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

fSO

Ejercicio de Evaluación Final
23 de Enero de 2015

etsinf
Escola Tècnica Superior d'Enginyeria Informàtica

SOLUCIONES

1.Indique, si es que las hay, cuales son las diferencias y semejanzas entre un sistema operativo multiprogramado y un sistema de tiempo compartido.

(0,5puntos)

1	Los sistemas multiprogramados permiten la ejecución de varios procesos ejecutándose de forma concurrente, para ello se van alternado en la utilización de la CPU. Los sistemas de tiempo compartido son sistemas multiprogramados que permiten la interacción usuario máquina

2. Para un sistema operativo dotado de planificador de procesos a Corto y Medio Plazo, exponga un ejemplo donde puedan darse cada una de las situaciones propuesta a continuación y diga sobre qué planificador de los citados recae la decisión de dicha transición:

(0,5puntos)

2	a)Un proceso cambia de estado pasando de En Ejecución a Preparado El planificador a corto plazo es el que toma la decisión en cada momento de quien ocupa la CPU. Cuando se trata de planificadores expulsivos puede darse el caso de que a la cola de preparados lleguen procesos más prioritarios que el que esta ejecución con lo que este proceso pasará a la cola de preparados. Con algoritmos de RR, puede darse el caso de que haya procesos en la cola de preparados y al proceso en ejecución se le haya acabado el quantum y todavía necesite más tiempo de CPU, con lo que el proceso en ejecución pasará a la cola de preparados
	b)Un proceso cambia de estado pasando de Preparado a estar almacenado en el área de intercambio o swap del sistema. Los planificadores a medio plazo se encargan de llevar procesos a medio ejecutar al área de swap. Esta decisión se toma cuando el grado de multiprogramación del sistema es elevado y se detectan muchos fallos de página. Al almacenar el proceso en el área de swap, se libera espacio en memoria que puede asignarse al resto de procesos, reduciéndose el número de fallos de página.

3. Dado el siguiente código cuyo archivo ejecutable ha sido generado con el nombre "Ejemplo1".

```
   /*** Ejemplo1***/
 1 #include "todas_las_cabeceras_necesarias.h"
 2
 3 main()
 4 { int i=0,
 5   pid_t pid, pid_x;
 6
 7   for (i=0; i<2; i++)
 8   { pid=fork()
 9     if (pid==0)
10      { sleep(5);
11        pid_x=fork();
12        exit(0);
13      }
14   }
15   sleep(10);
16   while (wait(NULL) =! -1);
17   exit(0);
18 }
```

Indique de forma justificada:

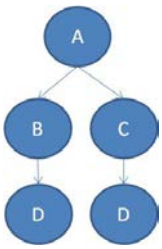
a) El número de procesos que se generan y dibuje el esquema de parentesco entre los procesos creados al ejecutar este código.

b) Si pueden llegar a generarse durante su ejecución procesos **zombies** o **huérfanos**.

(1,0 puntos)

3

El número de procesos creados al ejecutar ejemplo1 son 5, un padre y dos hijos que a su vez crean un hijo cada uno. El esquema sería el siguiente:



Cuando los hijos finalicen el sleep(5) y hayan creado a su vez un hijo (nieto), es probable que el proceso padre se encuentre en el sleep(10), y por lo tanto los hijos permanezca durante un intervalo breve de tiempo en estado zombie.

Tal y como está estructurado el código, sólo en el caso de algún error o la llegada de una señal que afecte al proceso padre (antes de que acaben los procesos hijos) se podría dar la circunstancia de que los procesos hijos se queden huérfanos.

4.Un sistema dispone de un planificador a corto plazo con un algoritmo Round Robin con **q=2 ut**. El sistema está dotado de un único dispositivo de E/S gestionado con FCFS. El orden de llegada de los procesos a las colas es el siguiente: en primer lugar los nuevos, a continuación los procedentes de E/S y por último los que salen de CPU. A dicho sistema llegan 3 procesos cuyo perfil de ejecución e instante de llegada son:

(1.5=0,75+0.5+0.25 puntos)

Ejercicio de Evaluación Final
23 de Enero de 2015

Proceso	Perfil de ejecución	Instante de llegada
A1	3 CPU + 1 E/S + 1 CPU	0
A2	3 CPU + 1 E/S + 1 CPU	1
B	1 CPU + 3 E/S + 1 CPU	2

a)Represente mediante el diagrama temporal la ocupación de la CPU, del periférico de E/S y de las diferentes colas en cada unidad de tiempo. (0,75 puntos)

