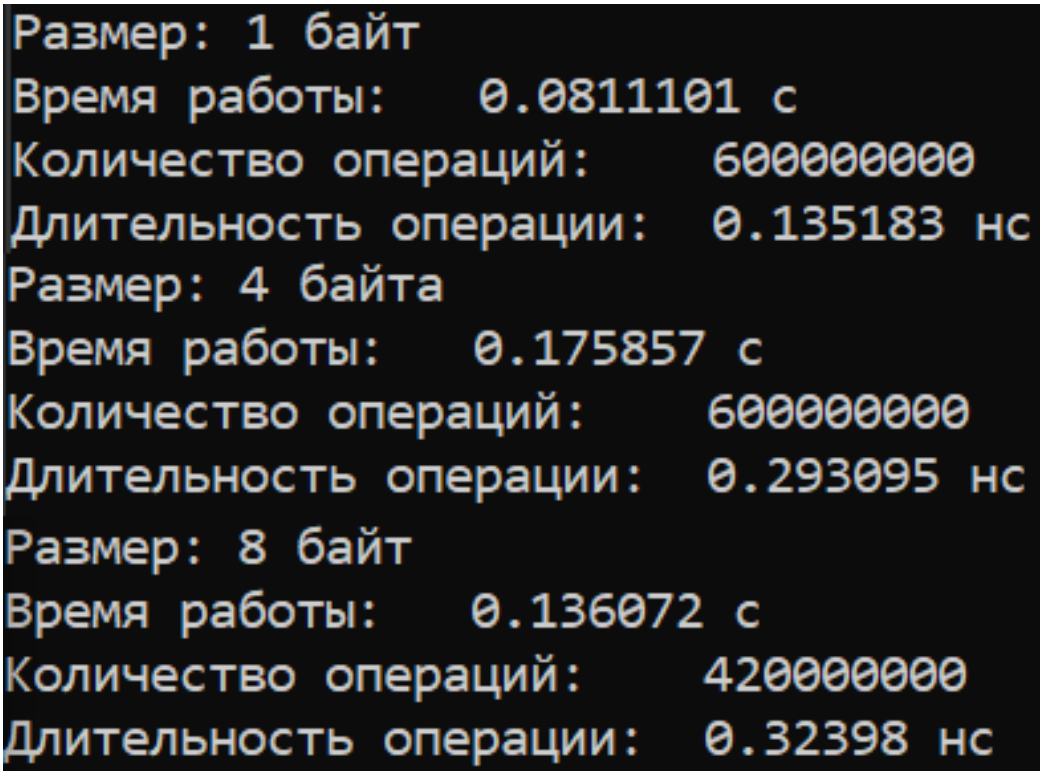
Лабораторная работа 7

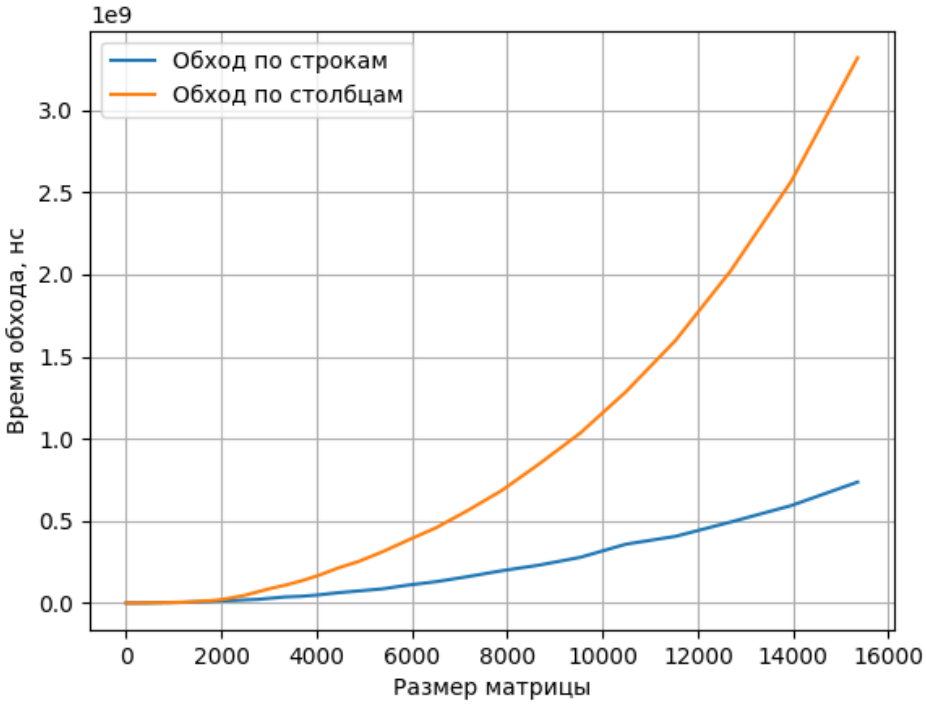
Калашников Михаил, Б03-205

Время, которого мало

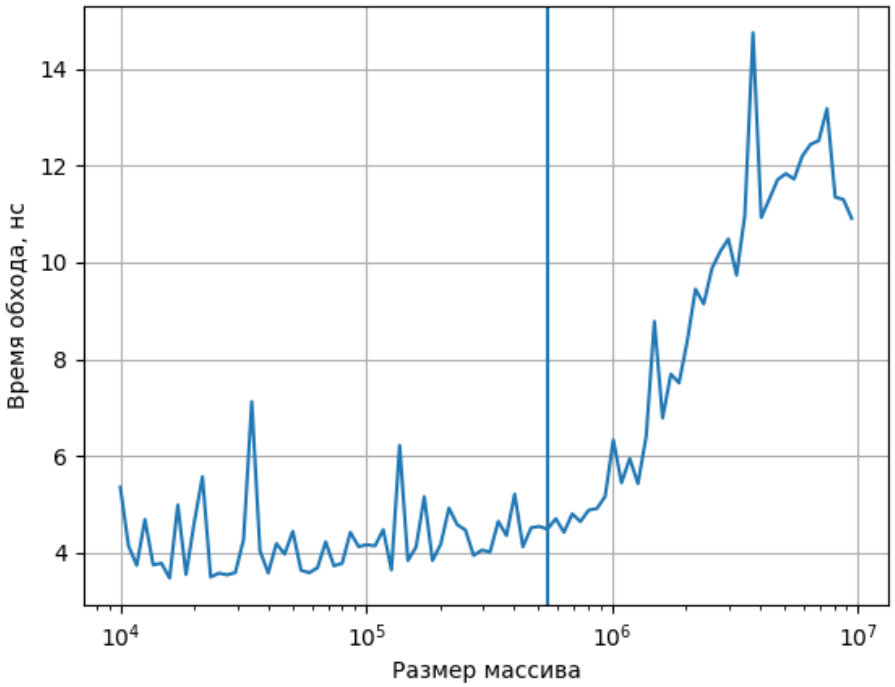
1. Напишем страшненький цикл, каждая итерация которого будет содержать много последовательных операций сложения. Измерим время и получим следующие результаты:



2. Будем обходить матрицу. Как можно заметить, обход по строкам значительно быстрее обхода по столбцам.

