Романов А.Э.
А-14м-24
Лабораторная работа №4

Задание

Проверка простоты числа

Алгоритм

Входные данные

Число *a (uint64_t)*, над которым производится проверка.

Выходные данные

- Результат проверки;
- Потенциальный делитель
- Затраченное время (мс)

Алгоритм решения

Последовательный перебор делителей в отрезке $[2, \sqrt{a}]$ и проверка наличия остатка от деления. В случае, если остаток найден – алгоритм останавливается, дальнейшая проверка не требуется.

Тесты

Будем проверять два числа: 2305843009213693951 (простое) и 2305843009213693957 (составное)

2305843009213693951

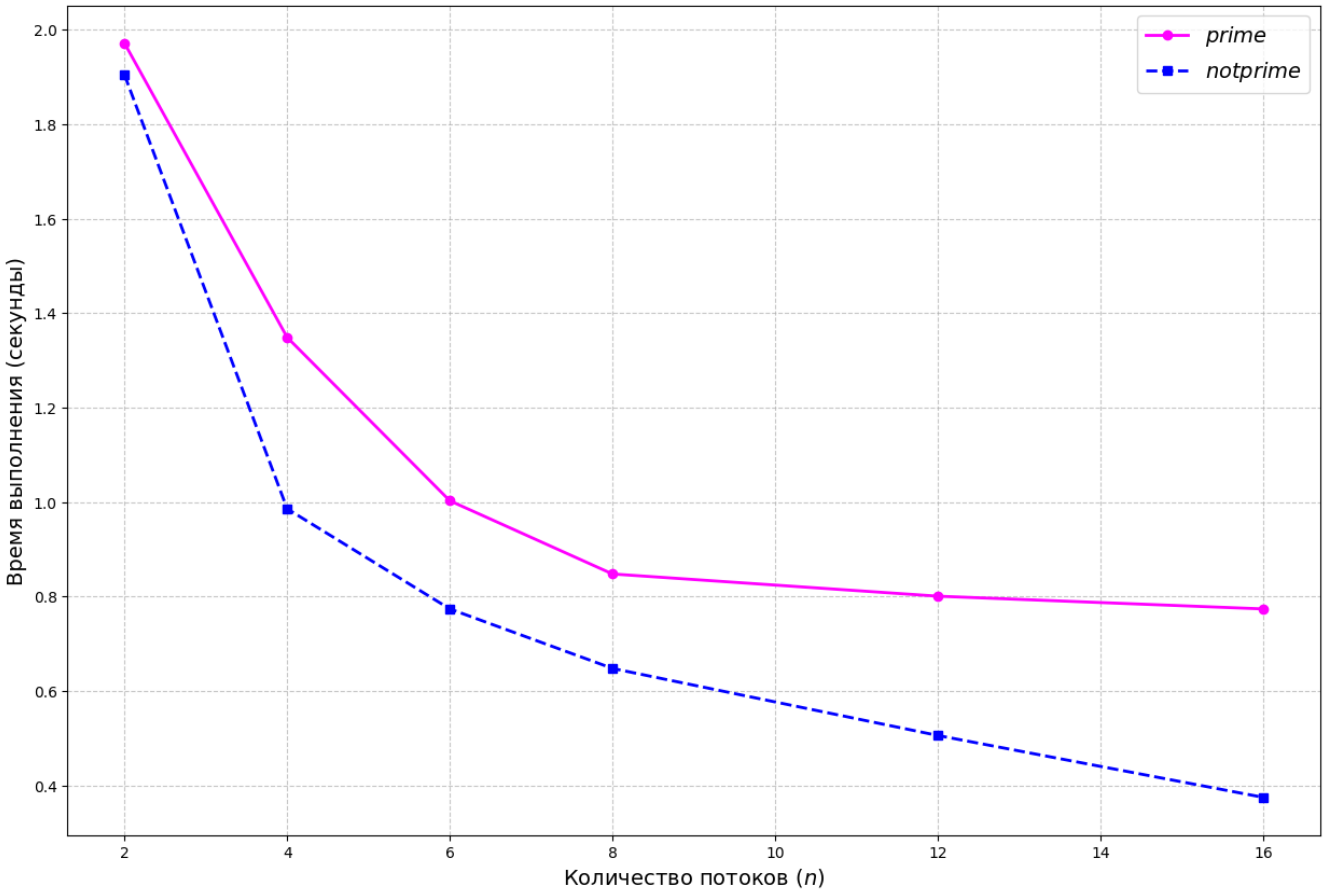
n	2	4	6	8	12	16
t	1.972	1.349	1.003	0.848	0.801	0.774

2305843009213693957

n	2	4	6	8	12	16
t	1.905	0.986	0.774	0.648	0.506	0.375

Графики

Представим полученные результаты на графике



Программный код

```
#include <iostream>
#include <sstream>
#include <thread>
#include <mutex>
#include <cmath>

std::mutex mutex;

uint64_t number, segment;
uint64_t divider = 0;

void find_divider(uint64_t index)
{
    uint64_t start = segment * index + 2;
    uint64_t end = start + segment;

    for (uint64_t div = start; div < end && !divider; div++)
        if (number % div == 0)
        {
            mutex.lock();
            divider = div;
            mutex.unlock();
            break;
        }
}

int main(int argc, char* argv[])
{
    number = 0;
    size_t size = 4;

    // Если аргументы командной строки предоставлены, используем их
    if (argc > 2) {
        std::stringstream sstream;
        sstream << argv[1];
        sstream >> size;
        sstream.clear();
        sstream << argv[2];
        sstream >> number;
    }
    else {
        // Запросить число у пользователя
        std::cout << "Enter number: ";
        std::cin >> number;

        // Запросить количество потоков у пользователя
        std::cout << "Enter number of threads: ";
```

```
        std::cin >> size;
    }

    segment = (sqrt(number) - 2) / size;

    std::thread* threads = new std::thread[size];

    clock_t timeStart = clock();
    for (int i = 0; i < size; i++)
        threads[i] = std::thread(find_divider, i);
    for (int i = 0; i < size; i++)
        threads[i].join();
    clock_t timeEnd = clock();

    if (divider != 0)
        std::cout << number << " is not prime!\nDivider: " << divider << std::endl;
    else
        std::cout << number << " is prime!\n";
    std::cout << "Time: " << double(timeEnd - timeStart) / CLOCKS_PER_SEC << " seconds\n";

    delete[] threads;
    return 0;
}
```