T	Preparados	CPU	Cola E/S	Disp. E/S	Evento
0		A1(2)			Llega A1
1	A2	A1(1)			Llega A2
2	A1, B	A2(2)			Llega B
3	A1, B	A2(1)			
4	A2, A1	B(0)			
5	A2	A1(0)		B	
6		A2(0)	A1	B	
7			A2, A1	B	
8		B(0)	A2	A1	
9		A1(0)		A2	Termina B
10		A2(0)			Termina A1
11					Termina A2
12					

b)Obtenga el tiempo medio de espera, tiempo medio de retorno y el índice de ocupación de CPU.
(0,5 puntos)

4b

Tiempo medio de espera :

Ocupación de CPU:

Tiempo medio de retorno: **((10-0)+(11-1)+(9-2)) / 3 = (10+10+7) / 3 = 27 / 3 = 9**

c)Calcule el **tiempo medio de respuesta** en este sistema con q=2ut (apartado anterior), e indique cual sería el valor del tiempo de medio de respuestas con q=1ut. Demuestre que con q=1ut se reduce dicho tiempo respecto al obtenido con q=2 ut, para ello puede rehacer parte del diagrama de ocupación de CPU.
(0,25 puntos)

4c

Tiempo medio de respuesta (q=2): **((0-0)+(2-1)+(4-2)) / 3 = (0+1+2) / 3 = 3 / 3 = 1**

Tiempo medio de respuesta para **q=1**: **((0-0)+(1-1)+(3-1)) / 3 = (0+0+1) / 3 = 1/3 = 0,33**

T	Preparados	CPU	E/S	Cola E/S	Evento
0		A1(2)			Llega A1
1	A1	A2(2)			Llega A2
2	A2, B	A1(1)			Llega B
3	A1, A2	B(0)			
4					
5					

Efectivamente hay una reducción de 1 a 0,33

5.Considere el siguiente programa con hilos y semáforos, que tiene definidas tres constantes N, M y Q.

```
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
#define N ...
#define M ...
#define Q ...

pthread_t T[N];
pthread_attr_t atr;
sem_t S1,S2;
int I;

void *P (void *I)
{
  sem_wait(&S1);
  printf("Thread working\n");
  sem_post(&S2);
}

int main()
{ int i;
  pthread_attr_init(&atr);
  sem_init(&S1,0,M); /* Semáforo S1 se inicializa a M*/
  sem_init(&S2,0,0); /* Semáforo S2 se inicializa a 0*/
  for (i=0;i<N;i++)
     pthread_create(&T[i],&atr,P,NULL);
  sleep(10);
  for (i=0;i<Q;i++)
     sem_wait(&S2);
  printf("Bye\n");
  return(0);
}
```

Describa y justifique el número de hilos que se crean, así como los mensajes que se muestran por la salida estándar al ejecutar dicho código para los siguientes valores de las constantes N, M y Q:

(1,0 puntos)

5

a) N=10, M=4, Q=2

Se muestra por pantalla 4 líneas con el texto "Thread working" y una línea con el texto "Bye".

Se crean diez hilos, pero el semáforo S1 limita a cuatro los que superan la línea **sem_wait(&S1);** todos ellos tienen tiempo suficiente (el intervalo de 10 segundo que permite la línea **sleep(10)** del hilo main) para imprimir el texto citado en la salida estándar.
El hilo main llegará sin problemas a la línea **printf("Bye\n")** cuando el semáforo S2 valga 2

b)N=5, M=2, Q=4

Se muestra por pantalla 2 líneas con el texto "Thread working"; el proceso no acaba.

Se crean cinco hilos, y el semáforo S1 limita a dos los que superan la línea **sem_wait(&S1);** y escribirán su texto. Sin embargo, el hilo main quedará bloqueado por el semáforo S2 en la tercera iteración de **for (i=0;i<Q;i++) sem_wait(&S2)** y no llegará a la línea **printf("Bye\n")**.

