НАУЧНО ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

МОСКОВСКИЙ ИНСТИТУТ ЭЛЕКТРОННОЙ ТЕХНИКИ

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1:

По дисциплине: “Разработка многопоточных приложений в среде IPS ”

Преподаватель

Д. В. Калеев

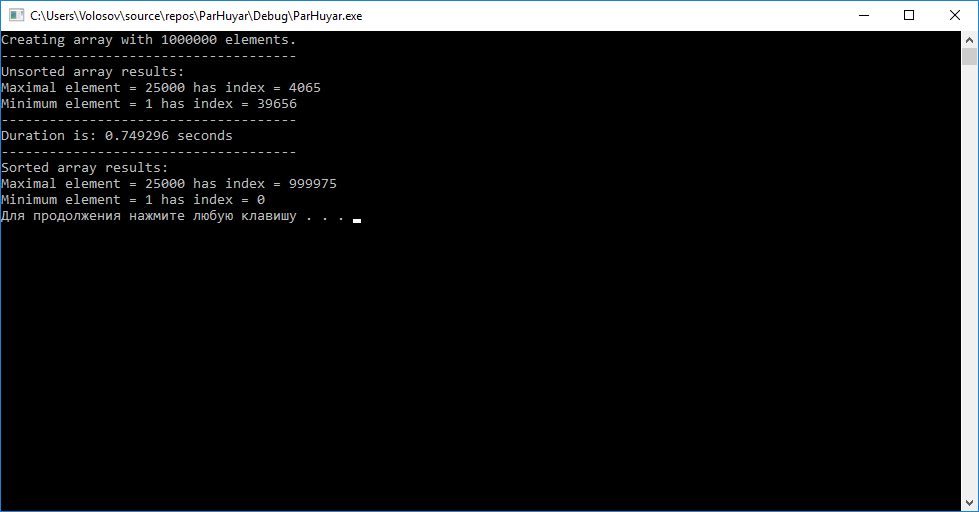
Выполнил

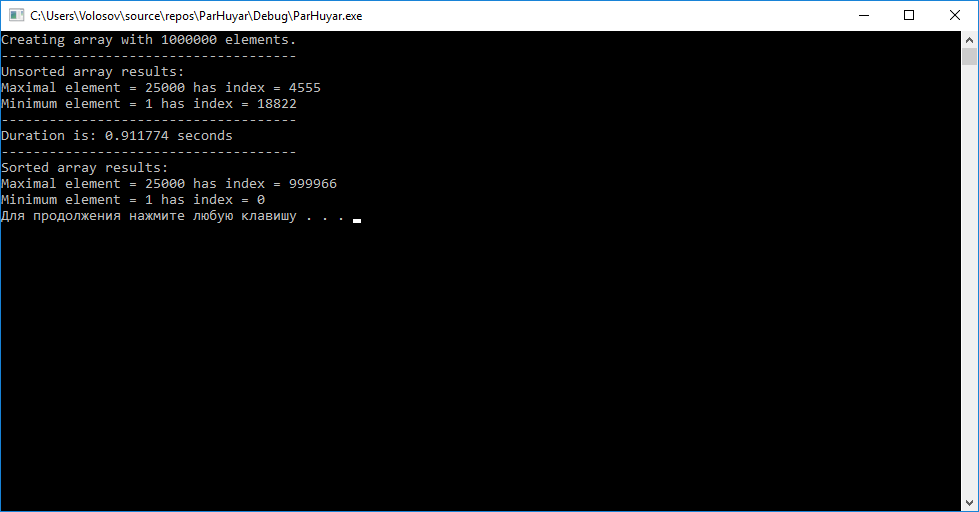
П.А. Николенко, ИВТ-21М

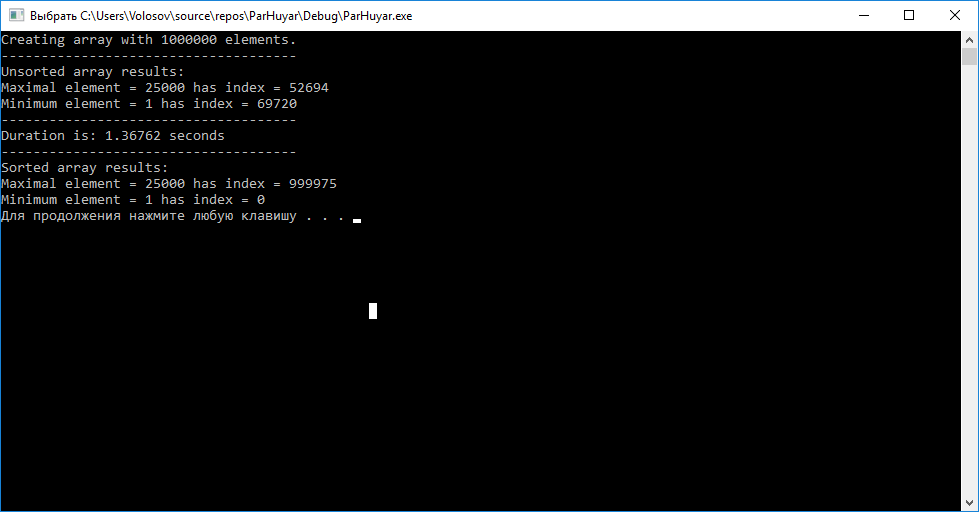
**Задания 1-3**

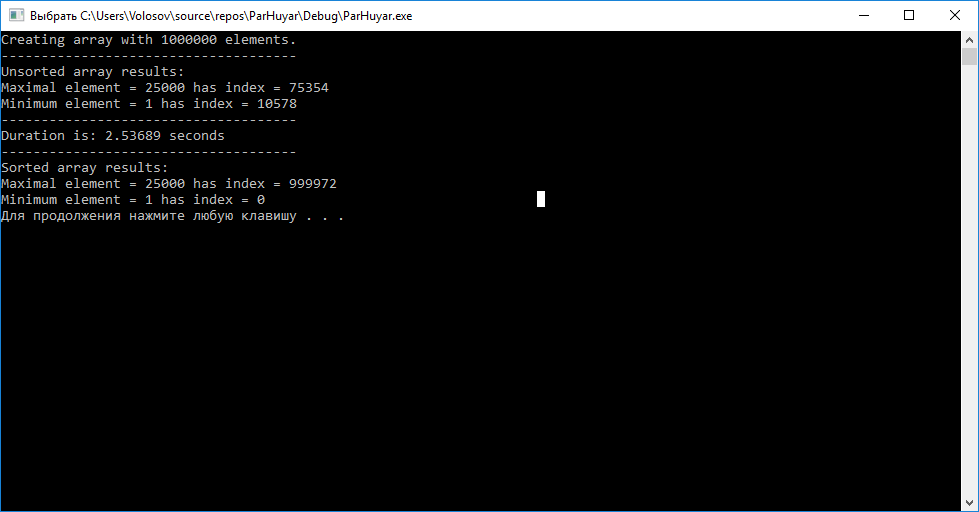
Результаты поиска максимального и минимального элемента в массиве из миллиона элементов. В ходе проверки видно, что максимальный элемент стоит не в самом конце. Это происходит из-за особенностей кода, так как он не учитывает, что может быть несколько максимальных элементов, которые равны друг другу и соответственно занимают несколько последних ячеек в массиве после сортировки. Так как данная часть кода уже была в лабе, то считал это не багом а фичей.

Обработка масссива при 8, 4, 2 и 1 потоке соответвственно.

