje que comprende el procesador.

ORDENADOR - Equipo electrónico capaz de manejar grandes volúmenes de información, realizando con dicha información operaciones aritméticas y lógicas a gran velocidad, habitualmente utilizadas para transformar los datos de entrada (facilitados por el usuario) en la información de salida (esperada por el mismo).

REGISTRO - Un registro es un tipo de información estructurada compuesta por una serie de elementos denominados campos (véase CAMPO).

REGISTRO LÓGICO - Es una estructura de datos homogénea que hacen referencia a una misma entidad. Podemos considerar al registro como una unidad de tratamiento incluida dentro de un fichero.

REGISTRO FÍSICO - (Bloque) Un registro físico es el conjunto de registros lógicos que el sistema operativo puede transferir en una única operación de entrada y salida. Estos registros son transferidos entre la memoria principal del ordenador y los periféricos o dispositivos de almacenamiento y el tamaño del bloque depende del ordenador.

SGBD – Sistema de gestión de base de datos. Consiste en un conjunto de programas y utilidades que permiten la gestión y mantenimiento de una o más bases de datos.

SGD – Sistema de gestión documental.

SOFTWARE – Es el conjunto de programas y datos que se procesan en un sistema informático. Cabría distinguir entre el software del sistema, encargado de conseguir el funcionamiento del propio sistema informático, y el software de aplicaciones destinado al tratamiento específico de la información de una organización.

SQL – Structured Query Languaje

TPV – El Terminal Punto de Venta es un un ordenador con la funcionalidad de cajero automático y conectado al ordenador central.

TUI – Text-based User Interface.

UML – Unified Modeling Language es uno de los lenguajes más utilizados para la generación de modelos de software.

Bibliografía

Cooper, Mendel. *Advanced Bash-Scripting Guide.* (2014). Public Domain.

Donald E. Knuth (1980). *Algoritmos fundamentales I*. Barcelona: Editorial reverté.

Real Academia Española de la lengua. https://dle.rae.es

Tacket, Jack & Gunter, David. *Linux* (1998). Madrid. Prentice Hall.