Comando de utilidad en OpenMP

#pragma omp parallel:

Esta directiva crea un equipo de hilos, y el bloque de código dentro de ella se ejecuta en paralelo por cada hilo del equipo.
Por ejemplo:

```
#pragma omp parallel
{
    // Código ejecutado en paralelo por todos los hilos del equipo
}
```

#pragma omp for:

Se utiliza para paralelizar bucles. Distribuye las iteraciones de un bucle entre los hilos del equipo.
Por ejemplo:

```
#pragma omp parallel for
for (int i = 0; i < n; ++i) {
    // Código dentro del bucle ejecutado en paralelo
}
```

#pragma omp critical:

Indica una sección crítica del código que solo puede ser ejecutada por un hilo a la vez. Esto evita la condición de carrera.
Por ejemplo:

```
#pragma omp parallel
{
    #pragma omp critical
    {
        // Código crítico ejecutado por un solo hilo a la vez
    }
}
```

#pragma omp barrier:
Crea una barrera de sincronización entre los hilos. Todos los hilos deben alcanzar la barrera antes de continuar.
Por ejemplo:

```
#pragma omp parallel
{
   // Código ejecutado en paralelo

   #pragma omp barrier
    // Todos los hilos esperan aquí hasta que todos lleguen
}
```

#pragma omp sections y **#pragma omp section**:
Divide el código en secciones ejecutadas en paralelo. Cada sección es ejecutada por un solo hilo.
Por ejemplo:

```
#pragma omp parallel
{
   #pragma omp sections
   {
      #pragma omp section
      {
         // Código ejecutado por un hilo
      }

      #pragma omp section
      {
         // Código ejecutado por otro hilo
      }
   }
}
```

#pragma omp reduction:

Se utiliza para realizar reducciones en paralelo, como sumas o productos.
Por ejemplo:

```
int sum = 0;

#pragma omp parallel for reduction(+:sum)
for (int i = 0; i < n; ++i) {
    sum += i;
}
```

#pragma omp atomic:

Utilizada para garantizar que la operación especificada se realice atómicamente, evitando condiciones de carrera. La operación se ejecuta como una única unidad indivisible, sin intervención de otros hilos o procesos en el mismo instante
Por ejemplo:

```
#pragma omp parallel
{
    #pragma omp for
    for (int i = 0; i < n; ++i) {
        #pragma omp atomic
        sum += i;  // Operación atómica de suma
    }
}
```

#pragma omp single:

Indica que un bloque de código debe ser ejecutado solo por uno de los hilos del equipo. Útil para operaciones que deben realizarse solo una vez.
Por ejemplo:

```
#pragma omp parallel
{
    // Código ejecutado en paralelo
    #pragma omp single
    {
        // Código ejecutado solo por un hilo
    }
}
```

#pragma omp master:
Similar a **#pragma omp single**, pero solo el hilo maestro (hilo con ID 0) ejecutará el bloque de código. Por ejemplo:

```
#pragma omp parallel
{
   // Código ejecutado en paralelo
   #pragma omp master
   {
      // Código ejecutado solo por el hilo maestro
   }
}
```

#pragma omp task:
Se utiliza para generar tareas que pueden ejecutarse en paralelo. Puede combinarse con **#pragma omp taskwait** para sincronizar las tareas.
Por ejemplo:

```
#pragma omp parallel
{
   #pragma omp single
   {
      #pragma omp task
      {
         // Tarea 1
      }

      #pragma omp task
      {
         // Tarea 2
      }

      #pragma omp taskwait
      // Esperar a que todas las tareas finalicen
   }
}
```