Programación Concurrente y de Tiempo Real
Grado en Ingeniería Informática
Examen Final de Prácticas (**MODELO A**)
Junio de 2020

Apellidos: Nombre:

Grupo (A ó B):

1. Notas

1. Escriba su nombre y apellidos con letra clara y legible en el espacio habilitado para ello.
2. Firme el documento en la esquina superior derecha de la primera hoja y escriba debajo su D.N.I.
3. Dispone de 2 horas para completar el ejercicio.
4. Puede utilizar el material bibliográfico (libros) y copia de API que estime convenientes. Se prohíbe el uso de apuntes, *pen-drives* y similares.
5. Entregue sus productos, utilizando la tarea de entrega disponible en el Campus Virtual, en un fichero (`.rar, .zip`) de nombre

   `Apellido1_Apellido_2_Nombre`

   que contendrá una subcarpeta por cada enunciado del examen, la cual contendrá a su vez el conjunto de ficheros que den solución a ese enunciado en particular.
6. **No se aceptarán bajo ningún concepto entregas efectuadas por medios diferentes a la tarea de subida, ni fuera del plazo de examen establecido. Por tanto, debe asegurarse de entregar sus productos en el plazo establecido y mediante la tarea habilitada a tal efecto.**

2. Criterios de Corrección

1. El examen práctico se calificará de cero a diez puntos y ponderará en la calificación final al 30 % bajos los supuestos recogidos en la adenda a la ficha de la asignatura.

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. (Ecuación de Ondas-2D, 5.5 puntos) Suponga una superficie discreta $A_{n,n}$ donde $A(i,j) \in \{-100, ..., 0, ..., 100\}$. La ecuación de ondas bidimensional discreta en tal superficie viene descrita por la ecuación:

$$A_{(i,j)}^{(t+1)} = c^2 \left(A_{(i+1,j)}^{(t)} + A_{(i-1,j)}^{(t)} + A_{(i,j+1)}^{(t)} + A_{(i,j-1)}^{(t)} - 4A_{(i,j)}^{(t)} \right) + 2A_{(i,j)}^{(t)}$$

donde c es un número real entre 0 y 1 (ambos excluidos) que permite aproximar el modelo discreto al modelo continuo, y que tomaremos igual a $1/4$. Necesitará redondear valores a números enteros (por ejemplo, 1,8 se truncará a 1 y $-1{,}8$ se truncará a -1).Para estudiar la propagación de una onda en la superficie, se establece un estado inicial de la misma, y se aplica esta ecuación para un determinado número de intervalos temporales. Se pide escribir un programa en Java que implemente el cálculo mediante procesamiento paralelo de la ecuación ondas. Para ello deberá:

- Leer el número de intervalos temporales.
- Efectuar la división del dominio de datos de forma automatizada, en función del coeficiente de bloqueo C_b que estime adecuado para el problema y el número de núcleos lógicos disponibles en la arquitectura.
- Modelar sus tareas mediante la interfaz `Runnable`.
- Elegir una condición de frontera nula.
- Procesar sus tareas mediante un ejecutor de clase `ThreadPoolExecutor`.

- Ofrecer un corto menú de dos opciones, que permita al usuario cargar la retícula con datos aleatorios en el rango indicado (en este caso ajustar la dimensión $n = 10^3$ de forma automática), y con datos concretos introducidos manualmente por teclado (en este caso, la dimensión n se pedirá por teclado al usuario). En ambos casos, el programa ofrecerá al terminar su ejecución el tiempo total de cálculo.
- La ejecución mostrará en pantalla el estado de la superficie A en cada instante del tiempo, si se elige la opción de menú de datos introducidos por teclado.
- La ejecución mostrará además el valor de speedup alcanzado, si se elige la opción de menú de datos aleatorios.
- Si lo necesita, puede ampliar el tamaño del *heap* de la máquina virtual utilizando el *flag* `-Xmx`

Guarde su código en `waveEq2D.java`.

2. (Control de Aforo, 4.5 puntos) Se desea controlar el aforo a una playa gaditana por razones sanitarias. El sistema debe permitir la entrada a los usuarios hasta alcanzar el aforo máximo, establecido en 100 personas. Las personas que llegan después deben esperar hasta que otro usuarios se marchen y dejen aforo libre. Escriba un monitor en lenguaje C++ que modele el problema, y pruébelo utilizando un diseño con 200 personas simuladas mediante hebras concurrentes. Guarde su código en `monitorPlaya.cpp`.