NOMBRE Y APELLIDOS: _____ NIA: _____

Examen de Sistemas Operativos - 2º Parcial
17 de Junio de 2010

NOTAS:

* Para la realización del presente examen se dispondrá de 2 horas.
* No se pueden utilizar libros ni apuntes, ni usar móvil (o similar).
* Responda cada pregunta en hojas distintas.

Teoría . (2 ptos)

a.- ¿Qué es el BCP? Indica 3 datos que se almacenen en el mismo

b.-¿Qué ocurre cuando se genera un trap durante la ejecución de un proceso? ¿Indica alguna forma de generar un trap?

c.-. En un sistema de ficheros ext3 ¿Qué se almacena en los directorios? ¿Dónde se almacena el tamaño del fichero? Si tuvieras que configurar un disco con sistema de ficheros ext3 que se va a utilizar para almacenar únicamente grandes ficheros con imágenes de satélite, ¿qué tamaño de bloque elegirías?

Ejercicio 1. (2,5 ptos)

Escribe en lenguaje de programación C un programa que cuando se pulse Control-C (que manda la señal SIGINT al proceso) se cree un proceso hijo que imprima el mensaje Soy un proceso hijo.

El proceso padre esperará la finalización del hijo y cuando éste termine imprimirá el mensaje Mi hijo acaba de morir.

Ejercicio 2. (3 ptos)

Se desea realizar una aplicación con threads que simule el funcionamiento de la entrada en un recinto en el que existen 2 filas y en el que debe entrar, alternativamente, una persona de cada fila.

NOMBRE Y APELLIDOS: _____ NIA: _____

Examen de Sistemas Operativos - 2º Parcial
17 de Junio de 2010

El programa principal irá generando una serie de threads que irá asignando, aleatoriamente, a la fila 1 o a la fila 2. El número total de threads generados será 20. A cada thread creado le asignará un identificador del 0 al 19.

Para entrar en el recinto se escogerá siempre un thread de una final distinta a la anterior y se empezará siempre por la fila1. Según van entrando los threads éstos irán mostrando los mensajes Entra alguien de la Fila N: ID , donde N es el número de fila (1 o 2) y ID es el identificador asignado a ese thread (entre 0 y 19)

Por ejemplo si el programa principal crea los threads:

0 de la fila 2,

1 de la fila 2,

2 de la fila 2,

3 de la fila 1,

4 de la fila 2,

5 de la fila 1,

6 de la fila 1,.

En pantalla aparecerán los mensajes, escritos por los threads correspondientes:

Entra alguien de la Fila 1: 3

Entra alguien de la Fila 2: 0

Entra alguien de la Fila 1: 5

Entra alguien de la Fila 2: 1

Entra alguien de la Fila 1: 6

Entra alguien de la Fila 2: 2

No se debe asegurar que el orden de entrada de los threads de una fila dada es el orden de creación pero si que entra siempre, de forma alternativa, un thread de cada fila.

Ejercicio 3. (2,5 ptos)

a) Se tiene un disco con sistema de ficheros Unix con las siguientes características:
   - Tamaño del bloque de 2 KBytes.
   - Direcciones de los bloques: 4 bytes.
   - Estructura del i-nodo:
     - 10 punteros directos.
     - 1 puntero indirecto simple.
     - 1 puntero indirecto doble.
     - 1 puntero indirecto triple.

Indica cuál es el número de bloques que ocupa un fichero de 10 MBytes, incluyendo tanto los bloques de datos como los de direcciones.

NOMBRE Y APELLIDOS: _____ NIA: _____

Examen de Sistemas Operativos - 2º Parcial
17 de Junio de 2010

b) Se tiene un sistema de ficheros tipo Unix con la siguiente información:

Tabla de I-nodos:

Nº Inodo	1	2	3		
Tipo	Directorio	Directorio	Fichero		
Contador Enlaces Fis.	3	2	1		
Dirección Bloque Datos	11	12	13		
.					

