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

Apellidos: Nombre:

1 Notas

1. Escriba su nombre, apellidos y grupo con letra clara y legible en el espacio habilitado para ello. Firme el documento en la esquina superior derecha de la primera hoja y escriba debajo su D.N.I.

2. Dispone de diez minutos para leer los enunciados y formular preguntas o aclaraciones sobre ellos. Transcurrido ese tiempo, no se contestarán preguntas. Dispone de 2:30 horas para completar el ejercicio.

3. Puede utilizar el material bibliográfico (libros) y copia de API que estime convenientes.Entregue sus productos en un fichero `.rar` de nombre `Apellido1_Apellido_2_Nombre`, que contendrá una subcarperta por cada enunciado del examen, la cual contendrá a su vez el conjunto de ficheros que den solución a ese enunciado en particular.

2 Criterios de Corrección

1. El examen práctico se calificará con APTO O NO APTO. Se obtiene APTO cuando al menos el 50% de los enunciados del examen son resueltos de manera correcta.

2. Las condiciones que una solución a un enunciado de examen práctico debe cumplir para considerarse correcta son:

   a) Los ficheros subidos a través del Campus Virtual que conforman el examen práctico se ajustan al número, formato y nomenclatura de nombres explicitados por el profesor en el documento de examen.

   b) El contenido de los ficheros es el especificado por el profesor en el documento de examen en orden a solucionar el enunciado en cuestión.

c) Los programas elaborados por el alumno, se pueden abrir y procesar con el compilador del lenguaje, y realizan un procesamiento técnicamente correcto,según el enunciado de que se trate.

d) Se entiende por un procesamiento técnicamente correcto a aquél código de programa que compila correctamente sin errores, cuya semántica dá soporte a la solución pedida, y que ha sido escrito de acuerdo a las convenciones de estilo y eficiencia habituales en programación

3 Enunciados

1. (Sumador Concurrente) Se dispone de un array de números enteros comprendidos entre 1 y 10 con capacidad para 1000 números. Se desea disponer de un programa que efectúe la suma de los componentes del array utilizando una estrategia concurrente, que obtenga la suma final mediante sumas parciales calculadas por una serie de `threads` independientes. Escriba un programa que realice la tarea propuesta, y que cumpla con las especificaciones siguientes:

- Diseñará una clase llamada `ThrSum.java` que definirá los hilos mediante herencia de la clase `Thread`. Dicha clase dispondrá de un constructor que recibirá como parámetros las posiciones inicial y final que definen el segmento parcial del array que a cada hilo le corresponde sumar.
- El array, cuyo nombre será datArr se rellenará inicialmente mediante números enteros en el rango de 1 a 10 generados aleatoriamente mediante el uso de la clase `java.util.Random`.
- Habrá 100 hilos cada uno de los cuales proporcionará la suma parcial de un segmento de tamaño 10 del array.
- Habrá un programa principal que creará y rellenará el array de datos `datArr` y creará y lanzará los hilos, de manera que el programa principal espere a la terminación normal de todos los hilos, tras lo cual escribirá el valor final resultante de la suma realizada por los *threads*.

Ficheros a generar: como mínimo, dos; `ThrSum.java`, que implantará la clase que modela a los hilos, y `pruebaThrSum.java` que crea los hilos, los activa, y muestra el resultado final en pantalla.

2. (Matrices Remotas) Se desea implantar una arquitectura RMI de manera que un cliente realizará en el servidor las siguiente operaciones sobre matrices cuadradas de $n \times n$: trasposición, escalado, suma y producto. Desarrolle una arquitectura RMI completa para resolver este problema, teniendo en cuenta las siguientes especificaciones:

- El cliente ofrecerá un menú en pantalla que permitirá elegir la operación realizar.
- Una vez elegida, el programa cliente pedirá al usuario los datos necesarios para realizar la operación pedida (una matriz, dos matrices, una matriz y un escalar, etc. a leer desde teclado).
- El cliente contactará con el servidor remoto, que realizará la operación y devolverá el resultado, el cuál será impreso por el cliente en el terminal.

Ficheros a generar: como mínimo, tres; `IMatriz.java`, que implantará la interfaz remota, `sMatriz.java` que implantará un servidor y `cMatriz.java` que implantará los clientes