Общие механизмы OpenMP. Условия выполнения

```
#pragma omp parallel \
  shared(var1, var2, ....) \
  private(var1, var2, ...) \
  firstprivate(var1, var2, ...) \
  reduction(оператор:var1, var2, ...) \
  if(выражение) \
  default(shared|private|none)
{
   // parallel block
}
```

Общие механизмы OpenMP. Условия выполнения

- **`shared(var1, var2, ....)`**
  - *перечисленные переменные будут разделяться между потоками. Все потоки будут обращаться к одной и той же области памяти.*
- **`private(var1, var2, ...)`**
  - *каждый поток должен иметь свою копию переменной на всем протяжении своего исполнения.*
- **`firstprivate(var1, var2, ...)`**
  - *инициализируются при входе в параллельный участок кода значением, которое имела переменна до входа в параллельную секцию.*
- **`lastprivate(var1, var2, ...)`**
  - *сохраняют свое значение, которое они получили при достижении конца параллельного участка кода.*
- **`reduction(оператор:var1, var2, ...)`**
  - *гарантирует безопасное выполнение операций редукции, например, вычисление глобальной суммы.*
- **`if(выражение)`**
  - *параллельное выполнение необходимо только если выражение истинно.*
- **`default(shared|private|none)`**
  - *область видимости переменных внутри параллельного участка кода по умолчанию.*
- **`schedule(type[,chunk])`**
  - *контролируется то, как итерации цикла распределяются между потоками.*

Общие механизмы OpenMP. Условия выполнения

```
#pragma omp parallel shared(a) private(myid, x)
{
  myid = omp_get_thread_num( );
  x = work(myid);
  if(x < 1.0)
    a[myid] = x;
}
```

```
#pragma omp parallel default(private) shared(a)
{
  myid = omp_get_thread_num( );
  x = work(myid);
  if(x < 1.0)
    a[myid] = x;
}
```

Общие механизмы OpenMP. Условия выполнения

if(clause)

```
#pragma omp parallel
{
  #pragma omp for if(n>2000)
  {
    for(i=0; i<n; i++)
      a[i] = work(i);
  }
}
```

lastprivate(var [, var2 …])

```
#pragma omp parallel
{
  #pragma omp for private(i) lastprivate(k)
    for(i=0; i<10; i++)
      k = i*i;
}
printf("k = %d\n", k);
```

Общие механизмы OpenMP. Условия выполнения

reduction(op:var1 [, var2 ...])

```
#pragma omp parallel
{
  #pragma omp for shared(x) private(i, sum) reduction(+:sum)
    for(i=0; i<10000; i++)
      sum += x[i];
}
```

```
#pragma omp parallel
{
  #pragma omp for shared(x) private(i, gmin) reduction(min:gmin)
    for(i=0; i<10000; i++)
      gmin = min(gmin, x[i]);
}
```

Общие механизмы OpenMP. Условия выполнения

reduction(op:var1 [, var2 ...])
C/C++ +, - , *, &, ^, |, &&, ||, min, max
Fortran +, -, *, .and., .or., .eqv., .neqv., min, max, iand, ior, ieor