

examen-SCD-Teoria-Temas-1-y-2.pdf



Anónimo



Sistemas Concurrentes y Distribuidos



2º Grado en Ingeniería Informática



Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
Universidad de Granada

Máster

Online en Ciberseguridad

Nº1 en España según El Mundo



**Hasta el 46%
de beca**



Mejor Máster
según el
Ranking de
ELMUNDO

Para ser el mejor hay que aprender
de los mejores.

IMEF

Smart Education

Deloitte

Infórmate

Consigue Empleo o Prácticas

Matricúlate en IMF y accede sin coste a nuestro servicio de Desarrollo Profesional con más de 7.000 ofertas de empleo y prácticas al mes.



SISTEMAS CONCURRENTES Y DISTRIBUIDOS. TEMAS 1 y 2

(23-11-2021)

Tipo B

Apellidos + Nombre:

1.- (3 ptos.) Señalar la única opción correcta mediante un círculo. En caso de señalar una equivocadamente, marcar con una X la opción errónea (un fallo resta 1/3 de un acierto):

- Sobre paralelismo y concurrencia:
 - a) El paralelismo real solo se puede dar en sistemas distribuidos
 - b) En un sistema monoprocesador no puede implementarse ningún tipo de concurrencia
 - c) El paralelismo real solo se puede dar si los procesos comparten una memoria común
 - d) Todas las anteriores afirmaciones son falsas
- Considera la sentencia $x = x + 1$, que se ejecuta repetidas veces por varios procesos concurrentes:
 - a) Cada vez que se ejecuta por cualquier proceso el tiempo que tarda podría ser distinto
 - b) En sistemas monoprocesador, siempre tarda lo mismo en ejecutarse por cualquiera de los procesos
 - c) En sistemas multiprocesador, siempre tarda lo mismo en ejecutarse por cualquiera de los procesos
 - d) Siempre tarda lo mismo en ejecutarse por cualquiera de los procesos, ya que es atómica
- Dado un programa concurrente compuesto por dos programas secuenciales A y B que se ejecutan concurrentemente, donde el programa A se define como la secuencia de sentencias atómicas $A1 \rightarrow A2 \rightarrow A3$ y B como $B1 \rightarrow B2 \rightarrow B3$, se cumple que:
 - a) La secuencia de interfoliación $B2 A1 A2 A3 B1 B3$ es válida
 - b) La secuencia de interfoliación $B1 A1 A2 B3 A3 B2$ es válida
 - c) La secuencia de interfoliación $A1 A2 B1 B2 A3 B3$ es válida
 - d) Todas las anteriores son falsas
- Respecto a los semáforos, señalar la opción correcta:
 - a) Inicialmente, la cola de procesos del semáforo está vacía
 - b) En la cola de procesos del semáforo están todos los procesos que han ejecutado un `sem_wait`
 - c) El valor del semáforo puede ser cualquier número entero
 - d) La operación `sem_wait` debe ser atómica, pero no es necesario que lo sea la operación `sem_signal`
- Supongamos un algoritmo de exclusión mutua (para 2 procesos) que cumple todas las propiedades excepto la de *espera limitada*, siendo ambos procesos bucles infinitos (PE+SC+RS). Entonces:
 - a) Ambos procesos pueden permanecer en PE indefinidamente
 - b) Puede ocurrir que un proceso permanezca en PE mientras el otro accede a SC varias veces
 - c) Un proceso podría quedar en PE indefinidamente mientras el otro está en RS
 - d) Los dos procesos podrían estar a la vez en SC
- El siguiente algoritmo de exclusión mutua para 2 procesos:

<code>bool psc[2] := {falso, falso};</code>	
{ Proceso 0 } Mientras verdadero hacer 1. <code>psc[0] := verdadero;</code> 2. mientras <code>psc[1]</code> hacer nada; 3. { Sección crítica } 4. <code>psc[0] := falso;</code> 5. { RS }	{ Proceso 1 } Mientras verdadero hacer 1. <code>psc[1] := verdadero;</code> 2. mientras <code>psc[0]</code> hacer nada; 3. { Sección crítica } 4. <code>psc[1] := falso;</code> 5. { RS }

- Si en el anterior algoritmo permutamos el orden de las líneas 1 y 2:
 - a) No cumple la propiedad de progreso en la ejecución
 - b) No cumple la propiedad de exclusión mutua
 - c) Puede producir interbloqueo
 - d) Todas las afirmaciones anteriores son falsas.
- En un monitor con semántica señalar y salir y con 2 variables condición declaradas:
 - a) El número total de colas de procesos que gestiona el monitor es 4
 - b) El número total de colas de procesos que gestiona el monitor es 5
 - c) El número total de colas de procesos que gestiona el monitor es 3
 - d) El número total de colas de procesos que gestiona el monitor es 2

¿Quieres conocer todos los servicios?



















