**Отчёт по лабораторной работе № 2**

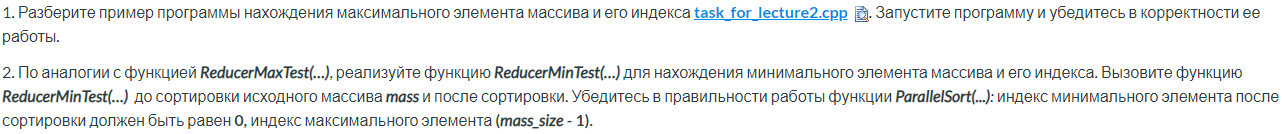
**Жуков Вадим, ИВТ-12М**

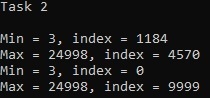
Используемое железо: 2 физических ядра, 4 логических, 3.4ГГц

ОС: Windows 10

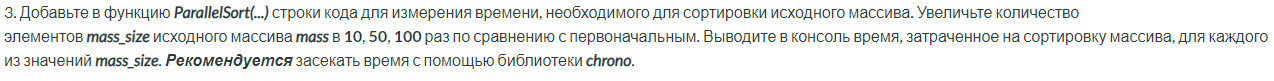
VS: Visual Studio 2017 v15.9.18

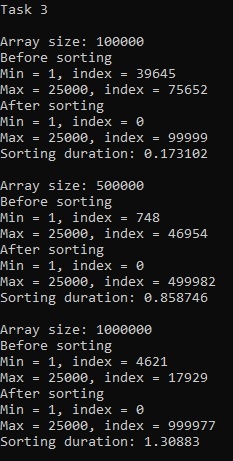
IPS: v2019



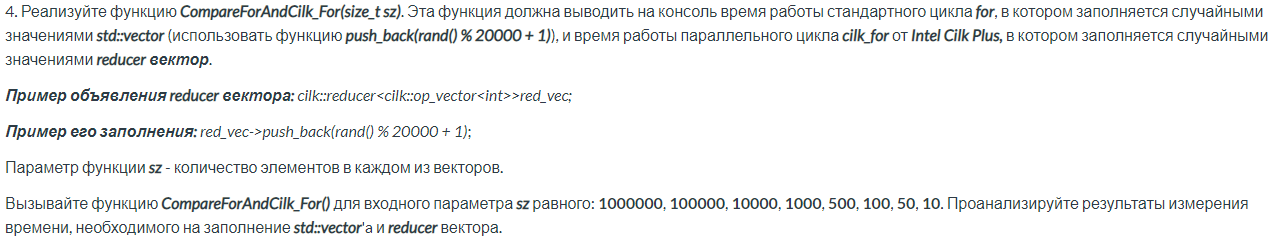


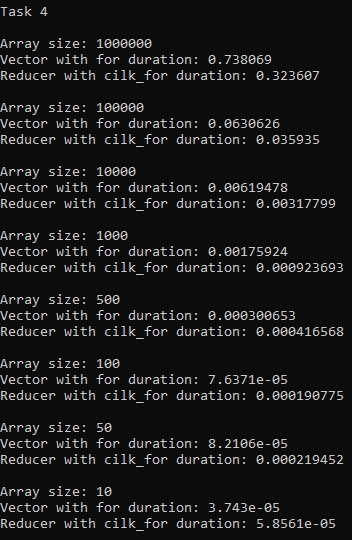
Немного изменил вывод и добавил функцию ReducerMinTest. Как видно в результатах, минимальный и максимальный элементы находятся в начале и конце соответственно, значит сортировка работает. В зависимости от задания я добавил соответствующую приписку каждой функции, чтобы можно было запустить всё в одном файле.





Добавил замер времени, за которое производится сортировка, и измерил её для различных размерностей массива. Как видно по результатам, сортировка работает корректно: минимальный и максимальный элементы находятся на правильных местах и время сортировки увеличивается при увеличении размерности массива.





На картинке представлено заполнение вектора с использованием стандартного for и заполнение reducer вектора с использованием cilk\_for. Как видно, при маленьких размерностях массива (10, 50, 100, 500) стандартный for отрабатывает быстрее, при больших (1000 и больше) – cilk\_for оказывается эффективнее.

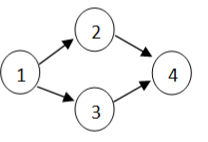


cilk\_for помимо вычисления также тратит время на создание потоков, распределение задач между ними и переключение контекста, которое сопоставимо с временем работы стандартного for при малых размерностях массива, поэтому получилось такое расхождение в 4 задании. Его использование будет эффективным при достаточно большом размере массива. Как показала практика, достаточно большая размерность для применения cilk\_for – 1000 элементов и больше.

cilk\_spawn: используется для указания, что данная функция может вызываться параллельно с вызывающей.

cilk\_sync: используется для синхронизации результатов родительской функции и дочерней.

На рисунке функции 2 и 3 вызываются параллельно, функция 4 начинает выполнение только по окончании работы предыдущих двух.



cilk\_for: используется для распараллеливания циклов с известным количеством повторений. В процессе компиляции тело цикла конвертируется в функцию, которая вызывается рекурсивно. Планировщик автоматически распределяет поддеревья рекурсии между обработчиками. На рисунке ниже показан принцип его работы.

