UNIVERSITAT POLITÈCNICA DE VALÈNCIA
Departamento de Informática de Sistemas y Computadores (DISCA)

fSO

Ejercicio de Evaluación FINAL
23 de Enero de 2020

etsinf
Escola Tècnica Superior d'Enginyeria Informàtica

APELLIDOS		NOMBRE		Grupo
DNI		Firma		

- No desgrape las hojas.
- Conteste exclusivamente en el espacio reservado para ello.
- Utilice letra clara y legible. Responda de forma breve y precisa.
- El examen consta de 8 cuestiones, cuya valoración se indica en cada una de ellas.
- Recuerde que debe justificar sus cálculos o respuestas para obtener buena calificación.

1. Suponga la ejecución del siguiente código sin errores y responda de forma justificada:

(1,2 puntos = 0,6 + 0,4 + 0,2)

```
 1 #include <all_needed.h>
 2 int main() {
 3     int i;
 4     pid_t pid;
 5
 6     for (i=0; i<2; i++) {
 7         pid = fork();
 8         if (pid != 0) {
 9             sleep(10 - 4 * i));
10             break;
11         }
12     }
13     if (i == 2) sleep(2);
14     while (wait(NULL) =! -1);
15     exit(0);
16 }
```

1

a) Indique justificadamente el número de procesos que se crean al ejecutarlo y dibuje el esquema de parentesco entre dichos procesos.

b) Indique de forma justificada, para cada uno de los procesos que se crean, si pueden permanecer zombies o no y por cuánto tiempo podrían estar en dicho estado.

c) Considerando un sistema multiprogramado y que el tiempo de ejecución de las instrucciones es despreciable en comparación con el tiempo de sleep ¿cuánto tiempo tarda en ejecutarse el programa?

2. Sea un sistema de tiempo compartido con un planificador a corto plazo con dos colas de procesos preparados: Cola0 gestionada con un algoritmo Round Robin con quantum q=1ut y Cola1 gestionada con FCFS. La planificación entre colas es por prioridades expulsivas, siendo la Cola0 la más prioritaria. Los procesos nuevos y los que provienen de E/S acceden al sistema por la Cola0, y se degradan a la Cola1 cuando consumen un quantum de tiempo y no acaban su ráfaga de CPU. Si se producen varios eventos en un mismo instante el orden de inserción en cola de los procesos es: nuevo, procedente de E/S y fin de quantum. Existe un único dispositivo de E/S que utiliza una planificación FCFS. A dicho sistema llegan 3 procesos A, B y C, cuyos instantes de llegada y esquema de solicitud de ráfagas de CPU y E/S es el siguiente:

Proceso	Instante de llegada	Ráfagas de CPU y E/S
A	0	2CPU + 3E/S + 4CPU
B	1	1CPU + 1E/S + 2CPU
C	2	4CPU + 2E/S + 1CPU

(1,6 puntos=1,2+0,4)

a) Represente mediante diagrama temporal la ocupación de CPU, del periférico de E/S y de las colas de preparado

T	Cola1	Cola0	CPU	Cola E/S-->	E/S	Evento
0						
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						

b) Indique tiempo de Retorno y Tiempo de Espera para cada proceso

	Proceso A	Proceso B	Proceso C
Tiempo Retorno			
Tiempo Espera			

3. En el siguiente trozo de código falta completar las líneas 7,13 y 25, cada una con una instrucción.

(1,2 puntos = 0,4 + 0,4 + 0,4)

```
 1  #include <all_needed.h>
 2
 3  int V = 0;
 4  pthread_t thr1, thr2;
 5
 6  void *thread1( void *ptr ){
 7     /**complete**/
 8     V = 1;
 9     printf("T1 V=%d\n",V);
10  }
11
12  void *thread2( void *ptr ){
13     /**complete**/
14     V = V + 10;
15     printf("T2 V=%d\n",V);
16  }
17
```

```
18  int main(int argc, char* argv[]){
19     pthread_attr_t atrib;
20     pthread_attr_init(&atrib);
21
22     pthread_create(&thr1,&atrib, thread1, NULL);
23     pthread_create(&thr2,&atrib, thread2, NULL);
24
25     /**complete**/
26     V = V + 100;
27     printf("Main V=%d\n",V);
28  }
...
```