Puntuación (conjunta para ambos apartados):
0.2 si tienen claro cuántos hilos se crean: N + main
+0.4 si tienen claro que sólo escribirán M veces "Thread Working"
+0.4 si observan que el programa sólo imprime "Bye" y termina si Q≥M

6.Dado tres hilos HILO1, HILO2 e HILO3, que se ejecutan concurrentemente, utilice semáforos con la notación propuesta por Dijkstra (P y V para las operaciones) para garantizar que los diferentes hilos ejecuten las funciones cuyo nombre empieza por "secuenciax" según el orden que sugieren el número x empleado en sus nombres. Si hay varias funciones con el mismo número, ninguna de ellas debe empezar antes de que termine la función con el número anterior y todas ellas deben haber terminado antes de que empiece la función con el nombre siguiente. Además, debe asegurarse que las funciones "sc()" se ejecuten en exclusión mutua. Indique qué valor inicial deberán tener los semáforos que haya utilizado.

(0.75 punto)

6

Declare e inicialice los semáforos

mutex=1, S2 =0, S3=0, S4=0

Código de HILO 1	Código de HILO 2	Código de HILO 3
secuencia1(); *V(s2);* *P(mutex);* **sc();** *V(mutex);* *P(s3);* **secuencia3();** *V(S4);*	*P(mutex)* **sc();** *V(mutex)* **secuencia1();** *V(S2);* *P(mutex);* **sc();** *V(mutex);* *P(S3);* **secuencia3();** *V(S4);*	*P(mutex)* **sc();** *V(mutex)* *P(s2); P(S2);* **secuencia2();** *V(S3); v(S3);* *P(S4); P(S4);* **secuencia4();**

7. Indique de forma justificada que puede estar ocurriendo en un sistema con Memoria Virtual, multiptogramado y multiusuario en cada una de las siguientes situaciones:
a) El procesador del computador tiene una tasa de utilización del 15% y el dispositivo de paginación (disco) esta ocupado el 97% del tiempo, ¿qué indican estos valores? ¿Cómo podría mejorarse la tasa de utilización del procesador?
b) El procesador del computador tiene una tasa de utilización del 15% y el dispositivo de paginación (disco) esta ocupado el 15% del tiempo, ¿qué indican estos valores? ¿Cómo podría mejorarse la tasa de utilización del procesador?

(0.75 punto)

7

a)
Una tasa de utilización baja del procesador y muy alta del dispositivo de paginación indica que el sistema está en un estado de hiperpaginación (thrasing). En este estado los procesos pasan la mayor parte del tiempo bloqueados esperando a que se les sirva una página que ha causado fallo de pagina desde el dispositivo de paginación. El dispositivo de paginación esta saturado de solicitudes. Esta situación se debe a que hay un grado de multiprogramación muy alto que impide que los procesos puedan tener en memoria suficientes páginas para trabajar. Desde el sistema operativo, una solución consiste en disminuir el grado de multiprogramación, para ello, se pueden suspender procesos llevándolos al área de swap del disco procesos y liberando la memoria que ocupaban. Esta memoria puede asignarse a los procesos en ejecución.
Otra opción puede ser aumentar la memoria principal del equipo.

a)
Si tanto el procesador como el dispositivo de paginación tienen una tasa baja de uso, esto indica que el sistema tiene poca carga o poco trabajo. El sistema puede aceptar mas procesos (aumentar el grado de multiprogramación) y atender a más usuarios para aprovechar las prestaciones del equipo. ejecutarse

8. Sea un sistema con gestión de memoria mediante dos niveles de paginación. La memoria principal es de 1MB y la capacidad de direccionamiento de cada proceso es de 16MB. Las tablas de primer nivel contienen 16 descriptores y el tamaño de página es de 4KB. Responda a las siguientes cuestiones:

(0.75 punto)

8

a)aFormato de la dirección lógica y de la dirección física exponiendo sus campos y números de bits.

Memoria Principal de 1MB -> 20 bits para la Dirección Física (DF)
Capacidad de direccionamiento lógico de -> 16MB -> 24 bits para la dirección Lógica (DL)
Tamaño de Página de 4KB -> 12 bits para desplazamiento de página (DP) o de marco

(DM) Formato de la direc
Desplazamiento
Formato de la dirección Lógica: 24 bits = 4 bits Páginas 1er nivel + 8 bits Páginas 2º nivel + 12 bits desplazamiento de página.

b)Calcule la fragmentación interna que podría generarse al ubicar en memoria los procesos P1, P2 y P3 cuyos tamaños son; 12524 Bytes, 20480 Bytes y 12289 Bytes, respectivamente.

