

Apellidos: _____ Nombre: _____

D.N.I: _____ Firma: _____

1	2	3	4	5	6	7	8	9	10	11	12	Suma
(5)	(5)	(5)	(5)	(5)	(5)	(5)	(5)	(5)	(5)	(5)	(5)	(30/60)

Lo/as estudiantes **con** evaluación continua deben contestar solamente las primeras **seis** (6) preguntas (Parte I). Tiempo: 1 hora y media.

Lo/as estudiantes **sin** evaluación continua deben contestar todas las **doce** (12) preguntas (Parte I+II+III). Tiempo: 3 horas y media.

Una vez empezado el examen, el/la estudiante acepta que cualquier uso de un dispositivo móvil sin previo aviso al profesor, resulta en un suspenso inmediato del examen con puntuación 0, informe a los órganos competentes de la Universidad de un posible intento de fraude, y posibilidad a un futuro examen en este curso solamente mediante orden escrita del órgano superior.

Parte I

Pregunta 1: [5 Puntos] La *región crítica* es un concepto abstracto útil para la programación concurrente. Este concepto está disponible en el lenguaje de programación Java. Detalla su sintaxis y uso. Razona críticamente sobre posibles ventajas y desventajas, tanto a nivel del propio concepto como herramienta de programación concurrente, como a nivel de implementación en el lenguaje Java.

Pregunta 2: [5 Puntos] Ha aparecido en el mercado un nuevo protocolo de entrada y salida para garantizar la exclusión mutua de una sección crítica involucrando dos procesos. Tienes acceso al código fuente y sabes que todas las instrucciones básicas se ejecutan de forma atómica en tu hardware. ¿Cuáles son las propiedades del protocolo que deberías analizar antes de comprarlo? ¿Entre ellas, cómo verificas si el protocolo es correcto (en el sentido que garantice la exclusión mutua)?

Pregunta 3: [5 Puntos] Detalle tres métodos diferentes como implementar (en Java) un contador concurrente (valores enteras) donde diferentes hilos pueden aumentar el valor del contador por cierta cantidad. Argumenta por qué con una declaración simple del entero como `volatile` no funciona correctamente.

Pregunta 4: [5 Puntos] Describe/implementa con la instrucción hardware CAS un protocolo de entrada y salida a una sección crítica para dos procesos con garantía de espera finita para los procesos participantes.

Pregunta 5: [5 Puntos] ¿En la tercera práctica (simulación simple de un ping-pong), existe en la propuesta de tu solución la posibilidad de un bloqueo entre hilos? En caso que no, ¿cuál es el argumento? (No te olvides hablar del caso de un sólo jugador que juega consigo mismo.)

Pregunta 6: [5 Puntos] Durante la fase de depuración de un programa concurrente, a la o al responsable de realizar las pruebas ocurre la idea de producir diferentes versiones del código introduciendo simples comandos de `sleep` de cierta duración al azar entre líneas sin cambiar nada más. ¿Debería funcionar el programa lógicamente igual? En caso que sí, ¿qué tipo de fallos se pueden detectar, si el programa se comporta diferente? ¿Te ocurre algún inconveniente de tal prueba (a parte que las pruebas siempre consumen tiempo en la producción de una aplicación software)?

Parte II

Pregunta 7: [5 Puntos] Durante la fase de análisis para una nueva aplicación que se debe realizar con un programa concurrente ¿cuáles serían aspectos a los cuales hay que prestar atención especial para el diseño del programa concurrente? Detalla cinco aspectos.

Pregunta 8: [5 Puntos] ¿Cuáles son las condiciones que se tienen que cumplir para que se produzca un bloqueo entre procesos? ¿Cuál es el peligro principal de un bloqueo entre procesos? ¿Cómo se puede prevenir la apariencia de un bloqueo entre procesos en muchas ocasiones ya durante el diseño de una aplicación? Razona críticamente sobre sus ventajas y limitaciones. ¿Te puedes imaginar una situación donde no se puede prevenir? Detalla.

Pregunta 9: [5 Puntos] Explica la semántica del modificador `volatile` de Java y su uso en programas concurrentes, entre otras, ¿qué tiene que ver con una relación *ha-pasado-antes*, es decir, qué garantías tiene el/a programador/a con el uso de `volatile` escribiendo y leyendo variables en su código?

Pregunta 10: [5 Puntos] Describe brevemente cinco (5) características disponibles en los paquetes `java.util.concurrent` y `java.util.concurrent.lock`. Destaca en cada caso su semántica principal y su especial relevancia para su uso en programas concurrentes.

Parte III

Pregunta 11: [5 Puntos] Describe como has implementado la medición de tiempo de ejecución de tu programa concurrente. ¿Qué tipos de tiempo hay que distinguir cuya combinación resulta en el tiempo global? ¿Cuáles fueron tus tiempos de ejecución medidos (a grandes rasgos) para el problema con la operación sobre matrices grandes y para el problema del ping-pong aumentando gradualmente el número de hilos participantes en ambos casos?

Pregunta 12: [5 Puntos] ¿Qué se entiende bajo una *condición de carrera*? Explica un caso de una condición de carrera que deberías haber considerado como posible fallo en las prácticas del ping-pong. ¿Cómo implementaste, respectivamente implementarías (en código de Java), una solución al problema?