- En un monitor ya inicializado, compartido por varios procesos de un mismo programa concurrente:
 - a) La cola del monitor puede estar vacía o tener como máximo un proceso
 - b) Si hay un proceso dentro del monitor, todos los demás procesos del programa concurrente estarán esperando en la cola del monitor
 - c) Si la cola del monitor no está vacía, significa que hay un proceso dentro del monitor
 - d) Si la cola del monitor está vacía, significa que hay un proceso dentro del monitor
- En un monitor que sigue la semántica Señalar y Salir, el proceso que se desbloquea al hacer un signal sobre una variable condición:
 - a) Se escoge aleatoriamente entre los procesos bloqueados en dicha variable condición
 - b) Se elige el primer proceso que estuviera esperando para acceder al monitor
 - c) Ninguna de las respuestas es correcta
 - d) Se elige el primer proceso que se bloqueó entre los que haya en la cola de dicha variable condición

2.- (1.5 pts.) Un proceso productor y tres procesos consumidores (Consumidor_i, $i=0,1,2$) se sincronizan de tal forma que cada dato producido por el productor debe ser consumido por los tres consumidores. El productor no podrá producir un nuevo dato hasta que los tres consumidores hayan consumido el anterior y los consumidores no podrán consumir hasta que el productor produzca un nuevo dato. Se supone que dato es una variable compartida por los 4 procesos que tiene tipo entero. Asegurar la sincronización requerida por lo procesos mediante el uso de semáforos:

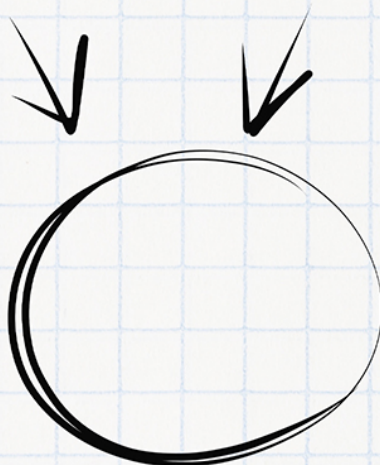
{Declaración e inicialización de variables globales y semáforos}	
int dato;	
Productor	Consumidor(i) $i=0,1,2$
While (true)	While (true)
begin	begin
Produce (dato);	Consume (dato);
end	end

Imagínate aprobando el examen

Necesitas tiempo y concentración

Planes	 PLAN TURBO	 PLAN PRO	 PLAN PRO+
 Descargas sin publi al mes	10 	40 	80 
 Elimina el video entre descargas			
 Descarga carpetas			
 Descarga archivos grandes			
 Visualiza apuntes online sin publi			
 Elimina toda la publi web			
 Precios Anual <input type="checkbox"/>	0,99 € / mes	3,99 € / mes	7,99 € / mes

Ahora que puedes conseguirlo,
¿Qué nota vas a sacar?



WUOLAH

Sistemas Concurrentes y Dist...



Comparte estos flyers en tu clase y consigue más dinero y recompensas



Banco de apuntes de la

WUOLAH

- 1** Imprime esta hoja
- 2** Recorta por la mitad
- 3** Coloca en un lugar visible para que tus compis puedan escanar y acceder a apuntes

- 4** Llévate dinero por cada descarga de los documentos descargados a través de tu QR



3.- (1.5 pts.) Dado el siguiente fragmento de programa concurrente P (programa de la izquierda), donde se consideran a las sentencias encerradas entre < .. > como atómicas:

Programa P	Grafo (apartado 3.1)
<pre> int x,y,z; Begin x:=3 ; y:= 2; z:= 1; {P0} cobegin begin <x:= x + 1;> {P1} <z:= x * y;> {P2} <y:= x - 1;> {P3} end <x:= y + z;> {P4} coend <print(x,y);> {P5} End </pre>	

3.1.- Construir el grafo de precedencia asociado a P y dibújalo justo a la derecha del programa P (véase arriba).

3.2.- De los fragmentos de programas que aparecen a la derecha, indicar si son o no equivalentes al fragmento P. Para ello, construir su grafo de precedencia justo debajo y compararlo con el obtenido en el apartado 3.1.

a)	b)
<pre> Begin P0; P1; fork P4; fork P2; P3; join P2; join P4; P5; End </pre>	<pre> Begin P0; fork P4; P1; P2; P3; join P4; P5; End </pre>

3.3.- Indicar todas las posibles interfoliaciones de sentencias atómicas que pueden derivarse de la ejecución concurrente de P así como los valores que se imprimen (x e y) en cada caso. Hazlo rellenando las filas que necesites de la siguiente tabla:

Interfoliación	x	y

Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

pierdo espacio



5. (4 ptos.) La conocida atracción "Splash" de un parque acuático tiene un aforo limitado a 30 usuarios, que disfrutan de la atracción durante un tiempo que depende del usuario y después salen por la puerta de salida. No obstante, dicha atracción necesita de un mantenimiento periódico reglamentario realizado por el especialista cada vez que la usan 200 usuarios. Este mantenimiento requiere que la atracción no esté siendo usada (esté vacía), y después del mantenimiento, podrán seguir entrando usuarios a la atracción hasta el siguiente ciclo de mantenimiento. De esta manera, cuando entra el usuario 200 desde el último ciclo de mantenimiento, no se dejará entrar a otros usuarios y, cuando la atracción se vacíe por completo, podrá actuar el especialista de mantenimiento. Una vez el especialista finaliza su trabajo, deberá esperar hasta que otros 200 usuarios hayan pasado por la atracción (y hayan salido de la misma) para poder volver a hacer el mantenimiento. Diseñar el monitor *Splash*, con semántica SU, que resuelva la sincronización requerida para el problema, teniendo en cuenta que dicho monitor se usará por parte de los procesos Usuario y Especialista de acuerdo al siguiente esquema:

Proceso Usuario (i), i=1,...,N	Proceso Especialista
<pre>begin Splash.entrada() ; { Disfrute de la atracción } Splash.salida() ; end</pre>	<pre>begin while true do begin Splash.espera() ; { Trabajos de mantenimiento } Splash.fin_trabajos() ; end end</pre>

Necesito concentración

ali ali ooh
esto con 1 coin me
lo quito yo...

WUOLAH

WUOLAH

Escaneado con CamScanner