Результаты:

```
[20]
✓ 2s
!nvcc -O2 -arch=sm_75 task1_parallel_stack.cu -o task1_stack
!./task1_stack

Stack output (pop results):
0 255 254 253 252 251 250 249 248 247
```

```
[21]
✓ 2s
!nvcc -O2 -arch=sm_75 task2_parallel_queue.cu -o task2_queue
!./task2_queue

Queue output (dequeue results):
0 0 0 0 0 0 0 0 0 0
```

```
[22]
✓ 3s
!nvcc -O2 -arch=sm_75 task3_compare_performance.cu -o task3_perf
!./task3_perf 100000 4

Task3: Compare Stack vs Queue performance
threads_total=100000, ops_per_thread=4, total_ops=400000, capacity=400000

=== Results (time in ms) ===
STACK push: 0.196576 ms, ok=400000, throughput=2.03484e+06 ops/ms
STACK pop : 0.045024 ms, ok=400000, throughput=8.88415e+06 ops/ms

QUEUE enq : 0.038912 ms, ok=400000, throughput=1.02796e+07 ops/ms
QUEUE deq : 0.03888 ms, ok=400000, throughput=1.02881e+07 ops/ms

=== Correctness check (counts) ===
Expected ops <= capacity: 400000
Stack push ok: 400000, Stack pop ok: 400000
Queue enq  ok: 400000, Queue deq  ok: 400000
```

Контрольные вопросы:

1. Отличие стека и очереди

Стек — LIFO, очередь — FIFO.

2. Проблемы параллельного доступа

Гонки данных, неконсистентность, потеря данных.

3. Роль атомарных операций

Гарантируют корректное обновление общих переменных.

4. Типы памяти CUDA

Глобальная, разделяемая, локальная.

5. Влияние синхронизации

Повышает корректность, но может снижать производительность.

6. Зачем нужна shared memory

Уменьшает задержки и число обращений к глобальной памяти.