Para calcular la fragmentación interna, calculamos primero la ocupación de la última págima (que es el resto de división del tamaño del proceso y el tamaño de página) y este valor lo restamos al tamaño de pagina. Es decir,
P1 4K- (12524 mod 4K) = 3860;
P2 3975;
P3 = 0;
P4 = 4095

9. Sea un sistema con Memoria Virtual, paginación por demanda con reemplazo LOCAL y tamaño de página de 64KBytes. En dicho sistema se están ejecutando los procesos A y B cuyo espacio lógico ocupa 5 páginas y cuyas tablas de páginas (Bit ref. es el bit de referencia y los tiempos están en decimal) en el instante t=20 contienen:

Tabla de páginas del proceso A				
Marco	Bit de validez	Bit Ref.	Instante último acceso	Instante carga en Memoria
-	i	1	10	1
-	i	1	8	2
0x002	v	0	17	9
0x001	v	1	20	8
0x000	v	1	15	4

Tabla de páginas del proceso A				
Marco	Bit de validez	Bit Ref.	Instante último acceso	Instante carga en Memoria
0x004	v	0	18	14
-	i	1	9	6
-	i	1	10	10
0x003	v	1	19	12
-	i	1	11	11

Suponga que en los instantes t=21 y t=22 la CPU emite las dirección lógicas A:0x00120AB y B:0x00431CB.
(1,25 puntos= 0,25+ 0, 5+0,5)

9

a)Utilice la información de las tablas de páginas e indique para el instante t=20, cuál es la dirección física que corresponde a la dirección lógica A:0x004247A

Las páginas son de 64KBytes = 2^{16} → son necesarios 16 bits para el desplazamiento de página o 4 dígitos en hexadecimal

La dirección lógica A:0x004247A, corresponde a la página 4 con desplazamiento 247A → Según la tabla de páginas del proceso A, la página 4 se encuentra en el marco 0x000 → dirección física 0x000247A

b)Considere los datos de t=20 e indique como quedarían los contenidos de las tablas de páginas después de emitirse las direcciones lógicas de t=21 y t=22, utilizando un algoritmo LRU con reemplazo LOCAL.

La dirección lógica A:0x00120AB corresponde a la página 1 del proceso A y B:0x00431CB corresponde a la página 4 del proceso B. Dado que el LRU con reemplazo LOCAL designa como victima aquella página del proceso que hace mas tiempo que no ha sido referenciada, en este caso, tendremos que para el proceso A la victima será la página 4, mientras que para el proceso B la victima será la página 0.

Tabla de páginas del proceso A				
Marco	Bit de validez	Bit Ref.	Instante último acceso	Instante carga en Memoria
-	i	1	10	1
0x000	v	1	21	21
0x002	v	0	17	9
0x001	v	1	20	8
-	i	1	15	4

Tabla de páginas del proceso A				
Marco	Bit de validez	Bit Ref.	Instante último acceso	Instante carga en Memoria
-	i	0	18	14
-	i	1	9	6
-	i	1	10	10
0x003	v	1	19	12
0x004	v	1	22	22

c)Considere los datos de t=20 e indique como quedarían los contenidos de las tablas de páginas después de emitirse las direcciones lógicas de t=21 y t=22, utilizando un algoritmo 2ª Oportunidad con reemplazo LOCAL

La dirección lógica A:0x00120AB corresponde a la página 1 del proceso A y B:0x00431CB corresponde a la página 4 del proceso B. Dado que el 2ª con reemplazo LOCAL designa como victima aquella página del proceso que tiene su bit de referencia a 0, en este caso, tendremos que para el proceso A la victima será la página 2, mientras que para el proceso B la victima será la página 0.

Tabla de páginas del proceso A				
Marco	Bit de validez	Bit Ref.	Instante último acceso	Instante carga en Memoria
0x002	v	1	21	21
-	i	1	8	2
-	i	0	17	9
0x001	v	1	20	8
0x000	v	1	15	4

Tabla de páginas del proceso A				
Marco	Bit de validez	Bit Ref.	Instante último acceso	Instante carga en Memoria
-	i	0	18	14
-	i	1	9	6
-	i	1	10	10
0x003	v	1	19	12
0x004	v	1	22	22

