Práctica 1- El editor vi de Unix y el compilador cc

1. Objetivos.

- Utilizar terminales NC para iniciar una sesión remota con un servidor UNIX
- Utilizar comandos básicos de UNIX para copiar, borrar y crear ficheros, así como para moverse dentro de una estructura de directorios.
- Aprender el funcionamiento básico del editor de textos vi.
- Compilar un programa escrito en lenguaje C y ejecutarlo.

2. Inicio de sesión utilizando el servidor *shannon*.

Enciende el terminal, observa cómo se descarga el sistema operativo desde el servidor shannon. Tras cargarse el sistema operativo, aparece una pantalla de inicio, en la que debes introducir tu nombre de usuario (*login*) y tu contraseña (*password*), que permite verificar la autenticidad del usuario.

Tras la validación del usuario, aparece el escritorio del sistema operativo. Navega por el menú que aparece en la esquina superior izquierda de la pantalla, y encuentra entre sus entradas algún **navegador** de Internet, la **consola** del sistema y la aplicación de **terminal** (*Unix shell*). Aunque tanto la consola como el terminal permiten el inicio de aplicaciones y la ejecución de órdenes propias del sistema operativo (*comandos*), es recomendable utilizar el terminal, pues en la consola pueden aparecer mensajes inesperados, generados por el sistema operativo, que pueden molestar mientras se está realizando un trabajo.

Un principio básico de funcionamiento en un sistema multiusuario como es UNIX, recomienda modificar nuestra palabra de paso (*password*) la primera vez que entremos en el sistema. Esto se consigue ejecutando la orden ***passwd*** en un terminal, lo que equivale a teclear la orden a continuación del *prompt*[1] del sistema. Esta orden pide al usuario la contraseña antigua y, a continuación, solicita la nueva palabra de paso dos veces.

Por otro lado, siempre que se deje de trabajar es necesario cerrar la sesión, es decir, hay que seleccionar la opción **logout** del menú. El peligro de dejar un equipo conectado sin realizar el *logout* correspondiente, radica en que otro usuario podría utilizarlo de forma

[1] El prompt del sistema operativo es un carácter que aparece en la consola y que indica al usuario que el sistema se encuentra a la espera de una orden. Para usuarios normales, el símbolo es el carácter **$**, mientras que para el superusuario (root), el símbolo es **#**.

fraudulenta (usurpación de personalidad para realizar actividades no permitidas - ataque a otros sistemas-, copia de trabajos, etc.).

Antes de continuar, asegúrate de que has entrado en el sistema, arrancado un terminal (*shell*) y modificado tu contraseña. Aunque existen más normas para elegir una buena contraseña, es recomendable que una palabra de paso no aparezca en el diccionario y que utilice una combinación de caracteres del alfabeto y caracteres no alfabéticos (como los números, por ejemplo).

3. Comandos básicos de UNIX.

Ayuda de UNIX

La orden **man** proporciona ayuda acerca de las instrucciones UNIX.

Sintaxis: `man` *`orden`*

Ejemplo: `man` *`man`*

Estructura de subdirectorios

La información que se almacena en un disco duro se organiza en una estructura de directorios y subdirectorios.

La ruta completa del subdirectorio actual se puede obtener con el comando **pwd**:

Sintaxis: `pwd`

El directorio por defecto de cada usuario se obtiene visualizando el valor de la variable HOME. Para comprobar cuál es la ruta (*path*) del subdirectorio propio, se teclea la siguiente orden:

```
echo $HOME
```

El comando **mkdir** permite crear un subdirectorio dentro del actual.

Sintaxis: `mkdir` *`nombre_subdirectorio`*

Crea un subdirectorio de nombre *IP* dentro de tu directorio HOME:

```
mkdir IP
```

Para navegar por la estructura de subdirectorios se emplea la instrucción **cd**.

Sintaxis: `cd` *`nombre_subdirectorio`*

Para poder salir de un subdirectorio se utiliza:

```
cd ..
```

Entra dentro del directorio IP:

```
cd IP
```

Comprueba, con pwd, que estás dentro del directorio IP:

```
pwd
```

Sal del directorio IP y vuelve al inmediatamente anterior:

```
cd ..
```

Comprueba, con pwd, que has vuelto a tu directorio HOME.