Bloques de datos:

Nº Bloque	11	12	13		
Contenido	. 1	. 2	Datos del Fichero f1		
	.. 1	.. 1			
	d 2	f1 3			

Indica cómo quedan los i-nodos y los bloques de datos después de realizar cada una de las siguientes operaciones (Hacer una tabla de i-nodos y de bloques de datos por cada apartado):

1- ln s /d/f1 /d/f2
2- ln /d/f1 /d/f3
3- rm /d/f3

NOMBRE Y APELLIDOS: _____ NIA: _____

Examen de Sistemas Operativos - 2º Parcial
17 de Junio de 2010

SOLUCIÓN

Teoría 1.

El BCP es el bloque de control de proceso que permite al SO almacenar información sobre los procesos en ejecución (almacena el valor de los registros del proceso cuando éste no está utilizando el procesador, las propiedades del proceso (pid, usuario y grupo real y efectivo, etc).

Teoría 2.

Los traps son interrupciones sw invocadas por el proceso (expresamente con instrucciones trap o provocadas por situaciones inesperadas (división por 0, etc.))

Cuando se genera un trap se pasa a modo nucleo y el SO pasará (después de que se salven adecuadamente los valores de los registros del proceso actual) a ejecutar el manejador del trap asociado (su rutina de atención a la interrupción o ISR).

Se puede provocar un trap solicitando servicios al sistema operativo como por ejemplo el que se produce al invocar a la función fork()

Teoría 3.

En los directorios sólo se almacena el nombre del fichero o directorio y el número del inodo correspondiente.

El tamaño del fichero se almacena en el inodo

Para un disco que sólo almacena ficheros grandes elegiría el tamaño máximo de bloque 8Kb porque así se necesita menos espacio para almacenar información de acceso a los bloques.

Ejercicio 1:

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

int crearhijo=0;
void controlC(int s){
  crearhijo=1;
}
main () {

signal (SIGINT, controlC);
printf ("Pulse CTRL-C para crear un proceso hijo\n");
while (!crearhijo) {
  sleep (1);
}
if (crearhijo)
  if (fork()==0)
          printf ("Soy un proceso hijo\n");
```

NOMBRE Y APELLIDOS: _____ NIA: _____

Examen de Sistemas Operativos - 2º Parcial
17 de Junio de 2010

```
  else {
    wait(NULL);
    printf ("Mi hijo acaba de morir");
  }
}
```

Ejercicio 2:

```
/* Este programa crea 20 threads de dos tipos. Usuarios de la fila 1 y usuarios de la fila 2
 * La entrada tiene que ser alterna, una usuario de cada fila*/
 /* José Manuel Pérez Lobato */
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#define MAX         20      /* Numero máximo*/
#define TRUE        1
#define FALSE       0

pthread_mutex_t mutex;     /* mutex para controlar el acceso compartido */
pthread_cond_t esperafila1;   /* controla la espera de la fila 1 */
pthread_cond_t esperafila2;   /* controla la espera de la fila 2 */
int turnofila1=TRUE;
int turnofila2=FALSE;

int espera=1;

void  *fila1(void *p)  {   /* código de los usuarios de la fila 1*/
int *num, usuario;
    num= p;
    usuario= *num;
    espera=0;

      pthread_mutex_lock(&mutex);
        while (turnofila1 == FALSE)
            pthread_cond_wait(&esperafila1, &mutex); // Espera que entren de la otra fila
            printf ("Entra alguien de la FILA 1: %d\n", usuario);
            turnofila2=TRUE;
            turnofila1=FALSE;
        pthread_cond_signal(&esperafila2); /*Señaliza que entró el de la fila 1*/
      pthread_mutex_unlock(&mutex);
    pthread_exit(0);
}
void  *fila2(void *p)  {   /* código de los usuarios de la fila 2*/
int *num, usuario;
    num= p;
    usuario= *num;
    espera=0;
```

NOMBRE Y APELLIDOS: _____ NIA: _____

Examen de Sistemas Operativos - 2º Parcial
17 de Junio de 2010

