UVG
UNIVERSIDAD DEL VALLE DE GUATEMALA

Universidad del Valle de Guatemala
Facultad de Ingeniería
Departamento de Ciencias de la Computación
CC3086 Programación de microprocesadores

Ciclo 2 de 2,024
Temario A

Laboratorio 05

Competencias para desarrollar
Distribuir la carga de trabajo entre hilos utilizando programación en C y OpenMP.

Instrucciones
Esta actividad se realizará individualmente. Al finalizar los períodos de laboratorio o clase, deberá entregar este archivo en formato PDF y los archivos .c en la actividad correspondiente en Canvas.

1. **(18 pts.)** Explica con tus propias palabras los siguientes términos:
   a) private: Significa que cada hilo tendrá su respectiva variables. Lo cual es útil cuando cada hilo necesita trabajar con su propia versión de variable sin interferir con los demás hilos.
   b) shared: Se comparte entre todos los hilos que ejecutan la seccion paralela. Todos los hilos pueden leer y escribir en esa variable, lo que puede ser útil pero también riesgoso si no se gestiona bien
   c) firstprivate:Cada hilo comienza con su propia copia de la variable, pero esta copia se inicializa con el valor de la variable original.
   d) barrier :Es cuando se encuentra una barrera, todos los hilos deben esperar hasta que todos los demás hilos hayan alcanzado esa misma barrera antes de continuar.
   e) critical :Es un bloque de código que solo puede ser ejecutado por un hilo a la vez. Esto es útil para proteger secciones de código que no deben ser ejecutadas simultáneamente por varios hilos.
   f) atomic : Asegura que la operación sobre la variable se ejecute de manera atómica, es decir, sin interferencias de otros hilos, pero con menos sobrecarga

2. **(12 pts.)** Escribe un programa en C que calcule la suma de los primeros N números naturales utilizando un ciclo ***for*** **paralelo**. Utiliza la cláusula **reduction con +** para acumular la suma en una variable compartida.
   a) Define N como una constante grande, por ejemplo, N = 1000000.
   b) Usa omp_get_wtime() para medir los tiempos de ejecución.

3. **(15 pts.)** Escribe un programa en C que ejecute tres funciones diferentes en paralelo usando la **directiva #pragma omp sections**. Cada sección debe ejecutar una función distinta, por ejemplo, una que calcule el factorial de un número, otra que genere la serie de Fibonacci, y otra que encuentre el máximo en un arreglo, operaciones matemáticas no simples. Asegúrate de que cada función sea independiente y no tenga dependencias con las otras.

4. **(15 pts.)** Escribe un programa en C que tenga un ciclo for donde se modifiquen dos variables de manera paralela usando #pragma omp parallel for.
   a. Usa la cláusula shared para gestionar el acceso a la variable1 dentro del ciclo.
   b. Usa la cláusula private para gestionar el acceso a la variable2 dentro del ciclo.
   c. Prueba con ambas cláusulas y explica las diferencias observadas en los resultados.

5. **(30 pts.)** Analiza el código en el programa Ejercicio_5A.c, que contiene un programa secuencial. Indica cuántas veces aparece un valor key en el vector a. Escribe una versión paralela en OpenMP utilizando una descomposición de tareas **recursiva**, en la cual se generen tantas tareas como hilos.

6. REFLEXIÓN DE LABORATORIO: se habilitará en una actividad independiente.