Práctica 2. Uso de bibliotecas de programación de interfaces de usuario en modo texto

Duración: 2 sesiones

1. Objetivos de la práctica

Los objetivos concretos son:

- instalar la librería *ncurses* en Linux
- crear programas sencillos basados en *ncurses*

2. Introducción

"*ncurses*" es una biblioteca de programación que provee una API que permite al programador escribir interfaces basadas en texto.

Podemos usar *ncurses* en cualquier sistema Unix que siga la norma ANSI/POSIX. Además, puede detectar las propiedades del terminal y proporcionar una interfaz independiente del mismo.

Como *ncurses* no es una librería estándar de C es necesario indicar al compilador (nosotros usaremos gcc) que la enlace con nuestro programa.

ncurses crea una capa sobre las capacidades del terminal y proporciona funciones para crear interfaces de usuario en modo texto. Permite crear fácilmente aplicaciones basadas en ventanas, menús, paneles y formularios. Además, las ventanas se pueden gestionar de forma independiente, facilitando su movimiento, e incluso se pueden ocultar/mostrar.

3. Instalación

Para instalar la librería en Linux sólo tenemos que usar el sistema de gestión de paquetes correspondiente (a continuación las órdenes para Ubuntu y Fedora):

```
sudo apt-get install libncurses5-dev libncursesw5-dev
sudo dnf install ncurses-devel
```

4. Crear y compilar un ejemplo sencillo

Un programa que use ncurses debe incluir el archivo de cabecera, y gestionar la creación de elementos de la interfaz llamando a determinadas funciones. A continuación se muestra un programa sencillo (***ej1***):

```
#include <ncurses.h>
int main(){
      initscr();              //inicializar modo curses
      printw("Holita");       //imprimir mensaje (aún no se verá)
      refresh();              //mostrarlo en pantalla
      getch();                //esperar la pulsación de una tecla
      endwin();               //terminar el modo curses

      return 0;
}
```

La función initscr() inicializa el terminal en modo "*curses*". En algunas implementaciones borra la pantalla y presenta una pantalla vacía. Es obligatorio llamarla al principio de cualquier programa para inicializar el sistema y asignar memoria para la ventana actual (llamada stdscr) y algunas otras estructuras de datos.

La siguiente línea imprime la cadena "Holita" en la pantalla, de forma parecida a como funciona printf, pero imprimiendo los datos en una ventana llamada stdscr con las coordenadas actuales (y,x). Por defecto, al inicio la coordenada es la (0,0), esto es, en la esquina superior-izquierda de la ventana.

Para evitar parpadeos, los datos se mandan a la ventana stdscr (realmente la función printw actualiza algunas variables, estructuras de datos y escribe los datos en una memoria intermedia correspondiente a stdscr), y sólo cuando se llama a la función de refresco (refresh()) se pasa el contenido de esa memoria intermedia a la pantalla real. De esta forma se pueden hacer múltiples actualizaciones en la pantalla o ventanas imaginarias y hacer el refresco de golpe, una vez que todo el contenido está preparado para mostrarse. Esto mejora el rendimiento y ofrece una mayor flexibilidad. Es por esto que no podemos olvidar la llamada a refresh() después de hacer alguna actualización.

Finalmente, una vez termine el programa y como última sentencia, hay que llamar a la función endwin() para liberar la memoria correspondiente a stdscr y sus estructuras de datos, así como para devolver el terminal a su modo normal.

Para compilarlo debemos hacer uso del compilador de C/C++ de la siguiente forma:

```
gcc hello.c -o hello -lncurses
```

Una vez ejecutemos el programa, veremos la cadena de texto en el terminal:

Una vez analizado un primer programa, pasemos a comentar las diferentes partes de un programa general y estudiar más funciones de la librería.

5. Funciones de la librería

A continuación se detallan las partes principales de un programa *ncurses*, así como las funciones a utilizar en cada parte:

Inicializar el entorno
Para usar *ncurses*, las funciones deben conocer las características del terminal, y el espacio para curscr y stdscr debe estar asignado. La función initscr() realiza ambas cosas. La función initscr() debe utilizarse al principio del programa, antes de intentar usar cualquier otra función de ncurses. Una vez que las estructuras de datos han sido

creadas en memoria, ya se pueden usar funciones para hacer scroll (función scrollok()), para situar el cursor (función leaveok()), crear o borrar nuevas de ventanas (funciones newwin(), derwin(), subwin(), delwin()).

