Departamento de Ingeniería Informática
Universidad de Santiago de Chile

Sistemas Operativos
Primer Semestre 2017
Laboratorio 3

Profesores:

- Fernando Rannou (fernando.rannou@usach.cl)
- Cristóbal Acosta (cristobal.acosta@usach.cl)
- Miguel Cárcamo (miguel.carcamo@usach.cl)

Ayudantes:

- Natalia Pérez (natalia.perez.g@usach.cl)
- Fernanda Estay (fernanda.estay@usach.cl)

1. Objetivo

Este laboratorio tiene como objetivo que los estudiantes apliquen los conceptos de sección crítica, exclusión mutua, creación de hebras y concurrencia, mediante el soporte de un sistema operativo basado en el núcleo Linux y el lenguaje de programación C.

2. Conceptos

2.1. Hebras

Los hilos POSIX, usualmente denominados pthreads, son un modelo de ejecución que existe independientemente de un lenguaje, así como un modelo de ejecución paralelo. Permite que un programa controle múltiples flujos de trabajo que se superponen en el tiempo.

Para poder utilizar una hebra, es necesario incluir la libreria `pthread.h`, luego definir la variable de referencia a la hebra para esto se utiliza el tipo `pthread_t`.

Debe existir una función de la forma `void * function (void * params)`, la cual recibe parametros del tipo void, por lo cual es necesario castear el o los parametros y así poder utilizarlos sin problemas.

Algunas funciones para manejar las hebras son:

- `phread_create`: Función que crea la hebra, y recibe como parámetros la variable de referencia a la hebra que desea crear, luego los atributos, si estos no se quieren modificar simplemente se deja NULL, luego, el nombre de la función que la hebra ejecutará, y que cumpla con la descripción antes mencionada y por último los parametros de entrada previamente casteados.
- `pthread_join`: Función donde la hebra que la ejecuta, espera por la hebra que se ingresa por parametro de entrada.
- `pthread_mutex_init`: Función que inicializa un mutex, pasando por parametros la referencia al mutex, y los atributos con que se inicializa.

Para más información puede ver el siguiente enlace: ncurses: `http://pubs.opengroup.org/onlinepubs/7908799/xsh/pthread.h.html`

2.2. Ncurses

Ncurses es una biblioteca de programación de C, que contiene funciones para poder generar una interfaz de programación de aplicaciones (API), estas son basadas en texto de una manera independiente del terminal.

Lo primero que se debe realizar es instalar la libreria, ejecutando el siguiente código en la terminal.

```
apt-get install ncurses-dev
```

Al momento de utilizar esta libreria es necesario agregar a la linea de comando `-lncurses`, de la siguiente forma:

```
gcc main.c -o lab -lncurses
```

Para poder ejecutar la interfaz gráfica, primero se debe incluir la libreria con `#include<ncurses.h>`, además se puede hacer uso de las siguientes funciones:

- `initscr()`: Función para inicializar la API.
- `mvprintw(int x, int y, char* texto)`: Esta función es similar a print, pero imprime la información en la posición (x, y) dentro de la API.
- `init_pair(int a, int b, int c)`: Función que se utiza para crear pares de colores, donde a es el identificador del color creado, b es el identificador del color del texto, y el c es el color de fondo.
- `attron(COLOR_PAIR(int i))`: Función que indica que lo que se imprima de ahora en adelante en el código, tendrá los colores determinados en el identificador 1.
- `move(int x, int y)`: Modifica la posición del cursor y lo deja en el punto (x,y).
- `getch()`: Función que pide un dato desde terminal. Puede ser utilizada para que la API reciba un dato antes de terminar y no se cierre inmediatamente.
- `endwin()`: Función que cierra la API.

Para más información puede ver el siguiente enlace: `http://www.ditec.um.es/~piernas/manpages-es/otros/tutorial-ncurses.html#La_Libreria_Curses`

3. Enunciado

En este laboratorio se simulará el *Torneo del poder – Chikara no Taikai*, donde todos los dioses de la destrucción se reunirán en la batalla más fiera jamás vista en Dragon Ball, que traerá como consecuencia la destrucción de universos enteros a manos de los guerreros más poderosos que existen en todos los mundos.

El programa recibirá un archivo de texto que por cada linea contiene información de un guerrero, de esta forma::

- XXXX: un número de cuatro dígitos que representará la defensa total del guerrero en términos de HP (hit points),
- Y: un número que indica el color del universo que representa el guerrero.
- ZZ: un número de dos dígitos que indica el universo al que pertenece el guerrero.
- El nombre del guerrero. Cabe destacar que la lectura debe ser dinámica, ya que todos los nombres tienen distinto largo, por ende cada fila tiene distinto largo.

