Part1:

```
Enter N: 100
Sequential min = 1 max = 100
Parallel   min = 1 max = 100
Sequential time = 1 us
Parallel time   = 0 us
PS C:\Users\Timing\Documents\GitHub\Heterogeneous-Parallelization\practices>
```

Part2:

```
PS C:\Users\Timing\Documents\GitHub\Heterogeneous-Parallelization\practices> C:/Users/Timing/anaconda3/Scri
pts/activate
Find 20 = 1
Stack pop = 2
Queue pop = 100
Enter number of elements: 100
Parallel insert finished
PS C:\Users\Timing\Documents\GitHub\Heterogeneous-Parallelization\practices>
```

Part3:

```
Enter N: 1000
Sequential average = 51.457 time = 1 us
Parallel   average = 51.457 time = 2 us
PS C:\Users\Timing\Documents\GitHub\Heterogeneous-Parallelization\practices>
```

```
pts/activate
Enter N: 1000000000
Sequential average = 50.5021 time = 1841541 us
Parallel   average = 50.5021 time = 2158216 us
PS C:\Users\Timing\Documents\GitHub\Heterogeneous-Parallelization\practices>
```

Контрольные вопросы

1. Чем отличаются массивы от динамических структур данных?

Массив — это набор элементов, расположенных подряд в памяти.

К любому элементу можно обратиться по индексу за O(1), но размер массива фиксированный.

Динамические структуры (списки, стек, очередь) строятся из отдельных узлов, связанных указателями.

Их размер можно менять во время работы программы, но доступ к элементам обычно требует прохода по структуре и работает медленнее, чем у массива.

2. Что такое указатель в C++?

Указатель — это переменная, которая хранит адрес другой переменной в памяти.

Через указатель можно получить доступ к данным, находящимся по этому адресу, и изменять их.

Указатели используются для работы с динамической памятью и структурами данных, такими как списки и деревья.

3. Как работает стек и очередь?

Стек работает по принципу LIFO (Last In – First Out):

последний добавленный элемент извлекается первым.

Очередь работает по принципу FIFO (First In – First Out):

первый добавленный элемент извлекается первым.

4. Какие преимущества и недостатки у односвязного списка по сравнению с массивом?

Преимущества:

можно легко добавлять и удалять элементы;

размер не фиксирован.

Недостатки:

требуется больше памяти из-за указателей;

доступ к элементам медленнее, так как нужно проходить список по узлам;

нет прямого доступа по индексу.

5. Как правильно освобождать динамическую память?

Если память выделена с помощью new, её нужно освобождать с помощью delete.

Если память выделена через new[], её нужно освобождать через delete[].

Для списков, стеков и очередей нужно удалять каждый узел по отдельности, проходя по структуре.

6. Почему важно понимать работу с указателями и динамической памятью в многопоточности?

Потому что несколько потоков могут одновременно обращаться к одним и тем же данным.

Если не контролировать доступ, возникают ошибки — утечки памяти, повреждение данных и гонки потоков.

Поэтому при работе с динамическими структурами в параллельных программах нужно использовать синхронизацию.

7. Что делает директива reduction в OpenMP?

reduction позволяет безопасно объединять значения, которые вычисляются в разных потоках.

Например, при суммировании каждый поток считает свою часть суммы, а затем все значения складываются в одну общую переменную.

8. Как параллельная обработка влияет на производительность при работе с массивами?

При больших объёмах данных параллельная обработка обычно ускоряет вычисления, потому что работа распределяется между потоками.

Но из-за накладных расходов на создание потоков и доступа к памяти для маленьких массивов параллельная версия может работать даже медленнее.