Salida
Las funciones básicas utilizadas para cambiar lo que se mostrará en una ventana son addch() y move(). La primera añade un carácter a las coordenadas (y,x) actuales, mientras que move() cambia las coordenadas (y,x) actuales a cualquier posición. La función mvaddch() combina ambas acciones de una sola vez. Otras funciones de salida son: addstr() y printw(). Como se ha comentado antes, una vez se ha añadido contenido a la ventana, hay que refrescarla usando la función refresh(), lo que actualizará sólo los últimos cambios. Si se quiere actualizar toda la ventana, se debe usar touchwin() para marcar que la ventana entera ha sido cambiada, y así se realiza refresh() del terminal completo.

Entrada
La función complementaria de addch() es getch(). Normalmente, la ventana tendrá activo el eco, por lo que se mostrará la tecla pulsada (se llamará a addch() para sacar el carácter por pantalla). Si se desea, se puede configurar la ventana para que no haya eco, usando la función noecho(). También existen funciones para hacer lectura de cadenas de caracteres (funciones wgetstr() y wscanw()).

Atributos de caracteres y color
ncurses permite establecer los atributos de pantalla, incluyendo salida normal, video-inverso, subrayado, parpadeo y colores. Para poner esos atributos a los caracteres debemos poner el valor del atributo de pantalla deseado en el argumento del carácter de la función addch().

Terminar
Como se ha indicado antes, al terminar el programa se debe liberar la memoria y recursos utilizados por ncurses. Para ello, como última sentencia se debe ejecutar la función endwin(), lo que restaura el terminal (el modo tty) a como estuviera antes de ejecutar nuestro programa, y mueve el cursor abajo a la izquierda.

6. Ejemplo: Mostrar una ventana en el terminal

Para mostrar una ventana, con sus marcos y un color de fondo, debemos utilizar una serie de funciones específicas a tal efecto:

- getmaxyx() : calcula el tamaño en caracteres del terminal en este momento (número de filas y columnas).
- newwin() : crea una nueva ventana del tamaño que indiquemos.
- wbkgd() : establece el color de fondo y de caracteres en la ventana recién creada (la creación de pares de colores se explica a continuación).
- box() : dibuja los marcos de la ventana recién creada. Se pueden indicar qué caracteres usar para las líneas verticales y horizontales.
- mvwprintw() : muestra contenido (cadenas, números, etc) en la ventana actual.
- wrefresh() : refresca el contenido para mostrar en la ventana los últimos cambios realizados en ella.
- werase(): limpia todo el contenido de la ventana indicada, incluso los marcos.

Por otro lado, para mostrar colores y fondo de la ventana, debemos crear “pares de colores”. Para ello, primero debemos inicializar el soporte de colores, a continuación crear dichos pares, y finalmente podremos usar esos pares de colores configurados en las funciones anteriores de manejo de la ventana:

- has_colors() : comprueba si el terminal permite mostrar colores.
- start_color() : inicializa el soporte de colores en la aplicación.
- init_pair() : crea nuevos pares de colores de fondo y de caracteres y le asigna un número al nuevo par, de forma que luego se pueda usar en otras funciones.
- clear() : limpia el contenido de la ventana.

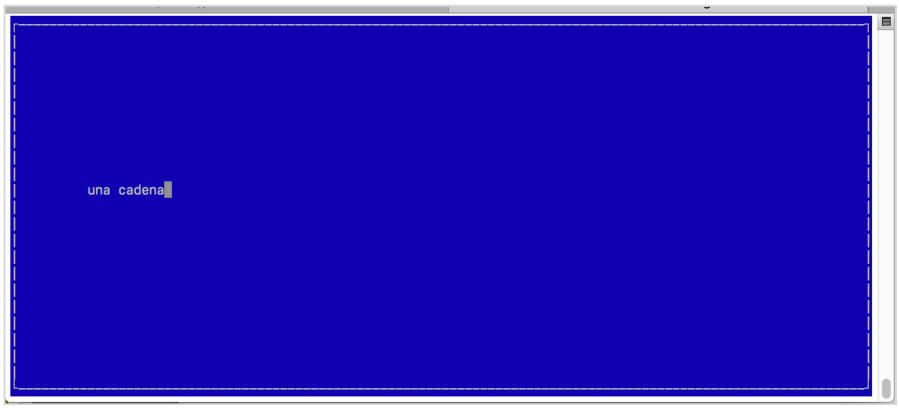
Veamos el ejemplo completo (***ej2***):

```
#include <stdlib.h>
#include <ncurses.h>

int main(void) {
    int rows, cols;

    initscr();

    if (has_colors() == FALSE) {
      endwin();
      printf("El terminal no tiene soporte de color \n");
      exit(1);
      }

    start_color();
    init_pair(1, COLOR_YELLOW, COLOR_GREEN);
    init_pair(2, COLOR_BLACK, COLOR_WHITE);
    init_pair(3, COLOR_WHITE,COLOR_BLUE);   //blanco sobre fondo azul
    clear();

    refresh();
    getmaxyx(stdscr, rows, cols);

    WINDOW *window = newwin(rows,cols,0,0);
    wbkgd(window, COLOR_PAIR(3));
    box(window, '|', '-');

    mvwprintw(window, 10, 10, "una cadena");
    wrefresh(window);

    getch();
    endwin();
    return 0;
}
```