Manejo de ficheros

ls vuelca en pantalla los nombres de los ficheros que hay en el subdirectorio actual.

```
ls
```

Observa que aparece también el subdirectorio IP recién creado. Esto se debe a que UNIX considera como archivos también a los subdirectorios.

rm permite borrar un fichero:

Sintaxis: `rm` *`nombre_fichero`*

Para borrar un subdirectorio completo se emplea la orden **rmdir**:

Sintaxis: `rmdir` *`nombre_subdirectorio`*

cp se utiliza para hacer una copia de un archivo.

Sintaxis: `cp` *`fichero_original fichero_destino`*

more muestra por la pantalla el contenido de un fichero de texto, esperando por la pulsación de una tecla por parte del usuario para indicarle que debe continuar mostrando el contenido del documento. Si se pulsa la barra espaciadora, se muestra la siguiente pantalla completa; si la tecla pulsada es el retorno de carro [intro] se muestra la siguiente línea; si se pulsa la letra **v**, se edita el fichero con el editor **vi**.

4. Utilización del editor vi.

El editor de texto **vi** (***v****isual* ***e****ditor*) es el editor típico de UNIX y, aunque no es fácil de utilizar, es el único que se encuentra con seguridad en cualquier sistema UNIX.

Sintaxis: `vi` *`nombre_de_fichero`*

Modos de trabajo

Al usar **vi**, en cualquier momento se está en uno de los siguientes tres modos:
modo de comandos, modo de inserción o modo de última línea.

1. **Modo de comandos**. Es el modo en el que arranca **vi**. Permite usar ciertos comandos para editar ficheros o para cambiar a otros modos.
2. **Modo de inserción**. Se utiliza para insertar o editar texto. Para salir de este modo y volver al modo comando, se pulsa [ESC].

3. **Modo de última línea**. Es un modo especial para ciertos comandos extendidos. Se accede a este modo pulsando [:], a continuación se introduce la orden (que aparecerá en la última línea de la pantalla) y se pulsa [intro] para ejecutarlo.

Edición con vi, compilación y ejecución de un programa en lenguaje C

1. Entra en el subdirectorio IP que has creado dentro de tu directorio HOME:

```
cd $HOME/IP
```

2. Comprueba cuál es el *path* o ruta completa del subdirectorio actual:

```
pwd
```

3. Crea otro subdirectorio dentro de IP llamado PRACTICA1 y accede a él.

```
mkdir PRACTICA1
cd PRACTICA1
```

4. Sal de PRACTICA1 y bórralo:

```
cd ..
rmdir PRACTICA1
```

5. Inicia el editor de textos **vi** para editar el fichero *practica1.c*:

```
vi practica1.c
```

En cuanto se inicia vi, el editor se encuentra en modo de comandos. Para pasar a modo **inserción**, se pulsa ***i***. En este momento puede teclearse un texto. Inserta la siguiente línea de texto:

```
/* funcion principal. Divide dos enteros. */
```

6. Para finalizar el modo de inserción y volver al modo de comandos, pulsa [ESC].

Existen otras formas de insertar texto además de pulsando ***i***. Por ejemplo, ***a*** permite insertar texto comenzando después de la posición actual del cursor.

7. Desde el modo de comandos, se puede desplazar el cursor utilizando las flechas o las letras ***h*** (izquierda), ***j*** (abajo), ***k*** (arriba) y ***l*** (derecha). Sitúa el cursor justo antes de la palabra *función*. Pulsa ***i*** y teclea las palabras: `Esta es la`. A continuación, pulsa [ESC] para volver al modo comando.

8. Para insertar texto una línea por encima de la actual se usa el comando ***O*** (mayúscula). Inserta al principio del documento la siguiente línea:

```
#include <stdio.h>
```

9. Para insertar texto una línea por debajo de la actual, se usa el comando ***o*** (minúscula). Sitúa el cursor en la última línea, pulsa ***o*** y teclea el siguiente texto:

```
int main(int argc, char *argv[])
```

10. Desde el modo de comandos, pulsando ***x*** se borra el carácter situado bajo el cursor. Borra la letra *V*.

11. Inserta la letra *v* (minúscula) antes de *oid*.

12. La forma de borrar una palabra completa es teclear ***dw*** cuando el cursor está situado encima de esa palabra. Borra la palabra *main*.