```
    pthread_mutex_lock(&mutex);
     while (turnofila2 == FALSE)
        pthread_cond_wait(&esperafila2, &mutex); /*Espera que entren de la otra fila */
        printf ("Entra alguien de la FILA 2: %d\n", usuario);
        turnofila2=FALSE;
        turnofila1=TRUE;
      pthread_cond_signal(&esperafila1); /*Señaliza que entró el de la fila 1*/
    pthread_mutex_unlock(&mutex);
    pthread_exit(0);
}


main(int argc, char *argv[]){
   pthread_t th[MAX];
   pthread_attr_t attr;
   int i, afila;

   srandom (getpid());
   pthread_mutex_init(&mutex, NULL);
   pthread_cond_init(&esperafila1, NULL);
   pthread_cond_init(&esperafila2, NULL);
   pthread_attr_init(&attr);
   for (i=0; i<MAX; i++){
    afila=random ()%2 +1 ;
    if (afila==1)
       pthread_create(&th[i], &attr, fila1, &i);
    else
       pthread_create(&th[i], &attr, fila2, &i);
     while (espera==1);
     espera=1;
printf ("Creado %d a fila %d\n", i, afila);
   }
   for (i=0; i<MAX; i++)
    pthread_join(th[i], NULL);

   pthread_mutex_destroy(&mutex);
   pthread_cond_destroy(&esperafila1);
   pthread_cond_destroy(&esperafila2);
   exit(0);
}
```

Ejercicio 3:

a)

NOMBRE Y APELLIDOS: _____ NIA: _____

Examen de Sistemas Operativos - 2º Parcial
17 de Junio de 2010

En cada bloque de direcciones caben 512 direcciones de bloque: 2KBytes /4 bytes =512 posiciones.
Los 10 apuntadores simples apuntarán a los 20 primeros Kbyte del fichero.
Cada bloque de direcciones que se necesite apuntará a 512 apuntadores*2Kbytes= 1 Mbyte del fichero

Por tanto para llegar a 10 Mbyte necesitamos 10 bloques de direcciones.
El primero lo obtenemos del apuntador indirecto simple, que apunta directamente a uno de ellos. Para los 9 restantes necesitamos del apuntador indirecto doble que apuntará a 1 bloque de direcciones que apuntará a esos 9 bloques de direcciones necesarios.
Luego se necesitan 10 + 1= 11 bloques de direcciones
El tamaño en bloques de datos del fichero es de : 10 Mbyte / 2Kbytes =10240 / 2 = 5120 bloques
Luego el tamaño total son 5131 bloques más el inodo.

b)

1-

Tabla de I-nodos después de ln s /d/f1 /d/f2:

Nº Inodo	1	2	3	4	
Tipo	Directorio	Directorio	Fichero	Enlace Simbólico	
Contador Enlaces Fis.	3	2	1	1	
Dirección Bloque Datos	11	12	13	14	
.					

Bloques de datos después de ln s /d/f1 /d/f2:

Nº Bloque	11	12	13	14	
Contenido	. 1	. 2	Datos del fichero	/d/f1	
	.. 1	.. 1			
	d 2	f1 3			
		f2 4			

2-

NOMBRE Y APELLIDOS: _____ NIA: _____

Examen de Sistemas Operativos - 2º Parcial
17 de Junio de 2010

Tabla de I-nodos después de ln /d/f1 /d/f3:

Nº Inodo	1	2	3	4	
Tipo	Directorio	Directorio	Fichero	Enlace Simbólico	
Contador Enlaces Fis.	3	2	2	1	
Dirección Bloque Datos	11	12	13	14	
.					

Bloques de datos después de ln /d/f1 /d/f3:

Nº Bloque	11	12	13	14	
Contenido	. 2	. 2	Datos del fichero	/d/f1	
	.. 2	.. 1			
	d 3	f1 3			
		f2 4			
		f3 3			

3-

Tabla de I-nodos después de rm /d/f1:

Nº Inodo	1	2	3	4	
Tipo	Directorio	Directorio	Fichero	Enlace Simbólico	
Contador Enlaces Fis.	3	2	1	1	
Dirección Bloque Datos	11	12	13	14	
.					

Bloques de datos después de rm /d/f1:

Nº Bloque	11	12	13	14	
Contenido	. 2	. 2	Datos del fichero	/d/f1	
	.. 2	.. 1			
	d 3	f1 3			
		f2 4			
		f3 3			