Министерство образования и науки Российской Федерации

Национальный исследовательский университет «МИЭТ»

Институт системной и программной инженерии и

информационных технологий

Технология программирования Open MP

Лабораторная работа №6

Вариант 1

Выполнил студент группы ПИН-31Д: Ельяшевич Н.А.

Проверилa: Можжухина А.В.

Зеленоград, 2024 г.

Задание 1. Модифицировать программы, составленные в Л.Р. №№3,4, используя переменные семафорного типа. Измерять время работы программы для тех же значений параметров, что были использованы при выполнении Л.Р. №3.

Модифицированная программа из Л.Р.№3:

#include <iostream>

#include <omp.h>

#include <vector>

#include <chrono>

**int** **main**() {

**using** **namespace** std;

setlocale(LC\_ALL, "Russian");

**const** **int** N = **1000000**;

std::vector<**int**> A(N, **2**);

std::vector<**int**> B(N, **3**);

**int** result = **1**;

**omp\_lock\_t** lock;

omp\_init\_lock(&lock);

**auto** start\_time = std::chrono::high\_resolution\_clock::now();

#pragma omp parallel for

**for** (**int** i = **0**; i < N; ++i) {

**int** local\_result = **1**;

**if** ((A[i] + B[i]) != **0**) {

local\_result \*= (A[i] + B[i]);

}

omp\_set\_lock(&lock);

result \*= local\_result;

omp\_unset\_lock(&lock);

}

**auto** end\_time = std::chrono::high\_resolution\_clock::now();

std::chrono::duration<**double**> duration = end\_time - start\_time;

std::cout << "Результат произведения ненулевых значений Ai + Bi: " << result << std::endl;

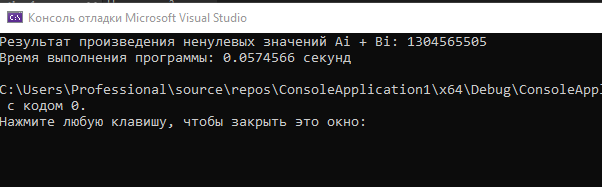
std::cout << "Время выполнения программы: " << duration.count() << " секунд" << std::endl;

omp\_destroy\_lock(&lock);

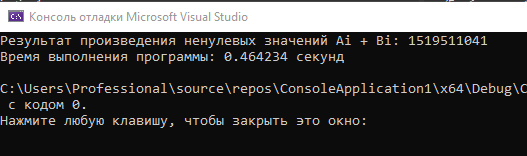
**return** **0**;

}

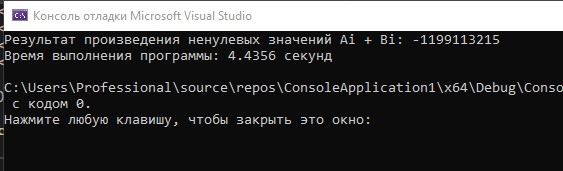
N = 1000000



N = 10000000



N = 100000000



Модифицированная программа из Л.Р.№4:

#include <iostream>

#include <omp.h>

#include <vector>

#include <chrono>

**int** **main**() {

**using** **namespace** std;

setlocale(LC\_ALL, "Russian");

**const** **int** N = **1000000**;

std::vector<**int**> A(N, **2**);

std::vector<**int**> B(N, **3**);

**int** result = **1**;

**int** local\_result = **1**;

**omp\_lock\_t** lock;

omp\_init\_lock(&lock);

**auto** start\_time = std::chrono::high\_resolution\_clock::now();

#pragma omp parallel private(local\_result)

{

local\_result = **1**;

#pragma omp for

**for** (**int** i = **0**; i < N; ++i) {

**if** ((A[i] + B[i]) != **0**) {

local\_result \*= (A[i] + B[i]);

}

}

omp\_set\_lock(&lock);

result \*= local\_result;

omp\_unset\_lock(&lock);

}

**auto** end\_time = std::chrono::high\_resolution\_clock::now();

std::chrono::duration<**double**> duration = end\_time - start\_time;

std::cout << "Результат произведения ненулевых значений Ai + Bi: " << result << std::endl;

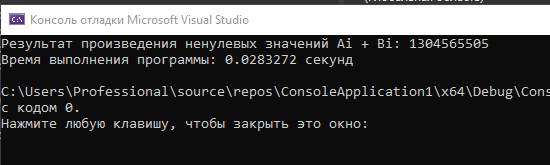
std::cout << "Время выполнения программы: " << duration.count() << " секунд" << std::endl;

omp\_destroy\_lock(&lock);

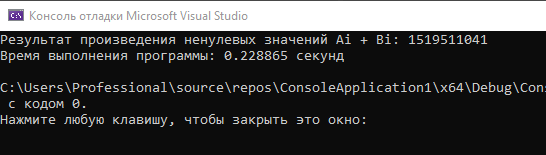
**return** **0**;

}

N = 1000000



N = 10000000



N = 100000000