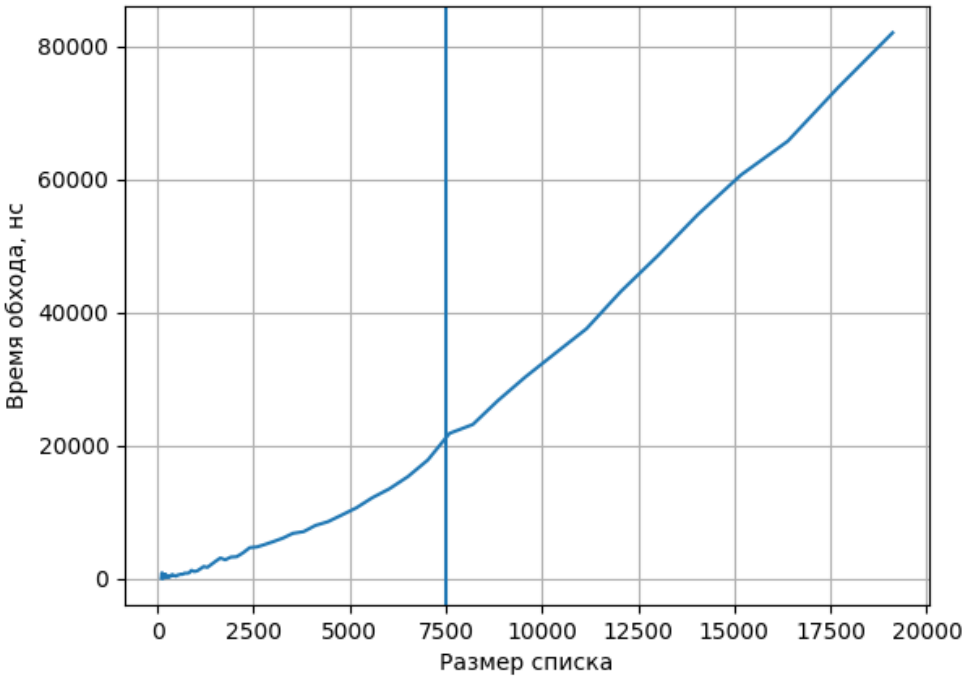


3. Теперь будем обходить одномерный массив интов по заранее сгенерированным случайным индексам. Построим график и заметим резкий скачок. Он происходит когда размер массива начинает превышать размер кэша процессора.



4. Скачок происходит приблизительно при размере массива, равном $5.5 \cdot 10^5$. Это соответствует 2.1 мегабайтам памяти. Согласно диспетчеру задач один из кэшей моего процессора (L2) равен 2 мегабайтам.

5. Со списком такой эффект повторить тяжелее, потому что описанные операции занимают значительно больше времени (и вообще не предусмотрены для списка!). Но я смог заметить незначительное изменение, отраженное на графике.



6. Выделение новой памяти с помощью new занимает примерно 20-23 нс. malloc занимает столько же времени. delete требует 10-12 нс, а free – 9-10 нс.

   Вот пример кода, измеряющего время delete.

```cpp
#include <iostream>
#include <chrono>

int** a;
const int size = 1e6;

int main() {
    a = new int*[size];
    for (int i = 0; i < size; i++) {
        a[i] = new int;
    }

    auto start = std::chrono::high_resolution_clock::now();

    for (int i = 0; i < size; i++) {
        delete a[i];
    }

    auto end = std::chrono::high_resolution_clock::now();
    auto nsec = end - start;

    std::cout << nsec.count() * 1.f / size << " нс" << std::endl;

    return 0;
}
```

7. shared_ptr и unique_ptr создаются в среднем за 140 нс, а за 65 нс.

8. Разыменование указателей (обычных и умных) происходит быстро, примерно за 3-3.5 нс.

   Вот так я измерял это время (в листинге убрал прибавление 23)

```cpp
#include <iostream>
#include <chrono>
#include <memory>

const int size = 1e6;

int main() {

    int* a[size];

    for (int i = 0; i < size; i++) {
        a[i] = new int;
    }

    auto start = std::chrono::high_resolution_clock::now();

    for (int i = 0; i < size; i++) {
        (*(a[i])) += 23;
    }

    auto end = std::chrono::high_resolution_clock::now();
    auto nsec = end - start;

    std::cout << nsec.count() * 1.f / size << " нс" << std::endl;

    return 0;
}
```