
 <p>Universidad Carlos III de Madrid</p>	<p>Departamento de Informática Grado en Ingeniería Informática Sistemas Operativos Parte 1 Examen de la convocatoria ordinaria 19 de enero de 2012</p>	
---	--	---

ATENCIÓN:

- Lea atentamente todo el enunciado antes de comenzar a contestar.
 - Dispone de 1,5 horas para realizar la prueba.
 - No se podrán utilizar libros ni apuntes, ni calculadoras de ningún tipo.
 - Los teléfonos móviles deberán permanecer desconectados durante la prueba (apagados, no silenciados).
 - Solamente se corregirán los ejercicios contestados con bolígrafo. Por favor no utilice lápiz.
-

Teoría [4 puntos]:

Pregunta 1.

¿Qué ocurre cuando un proceso termina y su padre aún no ha realizado el wait?
Razone la respuesta.

Pregunta 2.

Dado el siguiente código:

```
for (i=1; i<4; i++){
    pid=fork();
    if (pid==0){ //Hijo
        printf ("Hijo %d, pid =%d, ppid%d\n",i, getpid(),getppid());
    }
    else
        printf ("El padre ha creado %d hijos \n",i);
} // fin for
```

Muestre los mensajes que aparecen por pantalla. Suponga que el pid del padre es 700 y los de los hijos generados serán 701, 702 .

Pregunta 3.

Explique brevemente qué es un cambio de contexto. Enumere tres situaciones en las que se puede producir un cambio de contexto.



Pregunta 4.

Para la valoración de los distintos métodos de planificación de procesos se utilizan distintas medidas. Explica en qué consisten las siguientes medidas:

Utilización de CPU.

Tiempo de retorno (T_q)

Tiempo de servicio (T_s):

 <p>Universidad Carlos III de Madrid</p>	<p>Departamento de Informática Grado en Ingeniería Informática Sistemas Operativos Parte 1 Examen de la convocatoria ordinaria 19 de enero de 2012</p>	
---	--	---

Ejercicio 1 [3puntos]:

Realizar una aplicación que realice lo siguiente.

El padre

- ❑ Creará 2 hijos,
- ❑ Esperará un número aleatorio de segundos entre 1 y 10 y enviará a ambos hijos la señal SIGUSR1.
- ❑ Escribirá FIN en pantalla cuando ambos hijos hayan finalizado.

Cada uno de los hijos deberá

1. Esperar a que el padre le envíe la señal,
2. Mientras esperan escribirán su pid en pantalla.
3. Cuando reciban la señal escribirán el pid del hermano en pantalla y finalizarán

Se aconseja el uso de una tubería para la comunicación de la información

Ejercicio 2 [3puntos]:

Disponemos de un planificador de un sistema UNIX para la ejecución de procesos, el planificador funciona con colas multinivel con prioridades. Cada cola utiliza una política Round-Robin. En un instante determinado el estado en el que se encuentra el sistema es el siguiente:

Prioridad máxima

Proceso A: 250 instrucciones + 50 E/S + 300 instrucciones

Proceso B: 500 instrucciones + 200 E/S + 100 instrucciones

Prioridad media:

Proceso C: 750 instrucciones

Proceso D: 150 instrucciones + 300 E/S + 50 instrucciones



Prioridad mínima

Proceso E: 1500 instrucciones

Se pide:

Indicar la secuencia de terminación de los procesos (empezando en $t=0$), así como los cambios de contexto que se producen a partir de este momento, en los siguientes casos:

- a) Si entre cada cambio de contexto forzado por el planificador a corto plazo mediante las interrupciones del reloj da tiempo a ejecutar 500 instrucciones.
- b) Lo mismo si únicamente da tiempo a ejecutar 200 instrucciones.

 <p>Universidad Carlos III de Madrid</p>	<p>Departamento de Informática Grado en Ingeniería Informática Sistemas Operativos Parte 1 Examen de la convocatoria ordinaria 19 de enero de 2012</p>	
---	--	---

- c) Calcular los porcentajes de CPU útil en el supuesto a), si cada cambio de contexto necesita 50 instrucciones del SSOO.
- d) Calcular ese mismo porcentaje para el apartado b) y con cambios de contexto de 100 instrucciones del SSOO.