10.El siguiente programa ene15.c cuenta el número de líneas que contienen una determinada *"palabra"* en un archivo y escribe en la salida estándar *"El resultado es XX"* donde XX es el número de líneas. El programa trabaja con 3 procesos y utiliza los programas grep texto archivo y wc -l. El formato de la orden es: `$ ./ene15 palabra archive`

```
#include <"todos los includes necesarios">
int main(int argc, char *argv[]) {
int fd1[2], fd2[2];
char tira[1000];
int n;
    pipe(fd1);
    if (fork()==0){ // ********** proceso 2 *************
       dup2 (fd1[1],STDOUT_FILENO);
       close (fd1[0]);   close (fd1[1]);
//TABLA 2
       execlp("grep","grep",argv[1],argv[2],NULL);
    }else{
       close (fd1[1]);
       pipe(fd2);
       if (fork()==0){ // ********** proceso 3 *************
          dup2 (fd1[0],STDIN_FILENO);
          close (fd1[0]);
          dup2 (fd2[1],STDOUT_FILENO);
          close (fd2[0]);  close (fd2[1]);
//TABLA 3
          execlp("wc","wc","-l",NULL);
       }else{ // ********** proceso 1 **************
//TABLA 1
          close(_________);   close(_________);
          n=read(_________,tira,1000);
          tira[n]=0;
          printf("El resultado es %s",tira);
          close (fd2[0]);
       }
    }
    return 0;
}
```

(1,0 puntos= 0,3+0,4+0,3)

10

a) Indique el esquema de comunicación de los procesos que se crean

proceso 2 | proceso 3 | proceso 1 (el proceso 1 accede al tubo sin redirección)

b) Indique el contenido de las tablas de descriptores en los puntos que se indica en el programa

	TABLA 1
0	/dev/tty
1	/dev/tty
2	/dev/tty
3	tubo1_0
4	tubo2_0
5	tubo2_1
6	

	TABLA 2
0	/dev/tty
1	tubo1_1
2	/dev/tty
3	
4	
5	
6	

	TABLA 3
0	tubo1_0
1	tubo2_1
2	/dev/tty
3	
4	
5	
6	

c) Complete los descriptors que faltan en el código mostrado

```
        close(fd1[0]);   close(fd2[1]);
        n=read(fd2[0],tira,1000);
```

Criterios: 1 pto (0,3 + 0,3 + 0,4)
A) Bien/mal
B) 0,2+0,1+0,10,
C) 1 por descriptor

---

11. Un disco con una capacidad de 512MBytes, se formatea con una versión de MINIX y la siguiente estructura:

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

Los tamaños y valores usados en el formateo son:

- Nodo-i de 64Bytes con: 7 punteros directos, 1 indirecto y 1 doble indirecto
- Puntero a zona de 32 bits = 4 Bytes
- Entradas de directorio de 16 Bytes.
- 1 Bloque = 1KBytes
- 1 zona = 2^2 bloques= 4KBytes
- 4096 nodos-i

Se pide:

a) Indique de forma detallada el número de bloques que ocupa el Mapa de bits nodos-i, el Mapa de bits Zonas y los Nodos-i.
b) Indique el número de nodos-i diferentes que necesitará consultar para acceder al Byte 512 del archivo "/DirX/ArchX"

(1,0 puntos= 0, 75+0,25)

11

a)

Mapa de bits de nodos-i:
Para 4096 nodos-i, se necesita un mapa de bits con 4096 bits como mínimo. Como los bloque son de 1Kbyte → 4096 bits/1Kbyte= 4*1024 bits/ 1024 Byte= 4*1024 bits/8*1024 bits = ½=1 bloque

Mapa de bits de zonas:
Disco de 512 MBytes= 2^9*2^{20}Bytes → dividiendo por el tamaño de zona que es de 4KBytes tenemos →2^9*2^{20}Bytes/2^2*2^{10}Bytes= 2^7*2^{10}Zonas= 2^{17} Zonas =131072Zonas.
Se necesita 1 bit por cada zona → 2^{17} bit /1bloque= 2^7*2^{10} bit/ 2^{10} Byte= 2^7*2^{10} bit/ 2^3 *2^{10} bit =2^4 = 16 bloques

Nodos-i: Los nodos-i son de 64 byte y se requiere 4096 nodos-i, serán necesarios 4096 * 64 Bytes =2^2 2^{10} 2^6=2^{18} bytes → 2^{18} Bytes /2^{10} Bytes = 2^8 = 256 bloques.

b)

Para acceder al archivo "/DirX/ArchX" será necesario consultar un total de 3nodos-i diferentes que corresponden a los archivos:
/, /DirX, /DirX/ArchX