
 <p>Universidad Carlos III de Madrid</p>	<p>Departamento de Informática Grado en Ingeniería Informática Sistemas Operativos</p> <p>Prueba de Evaluación Continua 22 de octubre de 2009</p>	
---	---	---

**ATENCIÓN:**

- Lea atentamente todo el enunciado antes de comenzar a contestar.
- Dispone de 90 minutos 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.

---

**APELLIDOS:**

**NOMBRE:**



**NIA:**

Ejercicio 1 [0,5 puntos]: ¿Qué elementos debe tener un procesador para que se pueda utilizar planificación apropiativa?

Ejercicio 2 [0,5 puntos]: Indique una ventaja del uso de bibliotecas dinámicas.

Ejercicio 3 [0,5 puntos]: ¿En qué nivel de planificación (corto plazo, medio plazo o largo plazo) se gestiona la expulsión de procesos al área de swap?



Ejercicio 4 [0,5 puntos]: Enumere las dos razones para colocar un elemento de información fuera del BCP (bloque de control de procesos) de un determinado proceso.

 <p>Universidad Carlos III de Madrid</p>	<p>Departamento de Informática Grado en Ingeniería Informática Sistemas Operativos</p> <p>Prueba de Evaluación Continua 22 de octubre de 2009</p>	
---	---	---

Ejercicio 5 [1 puntos]: Dibuje el diagrama de estados del ciclo básico de un proceso, para el caso de un único procesador, incluyendo exclusivamente los estados que afectan a la planificación a corto plazo.

Ejercicio 6 [0,5 puntos]: Indique un ejemplo de caso en que se produce un cambio de contexto voluntario.

Ejercicio 7 [0,5 puntos]: Si un programa que utiliza hilos no realiza llamadas al sistema bloqueantes, ¿qué es mejor usar hilos de usuario (ULT) o hilos de kernel (KLT)?

 <p>Universidad Carlos III de Madrid</p>	<p>Departamento de Informática Grado en Ingeniería Informática Sistemas Operativos</p> <p>Prueba de Evaluación Continua 22 de octubre de 2009</p>	
---	---	---

Ejercicio 8 [3 puntos]: Dado el siguiente código, responder a las preguntas que aparecen a continuación:

```
#include <stdlib.h>
#include <stdio.h>
#include <signal.h>
#define MAX 1
int espera = 10;
void controlador ();
int main (int contargs, char *args[]){
    pid_t pid;
    int i=0;
    signal (SIGCHLD, controlador);

    for (i=0;i<MAX;i++){
        pid = fork ();
        if (pid == 0){
            while(1){
                sleep (1);
                printf("Soy %d \n", i);
            }
            exit(0);
        }
    }



    sleep (espera);
    signal (SIGCHLD, SIG_IGN);
    kill (pid, SIGINT);
    exit (0);
}

void controlador (){
    int id, est;
    id = wait (&est);
    exit (0);
}
```

- Describa el funcionamiento del programa
- ¿Qué pasaría si MAX tuviera un valor de 10?
- Modifique el programa para que funcione para cualquier valor de MAX.

a) El proceso padre crea un hijo, y después de 10 segundos le manda una señal para que termine su ejecución. Si se tiene en cuenta `signal (SIGCHLD, SIG_IGN);`, el programa padre termina. En caso contrario, cuando el hijo muera, se ejecutará el código de la función controlador.

b) Si MAX es igual a 10, se crearán 10 procesos hijos de los que solo 1 terminará.

 <p>Universidad Carlos III de Madrid</p>	<p>Departamento de Informática Grado en Ingeniería Informática Sistemas Operativos</p> <p>Prueba de Evaluación Continua 22 de octubre de 2009</p>	
---	---	---

c) Se proponen 2 soluciones:

<pre>#include &lt;stdlib.h&gt; #include &lt;stdio.h&gt; #include &lt;signal.h&gt; #define MAX 100 int espera = 10; pid_t pid[MAX];  void controlador (); int main (int contargs, char *args[]){     int i=0;     signal (SIGCHLD, controlador);     for (i=0;i&lt;MAX;i++){         pid[i] = fork ();         if (pid[i] == 0){             while(1){                 sleep (1);                 printf("Soy %d \n", i);             }             exit(0);         }     }     sleep (espera);     signal (SIGCHLD, SIG_IGN);     for (i=0;i&lt;MAX;i++){         kill (pid[i], SIGINT);     }     exit (0); }  void controlador (){     int id, est, i;     for (i=0;i&lt;MAX;i++){         id = wait (&amp;est);     }     exit (0); }</pre>	<pre>#include &lt;stdlib.h&gt; #include &lt;stdio.h&gt; #include &lt;signal.h&gt; #define MAX 100 int espera = 10; void controlador ();  int main (int contargs, char *args[]){     pid_t pid;     int i=0;     signal (SIGCHLD, controlador);     for (i=0;i&lt;MAX;i++){         pid = fork ();         if (pid == 0){             while(1){                 sleep (1);                 printf("Soy %d \n", i);             }             exit(0);         }         sleep (espera);         signal (SIGCHLD, SIG_IGN);         kill (pid, SIGINT);     }     exit (0); }  void controlador (){     int id, est;     id = wait (&amp;est);     exit (0); }</pre>
---	--





Universidad  
Carlos III de Madrid

Departamento de Informática  
Grado en Ingeniería Informática  
Sistemas Operativos

Prueba de Evaluación Continua  
22 de octubre de 2009



 <p>Universidad Carlos III de Madrid</p>	<p>Departamento de Informática Grado en Ingeniería Informática Sistemas Operativos</p> <p>Prueba de Evaluación Continua 22 de octubre de 2009</p>	
---	---	---

Ejercicio 9 [3 puntos]: En un determinado sistema operativo los procesos se ejecutan con planificación apropiativa y política de planificación cíclica (round-robin).

En la siguiente tabla se especifica para cada proceso, su tiempo de llegada y el tiempo que necesitan para ejecutarse. Todos los procesos realizan exclusivamente tareas de cálculo.

Proceso	Tiempo de llegada	Tiempo de ejecución
P1	0	500
P2	100	300
P3	300	400
P4	600	1000
P5	700	600

Se desea evaluar las diferencias que se producirán al variar la longitud de la rodaja de tiempo, considerándose valores de 200 y 500 milisegundos.

Para las dos posibilidades, se pide:

1. Determine el tiempo de finalización de cada proceso.
2. Determine el tiempo que cada proceso ha estado en el sistema (tiempo de retorno).
3. Determine el tiempo de servicio y el tiempo de espera de cada proceso.
4. Determine el tiempo de retorno normalizado
5. Determine el tiempo medio de espera.
6. Determine el tiempo medio de retorno normalizado.

¿Puede concluir algo de los resultados?



Universidad  
Carlos III de Madrid

Departamento de Informática  
Grado en Ingeniería Informática  
Sistemas Operativos

Prueba de Evaluación Continua  
22 de octubre de 2009





Universidad  
Carlos III de Madrid

Departamento de Informática  
Grado en Ingeniería Informática  
Sistemas Operativos

Prueba de Evaluación Continua  
22 de octubre de 2009