Nota:

$\text{CPU util} = \text{cociente instrucciones_de_aplicaciones} / \text{instrucciones_totales} * 100$.

Se considera que si un proceso permanece en ejecución una vez terminado su cuanto no se produce pérdida por cambio de contexto.



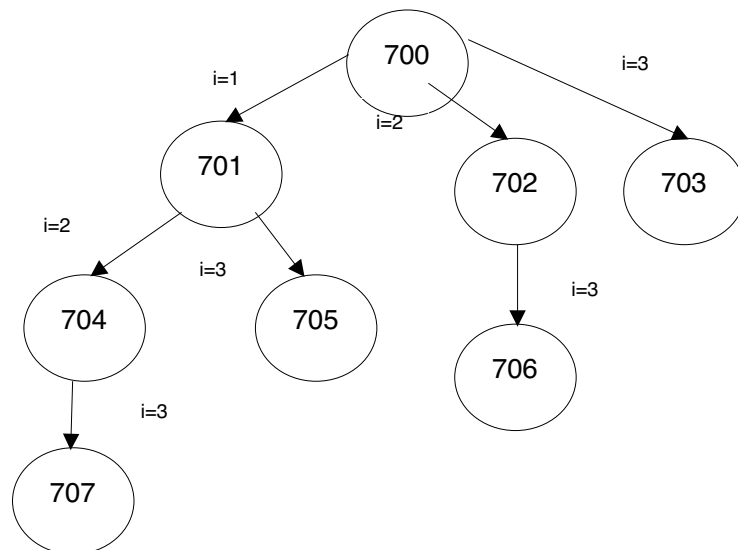
SOLUCIONES

TEORIA 1.

El proceso no puede liberar totalmente la zona de memoria que tiene asignada porque, entre otras cosas, tiene que almacenarse el valor que pasa al padre al ejecutar el exit (que el padre recibe con el wait)

TEORÍA 2.

El padre ha creado 1 hijos
Hijo 1 pid=701 ppid=700
El padre ha creado 2 hijos
Hijo 2 pid=702 ppid=700
El padre ha creado 3 hijos
Hijo 3 pid=703 ppid=700
El padre ha creado 2 hijos
Hijo 2 pid=704 ppid=701
El padre ha creado 3 hijos
Hijo 3 pid=705 ppid=701
El padre ha creado 3 hijos
Hijo 3 pid=706 ppid=702
El padre ha creado 3 hijos
Hijo 3 pid=707 ppid=704



TEORÍA 3.

Un cambio de contexto son todas las operaciones que realiza el sistema operativo para que se realice el cambio del programa que se encuentra en ejecución, es decir, descarga de los datos del proceso que se encuentra actualmente en ejecución y carga de los datos correspondientes al nuevo proceso que ha de ejecutarse.

Se produce un cambio de contexto cuando:

- Cuando el proceso que está ejecutándose termina.
- Cuando el proceso que está ejecutándose solicita una E/S y queda bloqueado.
- Cuando se acaba el tiempo de ejecución asignado a un proceso y ha de cargarse un nuevo proceso.

TEORÍA 4.

Utilización de CPU:

o Porcentaje de tiempo que se usa la CPU. o Objetivo: Maximizar.

Tiempo de retorno (Tq)

o Tiempo que está un proceso en el sistema. Instante final (Tf) menos instante inicial (Ti). o Objetivo: Minimizar.

