2º curso / 2º cuatr.

Grado en Ing. Informática

Arquitectura de Computadores: Exámenes y Controles

Examen de Prácticas 20/06/2012 resuelto

Material elaborado por los profesores responsables de la asignatura:
Mancia Anguita, Julio Ortega

1 Enunciado Examen de Prácticas del 20/06/2012

Cuestión 1.(1 punto) Considere el programa de la siguiente figura:

```
#include <stdio.h>
#include <stdlib.h>
#include <omp.h>
int main(int argc, char **argv)  {
  int i, n=20, tid, a[n],suma=0,sumalocal;
  if(argc < 2)    {
     fprintf(stderr,"\nFalta iteraciones\n");
     exit(-1);
    }
  n = atoi(argv[1]); if (n>20) n=20;

  for (i=0; i<n; i++)   a[i] = i;

 #pragma omp parallel private(sumalocal,tid)
 {  sumalocal=0;
    tid=omp_get_thread_num();
    #pragma omp) for  schedule(static)
    for (i=0; i<n; i++)
    {    sumalocal += a[i];
         printf(" thread %d suma de a[%d]=%d sumalocal=%d \n",  tid,i,a[i],sumalocal);
    }
        suma += sumalocal;
    #pragma omp barrier
    #pragma omp master
        printf("thread master=%d imprime suma=%d\n", tid,suma);
 }
}
```

(a) ¿Permite calcular correctamente la suma de todos los elementos del array `a[]`? (Indique cómo solucionaría el problema en el caso de que lo hubiera).

(b) ¿Qué se pretende al incluir `#pragma omp master` en este programa?

(c) ¿Se puede prescindir de `#pragma omp barrier`? (Justifique su respuesta).

(d) ¿Cuál es la utilidad de la clausula `schedule(static)` en la directiva `#pragma omp for schedule(static)`?

(e) Qué diferencias habría en lo que imprime en pantalla el programa si se sustituyera `master` por `single` en el programa? Razone su respuesta. ¿Para qué sirve la función `omp_get_thread_num()`?

Cuestión 2. (1 punto) Considere el programa de la siguiente figura

```
#include <stdio.h>
#include <stdlib.h>
#ifdef _OPENMP
 #include <omp.h>
#else
  #define omp_get_thread_num() 0
#endif

main(int argc, char **argv)  {

  int i, n = 7,chunk, a[n],suma=0;

  if(argc < 2)    {
       fprintf(stderr,"\nFalta chunk \n");
       exit(-1);
    }
  chunk = atoi(argv[1]);

  for (i=0; i<n; i++)    a[i] = i;

  #pragma omp parallel for firstprivate(suma) lastprivate(suma) schedule(static,chunk)
  for (i=0; i<n; i++)
  {  suma = suma + a[i];
     printf(" thread %d suma a[%d] suma=%d \n",
          omp_get_thread_num(),i,suma);
  }

  printf("Fuera de 'parallel for' suma=%d\n",suma);

}
```

(a) ¿Permite calcular la suma de los componentes del vector `a[]`? Justifique su respuesta.

(b) ¿Qué imprime el código cada vez que se ejecuta la función `printf` del bucle?

(c) ¿Qué imprime el código cuando se ejecuta la segunda función `printf`?

(d) ¿Para qué sirve el parámetro `chunk` en la construcción `#pragma omp parallel for firstprivate(suma) lastprivate(suma) schedule(static,chunk)`?

(e) ¿Para qué sirven las clausulas `firstprivate(suma)` y `lastprivate(suma)`?

(f) ¿qué imprime el programa si se eliminan `firstprivate(suma)` y `lastprivate(suma)` y se incluye `reduction(+:suma)` en la construcción `#pragma omp parallel for`?