El nombre del archivo de entrada será ingresado por linea de comando, haciendo uso de `getopt`, además del tamaño del tablero cuadrado (`ancho_tablero`), y la opción de mostrar la información por pantalla, si `opcion_debug` es 1 se muestra en pantalla un resumen, y si es 0 no muestra la información. De la siguiente forma:

```
>> gcc main.c -o laboratorio3 -l ncurses
>> ./laboratorio3 -I archivo_entrada.txt -N ancho_tablero -D opcion_debug
```

Cada guerrero será ejecutado en una hebra distinta, donde se puede mover dentro del tablero (de N x N) hacia arriba, abajo, a la izquierda y derecha o bien podría mantenerse en la misma posición de forma aleatoria, y cuando lo haga, si se encuentra cerca de otro guerrero (de forma horizontal o vertical, pero no diagonal), lo enfrentará atacándolo. La fuerza de ataque dependerá de cuanto tiempo lleva sin atacar, cada vez que no realiza un golpe significa que está guardando Ki, por cada energía (ki) le restará al enemigo 5 HP. Lo que provocará que el guerrero que es atacado baje su barra de HP, cuando llegue a tener 0, el guerrero será eliminado del torneo. De esta forma sólo quedará uno o más guerreros del mismo universo, que representarán al universo vencedor. Cuando esto ocurra, el proceso debe terminar de forma limpia mostrando por pantalla un mensaje que indique cuál fue el universo vencedor (no se puede utilizar `Ctrl+C`).

Con el uso de la libreria `ncurses`, se busca que muestre la interacción por cada movimiento que realice cada representante, cada guerrero se demora un segundo en realizar el siguiente movimiento. Al iniciar, el representante se ubicará en un lugar aleatorio dentro del tablero, se identificará con la primera letra de su nombre en mayúscula y el color de la letra del universo que representa. Cuando un guerrero sea atacado, se mostrará por un segundo donde está con fondo rojo. Si el guerrero llega a HP 0, deberá eliminarse del tablero.

Además se pide que arriba del tablero, se muestre una pantalla de lo que está pasando, que muestre cuanto HP le quedan a los participantes, cuanto ki llevan acumulado, y si han habido participantes eliminados. Esta pantalla se actualiza cada 5 segundos.

Se solicita realizar un buen análisis previo al desarrollo de la tarea, ya que debe proveer la correcta sincronización de las hebras para que no exista condición de carrera. Por ejemplo, no pueden ubicarse dos guerreros en la misma casilla. Queda a su criterio como va a manejar la sincronización de las hebras, para ello puede utilizar todo el material que se ha visto en clases. Además usted debe analizar una forma para que lo que ocurra en el tablero se muestre por pantalla en tiempo real. Finalmente, debe encontrar una estrategia para mostrar por pantalla el mensaje que indica qué universo ganó el torneo.

4. Entregables

El laboratorio es en parejas y se descontará 1 punto por día de atraso. Debe subir en un archivo comprimido a usachvirtual los siguientes entregables:

- Makefile: Archivo para make que compila el programa.
- main.c: archivo con el código. Puede incluir otros archivos fuente.

Recuerde que todas las funciones deben estar comentadas, explicadas de forma entendible. Si una función no está explicada se bajará puntaje. Además, trabajos con códigos que hayan sido copiados de un trabajo de otro grupo serán calificados con la nota mínima.

FECHA DE ENTREGA: 01/05/2017 hasta las 23:55.

5. Ejemplo

Por ejemplo si recibe el archivo:

```
8590 2 07 Goku
8591 2 07 Vegeta
6973 2 07 Piccolo
5432 3 02 Jerez
6773 3 02 Sour
4689 4 03 Mosco
6981 5 06 Cabba
7891 5 06 Frost
```

La Figura 1 muestra el tablero inicializado si es que el ancho del cuadrado es 20.

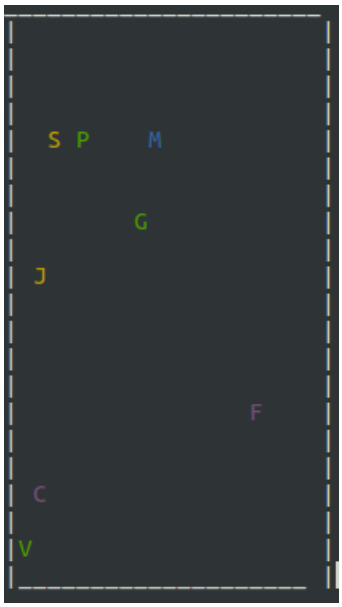

Figura 1: Ejemplo del tablero inicializado

Si Sour se mueve a la derecha, quedaría al lado de Piccolo, por lo que Sour ataca a Piccolo, y Piccolo al recibir el ataque cambiaría el fondo de donde se encuentra, como lo muestra la Figura 2.

La pantalla de información que se pide, puede tener una estructura como se muestra en la Figura 3.

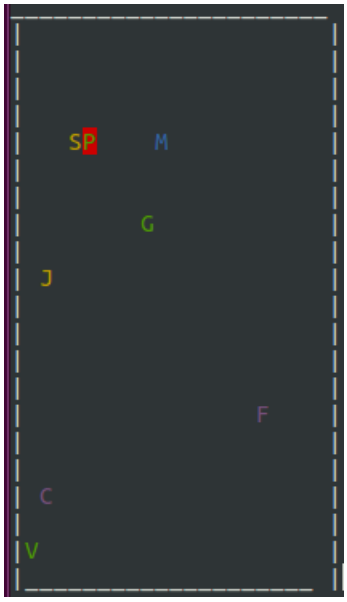

Figura 2: Ejemplo de un ataque

Guerrero	Hp	ki
Goku	8590	0
Vegeta	8591	0
Piccolo	6968	0
Jerez	5432	0
Sour	6773	1
Mosco	4689	0
Cabba	6981	0
Frost	7891	0

Figura 3: Ejemplo de pantalla de información