13. Se puede borrar una línea entera desde el modo comando tecleando ***dd***. Elimina la línea `#include <stdio.h>`.

14. Guarda el fichero sin salir de vi. Para ello, se debe teclear ***:w*** desde el modo de comando.

15. Completa el documento hasta tener el siguiente código fuente en lenguaje C:

```
#include <stdio.h>
/* Funcion principal. Divide dos enteros. */
int main(int argc, char *argv[])
{
int num,den; /*Se declaran dos variables de tipo entero */
float sol; /*Se declara una variable de tipo real */
printf("\nNumerador? "); /*Se imprime un mensaje */
scanf("%d",&num); /*Se solicita una entrada desde la consola */
printf("Denominador? ");
scanf("%d",&den);
if (den==0){ /*Si el denominador es cero, no se puede dividir */
    printf("\n Error: Division entre cero\n");
    return -1;
}
/*Cuando el denominador no es nulo, se divide */
{
sol=(float)num/den;
printf("\n\t%d / %d = %.2f\n",num,den,sol);
return 0;
}
}
```

16. Salva el programa y sal del editor. Para ello teclea, en modo comando, la orden ***:wq***.

17. Haz una copia del documento de texto practica1.c:

```
cp practica1.c practica1.old
```

18. Genera una lista de los archivos que hay en el directorio actual:

```
ls
```

19. Borra el documento *practica1.old:*

```
rm practica1.old
```

20. Comprueba con la instrucción ***ls*** que realmente se ha borrado el archivo *practica1.old.*

21. Para poder ejecutar el programa es necesario compilarlo, es decir, convertirlo a un código que pueda entender el computador. El compilador que se va a utilizar es el ***cc*** que viene incluido en el sistema operativo UNIX.

Sintaxis: `cc [-o fichero_ejecutable] fichero_fuente`

El texto que va entre corchetes es opcional; en caso de no incluirse dará lugar a que el fichero ejecutable se llame *a.out.*

Compila el fichero practica1.c y genera un ejecutable de nombre *practica1*.

```
cc –o practica1 practica1.c
```

22. Ejecuta el programa:

```
./practica1
```

Observa cómo se ha tratado el caso de que el denominador sea nulo para que no se produzca ningún error durante la ejecución del programa.

Resumen de órdenes UNIX utilizadas en esta práctica:

Instrucción	Utilidad
cd	Entra a un subdirectorio contenido en el actual
cp *fichero1 fichero2*	Copia fichero1 con el nombre fichero2
ls	Muestra los ficheros contenidos en el subdirectorio actual
man *comando*	Muestra la ayuda de un comando
mkdir *subdir*	Crea un subdirectorio de nombre *subdir* dentro del actual
more *fichero*	Vuelca en pantalla el contenido de un fichero de texto
pwd	Muestra el path o ruta del subdirectorio actual
rm *fichero*	Borra el archivo de nombre *fichero*
rmdir *subdir*	Borra el subdirectorio de nombre *subdir*. (*subdir* ha de estar vacío)

Resumen de comandos vi empleados:

Comando	Utilidad
i	Insertar texto delante del cursor.
a	Insertar texto detrás del cursor.
h	Desplazar el cursor hacia la izquierda.
j	Desplazar el cursor hacia abajo.
k	Desplazar el cursor hacia arriba.
l	Desplazar el cursor hacia la derecha.
O	Inserta una línea encima de la actual.
o	Insertar una línea debajo de la actual.
x	Borrar el carácter situado bajo el cursor.
dw	Borrar una palabra completa.
dd	Borrar una línea completa.
:w	Guardar el archivo sin salir de vi.
:wq	Guardar el archivo y salir de vi.
:x	Guardar el archivo y salir de vi.
:q!	Salir de vi sin guardar el archivo.

Nota: Al modo de comandos se accede pulsando [ESC]