A continuación se proponen varias posibilidades para completar dichas líneas. Indique para cada una de estas opciones los mensajes que se mostrarán por pantalla tras su ejecución y justifique su respuesta.

a) línea 7 - sleep(1);
línea 13 - sleep(3);
línea 25 - sleep(2);

b) línea 7 - sleep(5);
línea 13 - sleep(2);
línea 25 - pthread_exit(0);

c) línea 7 - sleep(5);
línea 13 - pthread_join(thr1,NULL);
línea 25 - pthread_join(thr2,NULL);

4. Sea el siguiente código que hace uso de hilos y semáforos POSIX:

```
#include <semaphore.h>

sem_t sem_A, sem_B;

void *compute(void *param) {

   ...
}
```

```
int main(void) {

   pthread_t th[10];
   pthread_attr_t attr;
   int n;
   sem_t sem_C;

   sem_init(&sem_A, 0, 1);
   sem_init(&sem_B, 0, 0);
   sem_init(&sem_C, 0, 4);

   pthread_attr_init(&attr);
   for (n=0; n<10; n++) {
      pthread_create(&th[n], &attr, compute, NULL);
   }

   ...
}
```

El programador desea emplear los semáforos para resolver diferentes cuestiones que surgirán al ejecutarse concurrentemente los hilos que se crean.

(1,0 puntos = 0,4 + 0,6)

4

a) Indique, tres situaciones o tres fines para los que son útiles los semáforos en los entornos de programación concurrente

b) Atendiendo tanto al ámbito en el que se han declarado los semáforos *semA*, *semB* y *semC,* así como a su inicialización en el código propuesto, y teniendo en cuenta que los semáforos serán empleados por los hilos *th[n]* en la función ***compute*(),** indique qué semáforo *semA*, *semB* o *semC* sería el indicado para cada una de las situaciones descritas en el apartado anterior (a).

5. Teniendo en cuenta el mecanismo de herencia de procesos en Unix y las llamadas POSIX, responda a los siguientes apartados: **(1,2 puntos =0,6+0,6)**

5

a) Suponga que no se producen errores en las llamadas al sistema y complete el siguiente programa en C con las instrucciones y llamadas al sistema necesarias (una por línea con número subrayado) para que se llegue a ejecutar la siguiente línea de órdenes: **$ cat < f1 2> ferr | grep ".c" > f2**

```
 1  #include <unistd.h>
 2. #include <fcntl.h>
 3. #define readfile O_RDONLY
 4. #define newfile (O_RDWR | O_CREAT | O_TRUNC)
 5. #define mode644 (S_IRUSR | S_IWUSR | S_IRGRP | S_IROTH)
 6. int main() {
 7.    int pipeA[2];
 8.    int fd1,fd2;
 9.
10.    if (fork()) {
11.        fd1 = open("ferr", newfile, mode644);
12.        fd2 = open("f1",readfile);
13.
14.
15.
16.        close(pipeA[0]); close(pipeA[1]); close(fd1); close(fd2);
17.        execlp("cat", "cat", NULL);
18.     } else {
19.        fd1 = open("f2",newfile, mode644);
20.
21.
22.        close(pipeA[0]); close(pipeA[1]);close(fd1);
23.        execlp("grep", "grep", ".c", NULL);
24.        }
25.     }
26.     return 0;
27. }
```

b) Rellene la tabla de descriptores de archivos del proceso padre tras ejecutar la línea 15 y del proceso hijo tras la línea 21. Las tablas deben ser correctas con los requisitos y la implementación de la sección a)

Tabla del Proceso padre en **15**	
0	
1	
2	
3	
4	
5	
6	
7	

Tabla del Proceso hijo en **21**	
0	
1	
2	
3	
4	
5	
6	
7	

6. Suponga un sistema de archivos Minix de 1 GBytes con las siguiente características:

- Nodos-i con un tamaño de 32 Bytes con 7 punteros directos a zonas, 1 indirecto y 1 doblemente indirecto
- Punteros a zona de 16 bits (2Bytes)
- Entradas de directorio de 16 Bytes (14 Bytes para el nombre y 2 Bytes para el nodo-i)
- **1 Bloque = 1 Zona = 2 KBytes**

(1,2 puntos = 0,8 +0,4)

6

a) Indique de forma justificada los tamaños de cada elemento de la cabecera al formatearlo para almacenar 32768 nodos-i (32K nodos-i).