Lo compilamos como ya sabemos:

```
gcc ventana.c -o ventana -lncurses
```

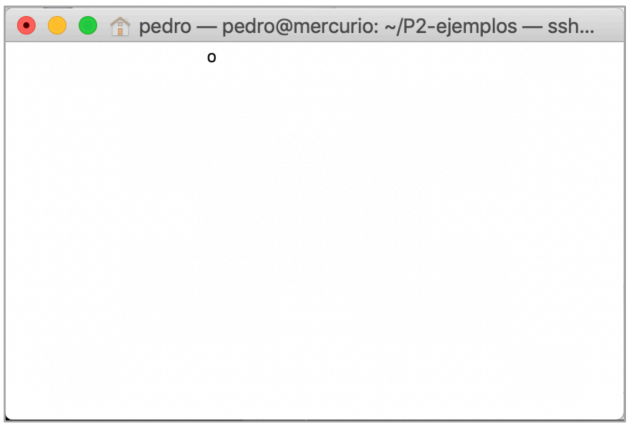
Y vemos que la configuración de colores se aplica a la ventana y al texto mostrado:



7. Ejemplo: Mover una “pelotita” en pantalla

Como ejemplo más completo, veremos cómo hacer el movimiento de una pelotita (realmente no es más que un carácter 'o') en pantalla (***ej3***):

```
#include <ncurses.h>
#include <unistd.h>

#define DELAY 30000

int main(int argc, char *argv[]) {
 int x = 0, y = 0;
 int max_y = 50, max_x = 50;
 int next_x = 0;
 int direction = 1;

 initscr();
 noecho();
 curs_set(FALSE);

 while(1) {
   clear();
   mvprintw(y, x, "o");
   refresh();

   usleep(DELAY);

   next_x = x + direction;

   if (next_x >= max_x || next_x < 0) {
     direction*= -1;
   } else {
     x+= direction;
   }
 }

 endwin();
}
```



Para compilarlo y ejecutarlo sólo tenemos que hacer:

```
gcc -o pelotita pelotita.c -lncurses

./pelotita
```

8. Otras funciones útiles

Hay otras funciones que nos resultarán muy útiles para realizar programas interactivos, como por ejemplo algún tipo de juego:

noecho() -> Las rutinas echo() y noecho() controlan si los caracteres tecleados por el usuario son reproducidos por getch() a medida que son tecleados.

cbreak() -> hace que los caracteres tecleados por el usuario estén inmediatamente disponibles para el programa. La mayoría de los programas interactivos que utilizan curses establecen el modo cbreak().

curs_set(FALSE) -> establece el estado del cursor se establece en invisible, normal o muy visible para una visibilidad igual a 0, 1, 2 respectivamente.

nodelay(stdscr, TRUE) -> La opción nodelay() hace que getch() sea una llamada no bloqueante. Si no hay ninguna entrada lista, getch() devuelve ERR y continúa la ejecución sin más. Si está desactivada (FALSE), getch() espera hasta que se pulse una tecla.

keypad(stdscr, TRUE) -> Si está activada (TRUE), el usuario puede pulsar una tecla de función (como una tecla de flecha) y wgetch() devuelve un único valor que representa la tecla de función, como en KEY_LEFT.

Cuestiones a resolver

El objetivo principal es conocer cómo utilizar la librería *ncurses* para realizar entrada/salida en terminales de texto bajo Linux.

Los requisitos mínimos se valorarán sobre 7 puntos como máximo, los ampliados se valorarán con 3 puntos más como máximo.

