INTRODUCCIÓN A LA CONCURRENCIA

TRABAJO PRÁCTICO N° 3

“OpenMP”

RESUMEN

Para compilar programas en C utilizando la librería OpenMP, se debe realizar con la directiva −fopenmp. Para compilar con Makefile, podemos incluirlo dentro de las directivas de compilación.

Ejemplo de compilación:

```
gcc fuente.c -o ejecutable -fopenmp
```

EJERCICIOS

1. Cree un programa en C que muestre el ID de cada hilo dentro dos bloques paralelos distintos OpenMP y la cantidad de hilos que se crearon. En el primer bloque la cantidad de hilos se indicará mediante una “función de OpenMP” y en el segundo bloque mediante una “cláusula de OpenMP”. La cantidad de hilos debe ser suministrada por parámetro. En caso de no ingresarse un valor o que el valor sea menor a 2, ejecutar el bloque paralelo como si fuera uno secuencial.

   Ejemplo de impresión por consola:

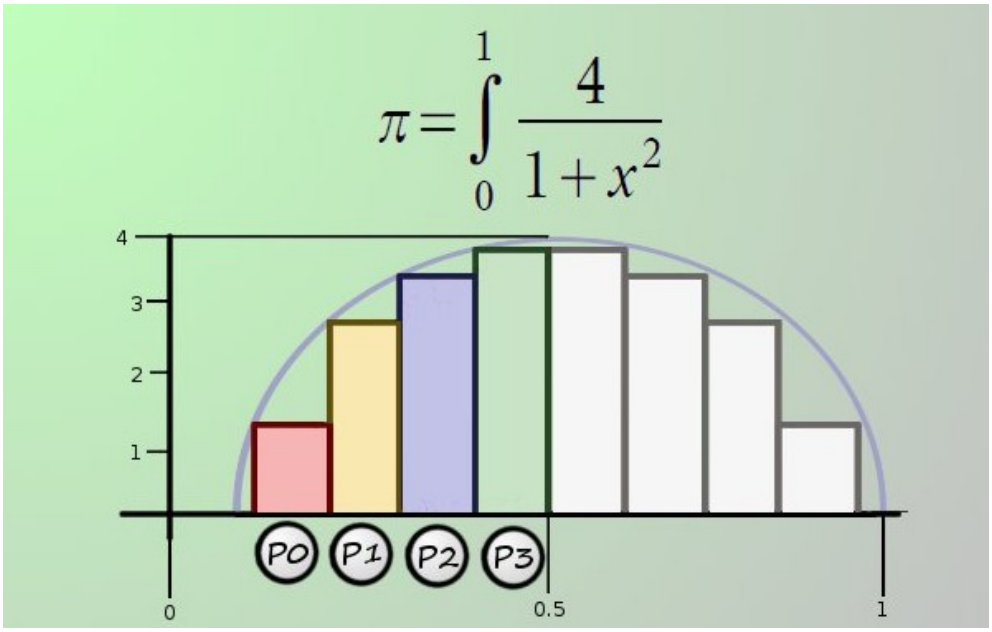
   ```
   Print “Soy el hilo 1 de 8 hilos y pertenezco al primer bloque”
   ```

2. Genere un programa en C que reparta la tarea de un bloque paralelo “for” (100 iteraciones) de forma dinámica en grupos de 15 iteraciones. Cada iteración debe mostrar por consola el número de hilo y número de iteración. Analice los resultados obtenidos por consola.

3. Cree un programa en C que realice dos funciones de manera concurrente. Estas funciones se llamarán “funcion_tiempo_1” y “funcion_tiempo_2”. Cada función hará un sleep(1) y sleep(2) respectivamente. Para esto debe usar la directiva “sections”. Medir el tiempo de ejecución de todo el programa con la función “omp_get_wtime()”, y mostrarlo por consola. Cada función genera un double, que al restarlos, se obtiene el tiempo de duración. Ejemplo:

```
double inicio, fin;

inicio = omp_get_wtime();

… Procesaminto …

fin = omp_get_wtime();

resultado = fin – inicio;
```

4. Dado la siguiente solución secuencial escrita en C, la cual realiza el cálculo de pi por aproximación, “convertirla” a concurrente usando OpenMP y la cláusula “reduction”. Al probar variando la cantidad de “num_steps” se va a conseguir mayor o menor precisión. A continuación se verá la explicación grafica del cálculo en sí mismo y el código secuencial:



solución secuencial

```
static long num_steps = 100000; //Factor de precisión

double step;

void main ()

{

        int i; double x, pi, sum = 0.0;

        step = 1.0/(double) num_steps;

        for (i=0;i< num_steps; i++){

                x = (i+0.5)*step;
```

```
            sum = sum + 4.0/(1.0+x*x);
        }
        pi = step * sum;
    }
```

5. Cree un programa en C que contenga un bloque paralelo y se ejecute con 4 hilos, donde cada hilo muestre su ID por consola en exclusión mutua, para lo cual se debe usar la directiva “critical”. Luego de que cada hilo muestre su ID, generar una barrera con la directiva “barrier” y mostrar un mensaje que termino la ejecución.