Tiempo de servicio (Ts):

o Tiempo dedicado a tareas productivas (cpu, entrada/salida). $T_s = T_{CPU} + T_{E/S}$



EJERCICIO 1.

```
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>
#include <sys/types.h>

int esperar=1;
void finespera(int s) { esperar=0; }

main () {
    int fd[2],pid1,pid2,p,segEspera;

    /* Preparo la ejecución de finespera cuando llegue SIGUSR1*/
    struct sigaction sa;
    sa.sa_handler=finespera;
    sa.sa_flags=0;
    sigemptyset (&(sa.sa_mask));
    sigaction (SIGUSR1, &sa, NULL);

    pipe (fd); //Creación de la tubería

    pid1=fork(); //Creo el 1º hijo
    if (pid1 == 0){
        close (fd[1]);
        //Leo de la tubería el pid del hermano
        read ( fd[0], &p, sizeof (int));
        //Espero la llegada de SIGUSR1 y cambie la variable esperar
        while (esperar){
            printf ("Mi pid es %d\n", getpid());
            sleep (1); //Escribo 1 vez por seg
        }
        printf ("Mi pid=%d pid hermano=%d\n", getpid(), p);
        close (fd[0]);
        exit (0);
    }
    else {
        pid2=fork(); //Creo el 2º hijo
        if (pid2 == 0){
            close (fd[0]);
            p=getpid();
            write ( fd[1], &p, sizeof (int)); //Envío mi pid a mi hermano
            //Espero la llegada de SIGUSR1 y cambie la variable esperar
            while (esperar){
                printf ("Mi pid es %d\n", getpid());
                sleep (1); //Escribo 1 vez por
            }
            //Soy el 2º hijo. El pid de mi hermano me lo da el padre en pid1
            printf ("Mi pid=%d pid hermano=%d\n", getpid(), pid1);
            close (fd[1]);
            exit (0);
        }
        else {
            close (fd[0]);
            close (fd[1]);
            srandom (getpid());
        }
    }
}
```



Universidad
Carlos III de Madrid

Departamento de Informática
Grado en Ingeniería Informática
Sistemas Operativos
Parte 1
Examen de la convocatoria ordinaria
19 de enero de 2012



```
segEspera=1+random()%10;//Genero n° aleatorio entre 1 y 10
printf ("Esperaré %d seg\n",segEspera);
sleep(segEspera);
kill (pid1, SIGUSR1); //Envío señales a hijos
kill (pid2, SIGUSR1);
waitpid ( pid1, NULL, 0); //Espero finalización de hijos
waitpid ( pid2, NULL, 0);
printf ("FIN de ambos hijos\n");
}
}
}
```

EJERCICIO 2.

a) Round-Robin 500 instrucciones

Tiempo	Proceso en ejecución	Motivo de expulsión de ejecución
de 0 a 250	A	expulsado por E/S
de 250 a 750	B	expulsado por E/S
de 750 a 1050	A	expulsado por finalización
de 1050 a 1150	B	expulsado por finalización
de 1150 a 1650	C	expulsado por fin de cuanto
de 1650 a 1800	D	expulsado por E/S
de 1800 a 2050	C	expulsado por finalización
de 2050 a 2550	E	expulsado por fin de cuanto
de 2550 a 2600	D	expulsado por finalización
de 2600 a 3600	E	expulsado por finalización

b) Round-Robin 200 instrucciones

Tiempo	Proceso en ejecución	Motivo de expulsión de ejecución
de 0 a 200	A	expulsado por fin de cuanto
de 200 a 400	B	expulsado por fin de cuanto
de 400 a 450	A	expulsado por E/S
de 450 a 650	B	expulsado por fin de cuanto
de 650 a 850	A	expulsado por fin de cuanto
de 850 a 950	B	expulsado por E/S
de 950 a 1050	A	expulsado por finalización
de 1050 a 1250	C	expulsado por fin de cuanto
de 1250 a 1350	B	expulsado por finalización
de 1350 a 1500	D	expulsado por E/S
de 1500 a 1700	C	expulsado por fin de cuanto
de 1700 a 1900	E	expulsado por fin de cuanto
de 1900 a 2100	C	expulsado por fin de cuanto
de 2100 a 2150	D	expulsado por finalización
de 2150 a 2300	C	expulsado por finalización
de 2300 a 3600	E	expulsado por finalización

c) Número de cambios de contexto = 9

$$\text{CPU util} = (3600 / (3600 + 9 \cdot 50)) \cdot 100 = 88,88 \%$$

d) Número de cambios de contexto = 15

$$\text{CPU util} = (3600 / (3600 + 15 \cdot 100)) \cdot 100 = 70,58 \%$$