Como **requisitos mínimos** se deben realizar y documentar adecuadamente las siguientes tareas:

1. Instalar la librería *ncurses*, crear los programas de ejemplo ofrecidos más arriba (*ej1, ej2* y *ej3*), y comprobar su funcionamiento.
2. Crear un juego sencillo tipo "*pong*" partiendo del ejemplo del movimiento de la pelotita.

Como requisitos ampliados (**opcionales para subir nota**) se propone:

1. que al iniciar el juego se muestre una pantalla de bienvenida en la que se muestren los datos de quienes han realizado el juego y explicando los controles de juego (p.ej. un recuadro con la explicación). Tras una pausa o pulsación de tecla se iniciará el juego en sí mismo.
2. que al terminar cada partida se muestre una pantalla de resumen mostrando el marcador final y felicitando al ganador. Se dará la opción de volver a jugar o terminar el programa.

Como resultado **se mostrará** al profesor el funcionamiento del programa realizado. Se aconseja sacar diversas capturas de pantalla de los programas en las que se muestre el funcionamiento de cada uno.

En el documento a entregar se describirá cómo se ha realizado la instalación y configuración de ncurses y se incluirá tanto el código de los programas desarrollados como capturas de pantalla en las que se muestre el funcionamiento de los mismos.

Normas de entrega

La práctica o seminario podrá realizarse de manera individual o por grupos de hasta 2 personas.

Se entregará como un archivo de texto en el que se muestre la información requerida. También se puede utilizar la sintaxis de Markdown para conseguir una mejor presentación e incluso integrar imágenes o capturas de pantalla. La entrega se realizará subiendo los archivos necesarios al repositorio "**PDIH**" en la cuenta de GitHub del estudiante, a una carpeta llamada "**P2**".

Toda la documentación y material exigidos se entregarán en la fecha indicada por el profesor. No se recogerá ni admitirá la entrega posterior de las prácticas/seminarios ni de parte de los mismos.

La detección de copias implicará el suspenso inmediato de todos los implicados en la copia (tanto de quien realizó el trabajo como de quien lo copió).

Las faltas de ortografía se penalizarán con hasta 1 punto de la nota de la práctica o seminario.

Referencias

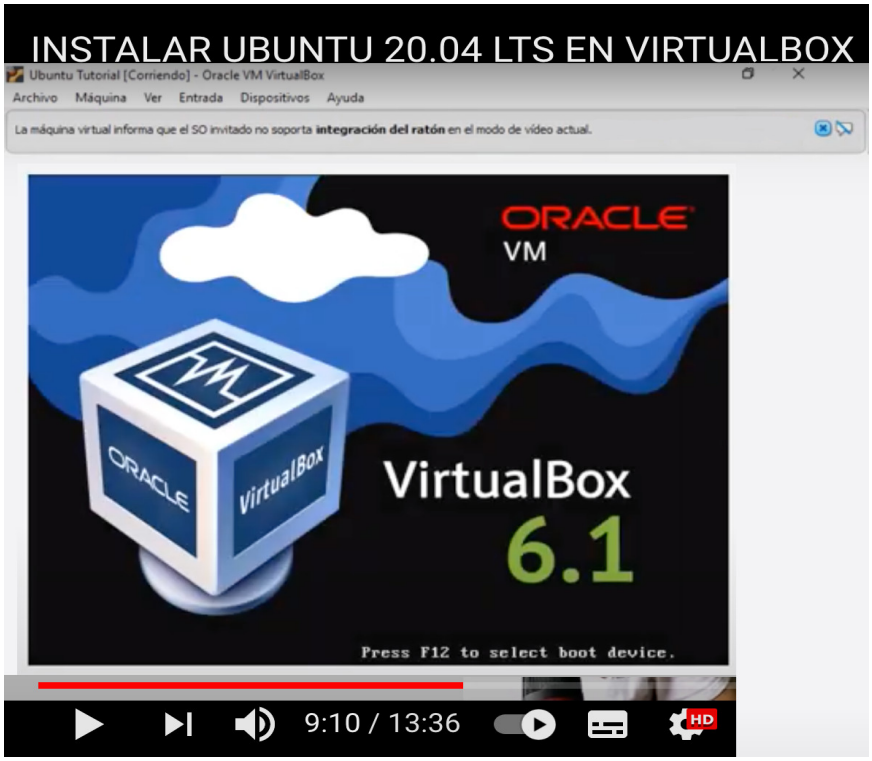
https://es.wikipedia.org/wiki/Ncurses
http://www.gnu.org/software/ncurses/ncurses.html
https://www.mkssoftware.com/docs/man3/ncurses.3.asp
http://www.ditec.um.es/~piernas/manpages-es/otros/tutorial-ncurses.html
https://sinfallas.wordpress.com/2015/04/06/que-es-ncurses/
http://tldp.org/HOWTO/NCURSES-Programming-HOWTO/
https://invisible-island.net/ncurses/ncurses-intro.html
https://www.osetc.com/en/how-to-install-ncurse-library-in-ubuntu-debian-centos-fedora-linux.html
https://www.viget.com/articles/game-programming-in-c-with-the-ncurses-library/
https://www.youtube.com/watch?v=dpyTAikv7s0
https://moidev.com/posts/creando-una-aplicacion-curses-con-python/
https://docs.python.org/es/3/howto/curses.html
https://docs.python.org/3/howto/curses.html
https://magmax.org/blog/python-curses/
https://stackoverflow.com/questions/33271000/install-ncurses-on-python3-for-ubuntu
https://stackoverflow.com/questions/55885055/unable-to-install-curses-python-windows-10
https://gist.github.com/cnruby/960344
https://www.devdungeon.com/content/curses-windows-python
https://github.com/justinmeza/wincurses
https://blogs.windows.com/windowsdeveloper/2016/07/22/fun-with-the-windows-subsystem-for-linux/

