Programación Concurrente y de Tiempo Real
Grado en Ingeniería Informática
Examen Final de la Asignatura
Junio de 2012

Apellidos: Nombre:

D.N.I.: Grupo:

1 Notas

1. Escriba su nombre, apellidos, D.N.I. y grupo en el espacio habilitado para ello, y en todos los folios blancos que utilice.
2. Firme el documento en la esquina superior derecha de la primera página.
3. Dispone de diez minutos para leer los enunciados y formular preguntas o aclaraciones sobre ellos. Transcurrido ese tiempo, no se contestarán preguntas. Dispone de 1:30 horas para completar el ejercicio.
4. No complete el documento a lápiz. Utilice bolígrafo o rotulador. Escriba con letra clara y legible. No podemos corregir lo que no se no puede leer.
5. Utilice los folios blancos que se le proporcionan para resolver los enunciados, pero traslade a este documento únicamente la solución final que obtenga, utilizando el espacio específicamente habilitado para ello, sin sobrepasarlo en ningún caso, y sin proporcionar información o respuestas no pedidas. Entregue tanto el enunciado como los folios blancos. Únicamente se corregirá este documento.

2 Criterios de Corrección

1. El examen se calificará de cero a diez puntos, y ponderará en la calificación final al 90% bajo los supuestos recogidos en la ficha de la asignatura.
2. Cada enunciado incluye información de la puntuación que su resolución correcta representa, incluida entre corchetes.
3. Un enunciado (cuestión teórica o problema) se considera correcto si la solución dada es correcta completamente. En cualquier otro caso se considera incorrecto y no puntúa.

4. Un enunciado de múltiples apartados (cuestión teórica o problema) es correcto si y solo si todos los apartados que lo forman se contestan correctamente. En cualquier otro caso se considera incorrecto y no puntúa

3 Cuestiones Teóricas de Opción Simple

Conteste VERDADERO o FALSO a las siguientes aserciones. Marque con un aspa (así ☒) la respuesta que proceda. Si se equivoca, tache y vuelva a marcar nuevamente. Cada cuestión correcta puntúa 0.1 puntos. **Se muestra un ejemplo de cuestión**.

1. En un *pool* de *threads* es posible que el número de hilo activos crezca por encima del valor fijado en el parámetro `corePoolSize` del constructor `ThreadPoolExecutor`.
   ☐ V ☐ F

4 Cuestiones Teóricas de Opción Múltiple

Escoja la respuesta correcta rodeándola con un círculo. Si se equivoca, tache y vuelva a marcar nuevamente. Marque una única respuesta. Cada cuestión correcta puntúa 0.1 puntos. **Se muestra un ejemplo de este tipo de cuestión**.

1. En la especificación JRTS los objetos `Schedulable` .

a) No guardan relación alguna con los objetos `RealTimeThread`.

b) Son una abstracción que representa objetos con capacidad de ejecución en JRTVM.

c) Únicamente tienen acceso a la memoria inmortal.

d) Ninguna de las respuestas anteriores es correcta.

5 Cuestiones de Desarrollo Corto

Conteste a las preguntas que se le formulan en el espacio habilitado para ello. Deberá razonar o justificar su respuesta simpre que se le indique. La ausencia del razonamiento o de la justificación invalidarán la respuesta al no ser esta completa.Cada cuestión correcta puntúa 1 punto. **Se muestra algunos ejemplos de este tipo de cuestión**.

1. Defina el concepto de exclusión mutua.

Escriba aquí su respuesta:

2. Describa el resultado de ejecutar el siguiente programa. Razone su respuesta.

```
import java.util.concurrent.*;
import java.util.concurrent.locks.*;
public class ej1
 implements Runnable {
     public static int n = 0;
     public static ReentrantLock k = new ReentrantLock();

 public ej1() {}
 public void run()
  {try{k.lock();
    n++;
  }finally{k.unlock();}
 }
 public static void main(String[] args) {
 ThreadPoolExecutor tp= new ThreadPoolExecutor(10,10,2000L, TimeUnit.MILLISECONDS,
            new LinkedBlockingQueue<Runnable>());
 tp.prestartAllCoreThreads();
 ej1[] tareas = new ej1[5];
 for(int k=0; k<tareas.length; k++){
     tareas[k] = new ej1();
     tp.execute(tareas[k]);
 }
 tp.shutdown();
 System.out.println(n);
 }
}
```

Escriba aquí su respuesta:

6 Problema

1. Una cuenta corriente de titularidad compartida permite a cada titular depositar o retirar dinero de la misma. Se pide escribir un monitor teórico *Hoare* que incorpore operaciones de `depósito(cantidad)` y `reintegro (cantidad)`, que responda a las siguientes especificaciones: [1 punto]

- Si un proceso externo desea realizar un reintegro y no hay saldo disponible suficiente, deberá esperar a recibir la notificación de que tal saldo existe.

- Si un proceso externo desea realizar un despósito, siempre podrá hacerlo, pero deberá considerar la posibilidad de que haya procesos en espera de la condición de saldo suficiente, y actuar en conscuencia.

(Utilice este espacio discrecionalmente para escribir su solución al problema)