[20] 0 сек.

```
!g++ -O3 -fopenmp task1_openmp_performance.cpp -o task1_openmp
```

[21] 0 сек.

```
!./task1_openmp
```

```
TASK 1 — OpenMP performance
Threads: 2
Sequential time: 0.00212888 s
Parallel time:   0.00790564 s
Total sum:       5.00089e+06
```

[22] 2 сек.

```
!nvcc -O3 task2_gpu_memory_access.cu -o task2_gpu
```

[23] 0 сек.

```
!./task2_gpu
```

```
TASK 2 — GPU memory access
Coalesced time:     11.153 ms
Non-coalesced time: 0.003968 ms
```

4] 2 сек.

```
!nvcc -O3 task3_hybrid_async.cu -o task3_hybr
!./task3_hybrid
```

```
TASK 3 — Hybrid async completed
```

]

Напишите программный код или сгенерируйте ог

```
!mpic++ -O3 task4_mpi_scaling.cpp -o task4_mpi
!mpirun --allow-run-as-root -np 1 ./task4_mpi
```

```
TASK 4 — MPI scaling
Processes: 1
Time: 0.0141999 s
Sum: 1e+07
```

```
!mpirun --allow-run-as-root --oversubscribe -np 4 ./task4_mpi
```

```
TASK 4 — MPI scaling
Processes: 4
Time: 0.0101567 s
Sum: 1e+07
```

Ответы на контрольные вопросы

---

1. В чём отличие измерения времени выполнения от профилирования?

Измерение времени — фиксирует общее время выполнения участка кода.

Профилирование — детально анализирует, где именно тратится время (функции, потоки, память, коммуникации).

---

2. Какие виды узких мест характерны для CPU, GPU и распределённых программ?

- CPU: ограничение по числу ядер, кэш, ветвления, синхронизация потоков.
- GPU: доступ к памяти, некоалесцированные обращения, divergence.
- MPI: коммуникации, синхронизации, дисбаланс нагрузки.

---

3. Почему увеличение числа потоков или процессов не всегда приводит к ускорению?

- существует последовательная часть (закон Амдала);
- растут накладные расходы синхронизации;
- ухудшается доступ к памяти;
- коммуникации начинают доминировать.

---

4. Как законы Амдала и Густафсона применяются при анализе масштабируемости?

- Закон Амдала: ускорение ограничено долей последовательного кода (strong scaling).
- Закон Густафсона: при росте задачи эффективность масштабирования выше (weak scaling).

---

5. Какие факторы наиболее критичны для производительности гибридных приложений?

- объём и частота передачи данных CPU ↔ GPU;
- возможность перекрытия передачи и вычислений;
- баланс нагрузки между CPU и GPU;
- организация памяти и асинхронные операции.