*Рис 1. Вычисления на 8 потоках.*

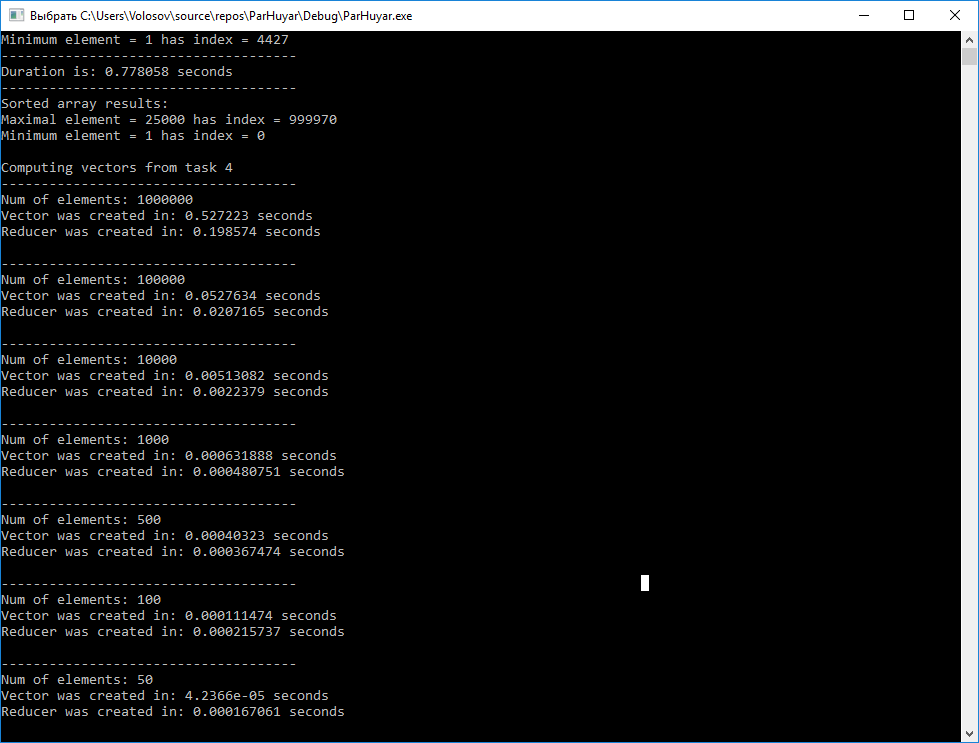
*Рис 2. Вычисления на 4 потоках.*

*Рис 3. Вычисления на 2 потоках.*

*Рис 4. Вычисления на 1 потоке.*

**Задание 4**

Время, которое требуется для объявления обычного вектора с помощью for и reducer вектора с помощью метода cilk\_for.

*Рис 5. Замеры времени, необходимого на инициализацию векторов разных размеров.*

**Задание 5**

**В:** Почему при небольших значениях **sz**цикл **cilk\_for**уступает циклу **for**в быстродействии?

**О:** Параллелизм имеет свои издержки. В частности, требуется время на то, чтобы разбить операции между разными ядрами, а потом собрать полученные данные обратно воедино. Соответственно доля операций, которые тратятся на распараллеливание в общем числе исполняемых операций тем выше, чем меньше объем операций, которые мы хотим распараллелить.

**В:** В каких случаях целесообразно использовать цикл **cilk\_for** ?

**О:** Когда нам требуется прогнать через цикл большой объем данных или выполнить большое число повторяющихся операций.

**В:** В чем принципиальное отличие параллелизации с использованием **cilk\_for** от параллелизации с использованием **cilk\_spawn**в паре с **cilk\_sync**?

**О: Cilk\_spawn** и **cilk\_sync** – пара команд, с помощью которых мы вручную создаем задачу распараллелить функции, находящиеся между ними. Создание задач требует ресурсов, поэтому использовать их внутри циклов не выгодно.

**Cilk\_for** – Разовое объявление задачи распараллеливания, которая распространяется на все команды внутри подобного цикла.

Генеральный спонсор ответов: <https://habr.com/ru/company/intel/blog/204838/>

**Задание 6**

**GitHub:** <https://github.com/mrgrin/IPS_Labs>