Arranque	Super bloque	Mapa de bits de Nodos-i	Mapa de bits de Zonas	Nodos- i	Zonas de datos

b) Dicho sistema contiene el directorio raíz y dentro de él 30 directorios vacíos y 30 archivos. La mitad de los archivos son archivos regulares y la otra mitad son enlaces simbólicos a esos archivos regulares.
Indique de forma justificada el número de i-nodos ocupados

Indique de forma justificada el tamaño en Bytes del directorio raíz.

7. Un sistema con paginación por demanda con dos niveles de paginación tiene un tamaño máximo de proceso de 4GBytes, un tamaño de página de 4KBytes y un total de 4096 entradas en el primer nivel de paginación. La siguiente tabla muestra información relativa al instante t=25 para el proceso P y el proceso S del sistema operativo, el cual se haya ubicado en memoria en los marcos con las direcciones más altas disponibles en este sistema.

Proceso	Marco	Página	T. último acceso	BitV (validez)
P	0x4A000	0xC71FF	5	1
P	0x4A001	0xC7200	10	1
P	-	0xA70C0	-	0
P	0x4A003	0xA73DC	15	1
S	0xFFFFE	0xB7001	20	1
S	0xFFFFF	0xB7002	25	1

Partiendo del enunciado y la información de la tabla, conteste cada apartado de forma justificada:

(1,1 punto = 0,5 + 0,2 + 0,4)

7

a) Formato de la dirección lógica y física con nombre y número de bits de cada campo o elemento:

Dirección lógica (nombre y número de bits de cada elemento)

Dirección física(nombre y número de bits de cada elemento)

b) Determine el número máximo de descriptores de páginas de 2º nivel que el proceso P puede utilizar.

c) Indique las Direcciones Físicas correspondiente, así como si se produce fallo de página, al solicitar acceso a las Direcciones lógicas propuestas a continuación:

Proceso → Dir. Lógica	***Dir. Física (o FP si hay fallo de página)***
P→ 0xA70C0102	
P→ 0xA73DC102	
S→ 0xB7000102	
S→ 0x B7001003	

8. Considere el sistema descrito en la pregunta 7 y la asignación inicial de marcos que se detalla en su tabla para el instante t=25. Suponga que se utiliza paginación por demanda con un algoritmo **LRU** con reemplazo **LOCAL,** gestionado mediante contadores y que el sistema asigna 4 marcos al proceso P y 2 al proceso S.

(1,5 puntos = 0,2 + 1,0 + 0,3)

8

a) A partir del instante t=26 los procesos emiten la siguiente secuencia de direcciones lógicas (en hexadecimal): P:A70C0102, P:A74D0F02, P:A73DC102, P:C7200C10, P:C7200C11, P:A70C0102, S:B7003A00, S:B7001000

Obtenga la serie de referencias correspondiente a la secuencia anterior:

b) Rellene la siguiente tabla con la evolución del contenido de Memoria Principal para la serie de referencias obtenida en el apartado anterior. En cada recuadro anote en su parte superior la página que corresponda y en la inferior el valor del contador. Puede rellenar sólo las casillas donde haya algún cambio.

Página→	(asignación inicial)							
Marco ↓	t= 25	t=	t=	t=	t=	t=	t=	t=
4A000								
4A001								
4A002								
4A003								
FFFFE								
FFFFF								
Marque fallos de página y reemplazos:								

Indique el número de FALLOS DE PÁGINA TOTALES :

c) Indique de forma justificada si el algoritmo LRU ha conseguido una gestión óptima de los marcos, es decir, si el número de fallos de página ha sido el mismo que habría obtenido el algoritmo de reemplazo óptimo.