Anexo. Instalación de Ubuntu 20.04 en Virtualbox

Para esta práctica se recomienda usar una instalación nativa de una distribución de Linux, pero si eso no es posible, siempre se puede crear una máquina virtual con la que trabajar.

A continuación se ofrecen unos videotutoriales en los que se detalla el proceso de instalación de la distribución Ubuntu 20.04 LTS en Virtualbox:

https://www.youtube.com/watch?v=Nu3EKEc-UTc
https://www.youtube.com/watch?v=GEx046EHphI
https://www.youtube.com/watch?v=L0tGpv1UjoE



Rúbrica para evaluar la práctica 2

El objetivo principal es conocer cómo utilizar la librería *ncurses* para realizar diversos programas que lleven a cabo entrada/salida en terminales de texto bajo Linux.

En la primera parte de la práctica se deben realizar y documentar adecuadamente dos tareas, que contarán hasta un **máximo de 8 puntos**. La siguiente tabla detalla cuánto contará cada tarea desarrollada, según la implementación realizada y su correcto funcionamiento:

Función a implementar	4 puntos	2 puntos	0 puntos
1. Instalar la librería ncurses, crear los programas de ejemplo ofrecidos (ej1, ej2 y ej3) y comprobar su funcionamiento	Instalación de la librería realizada y los tres programas desarrollados y funcionando correctamente	Instalación de la librería realizada pero no se ha conseguido que los tres programas funcionen correctamente	No se ha intentado implementar (ni la instalación ni los programas)
2. Crear un juego sencillo tipo "pong" partiendo del ejemplo del movimiento de la pelotita	Juego implementado adecuadamente y funcionando correctamente (buena presentación, dos palas)	El juego funciona con una sola pala y además la presentación es mejorable	No se ha intentado implementar

Como segunda parte, y para **subir nota**, se proponen **dos ejercicios adicionales**. Cada uno contará un máximo de 1 punto, de forma que haciéndolos se puede llegar a la máxima nota en la práctica:

Ejercicio adicional	1 punto	0,5	0 puntos
1. En el juego del pong, al iniciar el juego se mostrará una pantalla de bienvenida en la que se muestren los datos de quienes han realizado el juego y explicando los controles de juego (p.ej. un recuadro con la explicación). Tras una pausa o pulsación de tecla se iniciará el juego en sí mismo	El programa funciona correctamente y la pantalla de presentación e información se muestra correctamente de acuerdo con las instrucciones dadas	En el programa la pantalla de presentación e información no se muestra correctamente de acuerdo con las instrucciones dadas o presenta algún fallo	No se ha intentado implementar
2. En el juego del pong, al terminar cada partida se mostrará una pantalla de resumen mostrando el marcador final y felicitando al ganador. Se dará la opción de volver a jugar o terminar el programa	El programa funciona correctamente y la pantalla de "final de juego" se muestra correctamente de acuerdo con las instrucciones dadas	En el programa la pantalla de "final de juego" no se muestra correctamente de acuerdo con las instrucciones dadas o presenta algún fallo	No se ha intentado implementar