МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра информационных технологий**

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1**

**по дисциплине  
 «Параллельное и низкоуровневое программирование»**

Выполнил студент группы 25/2                                       А.А. Козин

Направление подготовки  02.03.03  Математическое обеспечение и администрирование информационных систем

Курс    2

Отчет принял доктор физико-математических наук, профессор                                                                                       А.И. Миков

Краснодар

2022 г.

**Задание 1**: Скалярное произведение 2-х векторов.

**Решение**: создадим 2 динамических массива a и b для векторов и c и d для результатов скалярного произведения этих векторов. Я задаю случайные значения в вектора a и b, а c и d заполняю нулями. Далее передаю в dot\_product1 и dot\_product2 массивы с векторами и кол-во векторов. Эти 2 функции параллельно вычисляют скалярное произведение. Функция dot\_product3 вычисляет последовательно. В main формируем объекты thread first, second, third. Используем метод join, который блокирует вызывающий поток (в нашем случае — поток main) до тех пор, пока first, second, third не выполнят свою работу. После вычисления считаем время в миллисекундах и выводим в консоль.

**Текст программы:**

#include <iostream>

#include <ctime>

#include <thread>

#include <chrono>

using namespace std;

using namespace std::chrono;

void dot\_product1(int a[], int b[], int c[],int n) {

for (int i = 0; i < n / 2; i++)

c[i] += a[i] \* b[i];

}

void dot\_product2(int a[], int b[], int c[],int n) {

for (int i = n / 2; i < n; i++)

c[i] += a[i] \* b[i];

}

void dot\_product3(int a[], int b[], int d[],int n) {

for (int i = 0; i < n; i++)

d[i] += +a[i] \* b[i];

}

void main() {

setlocale(LC\_ALL, "Ru");

srand(time(0));

cout << "Введите размер: "; int n; cin >> n;

auto t0 = high\_resolution\_clock::now();

int\* a = new int[n]; int\* b = new int[n];

int\* c = new int[n]; int\* d = new int[n];

for (int i = 0; i < n; i++) {

a[i] = rand() % 10;

b[i] = rand() % 10;

c[i] = 0; d[i] = 0;

}

thread first(dot\_product1, a, b, c,n); first.join();

thread second(dot\_product2, a, b, c,n);second.join();

auto t1 = high\_resolution\_clock::now();

thread third(dot\_product3, a, b, d,n); third.join();

auto t2 = high\_resolution\_clock::now();

int sum = 0;

for (int i = 0; i < n; i++) {

sum += c[i];

}

cout << "sum: " << sum << endl << endl;

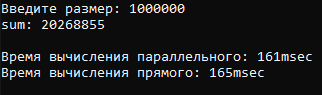
cout << "Время вычисления параллельного: " << duration\_cast<milliseconds>(t1 - t0).count() << "msec\n";

cout << "Время вычисления прямого: " << duration\_cast<milliseconds>(t2 - t0).count() << "msec\n";

delete[] a, b, c, d;

}

**Программа выдает в консоли результаты в следующем виде:**



**Задание 2**: Умножение квадратных матриц.

**Решение**: создадим 3 квадратных матрицы для перемножения и заполним их рандомно. Напишем функции для распараллеливания (firstplace & secondplace) и последовательного (place) методов. Передав туда матрицы перемножим их и выведем затраченное время в консоль.

**Текст программы:**

#include <iostream>

#include <ctime>

#include <thread>

#include <chrono>

using namespace std;

using namespace std::chrono;

void firstplace(int\*\* FirstArray, int\*\* SecondArray, int\*\* ThirdArray, int n) {

for (int i = 0; i < n / 2; i++){

for (int j = 0; j < n / 2; j++){

for (int k = 0; k < n / 2; k++) {

ThirdArray[i][j] += FirstArray[i][k] \* SecondArray[k][j];

}

}

}

}

void secondplace(int\*\* FirstArray, int\*\* SecondArray, int\*\* ThirdArray, int n) {

for (int i = n / 2; i < n; i++){

for (int j = n / 2; j < n; j++){

for (int k = n / 2; k < n; k++) {

ThirdArray[i][j] += FirstArray[i][k] \* SecondArray[k][j];

}

}

}

}

void place(int\*\* FirstArray, int\*\* SecondArray, int\*\* FourArray, int n) {

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

for (int k = 0; k < n; k++) {

FourArray[i][j] += FirstArray[i][k] \* SecondArray[k][j];

}

}

}

}

void main() {

setlocale(LC\_ALL, "Ru");

srand(time(0));

auto t0 = high\_resolution\_clock::now();

cout << "Введите размер:"; int n; cin >> n;

int\*\* FirstArray; FirstArray = new int\* [n];

int\*\* SecondArray; SecondArray = new int\* [n];

int\*\* ThirdArray; ThirdArray = new int\* [n];

int\*\* FourArray; FourArray = new int\* [n];

for (int i = 0; i < n; i++) {

FirstArray[i] = new int[n];

SecondArray[i] = new int[n];

ThirdArray[i] = new int[n];

FourArray[i] = new int[n];

}

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

FirstArray[i][j] = rand() % 15 + 3;

SecondArray[i][j] = rand() % 15 + 3;

ThirdArray[i][j] = 0;

FourArray[i][j] = 0;

}

}

thread first(firstplace, FirstArray, SecondArray, ThirdArray, n);

first.join();

thread second(secondplace, FirstArray, SecondArray, ThirdArray, n);

second.join();

auto t1 = high\_resolution\_clock::now();

thread third(place, FirstArray, SecondArray, FourArray, n);

third.join();

auto t2 = high\_resolution\_clock::now();

cout << "Время вычисления параллельного: " << duration\_cast<milliseconds>(t1 - t0).count() << "msec\n";

cout << "Время вычисления прямого: " << duration\_cast<milliseconds>(t2 - t0).count() << "msec\n";

for (int i = 0; i < n; i++) {

delete[] FirstArray[i];

delete[] SecondArray[i];

delete[] ThirdArray[i];

delete[] FourArray[i];

}

delete[] FirstArray, SecondArray, ThirdArray, FourArray;

}

**Программа выдает в консоли результаты в следующем виде:**

