Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Пермский национальный исследовательский

политехнический университет»

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6**

по дисциплине **«Технологии блокчейн и распределенные информационные системы»**

**Выполнил** студент гр. АСУ-20-1б

Чувашев Максим Алексеевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(подпись студента)*

**Проверил** доцент кафедры ИТАС

Щапов Владислав Алексеевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(оценка, дата, подпись преподавателя)*

Пермь, 2024

# 1 Анализ предметной области

## 1.1 Постановка задачи

С использованием языка C++ и библиотеки oneAPI Threading Building Blocks (oneTBB) разработать программу для решения СЛАУ методом Гаусса.

## 1.2 Инструменты разработки

Для выполнения лабораторной работы были использованы следующие инструменты:

1. редактор CLion;

# 2 Технология разработки

На листинге 1 представлена реализация «прямого хода» умножения СЛАУ методом Гаусса с использованием функции «parallel\_for» из библиотеки oneTBB.

// Вычесть текущую строку из остальных

parallel\_for(blocked\_range<int>(i + 1, n), [&](const blocked\_range<int>& r) {

for (int k = r.begin(); k != r.end(); ++k) {

double factor = A[k][i];

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

A[k][j] -= factor \* A[i][j];

}

b[k] -= factor \* b[i];

A[k][i] = 0;

}

});

Листинг 1 – «Прямой ход»

На листинге 2 представлена реализация «обратного хода» умножения СЛАУ методом Гаусса.

// Обратный ход

for (int i = n - 1; i >= 0; --i) {

for (int j = i - 1; j >= 0; --j) {

b[j] -= A[j][i] \* b[i];

A[j][i] = 0;

}

}

Листинг 2 – «Обратный ход»

# 3 Результаты работы

Результаты выполнения задания лабораторной работы проиллюстрированы на рисунке 1.

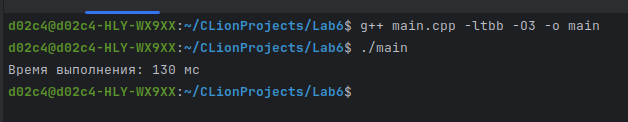


Рисунок 1 – Результат работы кода

ПРИЛОЖЕНИЕ А

#include <iostream>

#include <vector>

#include <random>

#include <oneapi/tbb.h>

// #include <tbb/parallel\_for.h>

// #include <tbb/blocked\_range.h>

#include <chrono>

using namespace std;

using namespace tbb;

// Функция для генерации СЛАУ

void generate\_system(vector<vector<double>>& A, vector<double>& b, int n) {

random\_device rd;

mt19937 gen(rd());

uniform\_real\_distribution<> dis(-10, 10);

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

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

A[i][j] = dis(gen);

}

b[i] = dis(gen);

}

}

// Функция для вывода СЛАУ

void print\_system(const vector<vector<double>>& A, const vector<double>& b) {

int n = A.size();

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

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

cout << A[i][j] << "\*x" << j;

if (j < n - 1) cout << " + ";

}

cout << " = " << b[i] << "\n";

}

}

// Функция для решения СЛАУ методом Гаусса

void solve\_system(vector<vector<double>>& A, vector<double>& b) {

int n = A.size();

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

// Нормализовать текущую строку

for (int j = i + 1; j < n; ++j) {

A[i][j] /= A[i][i];

}

b[i] /= A[i][i];

A[i][i] = 1;

// Вычесть текущую строку из остальных

parallel\_for(blocked\_range<int>(i + 1, n), [&](const blocked\_range<int>& r) {

for (int k = r.begin(); k != r.end(); ++k) {

double factor = A[k][i];

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

A[k][j] -= factor \* A[i][j];

}

b[k] -= factor \* b[i];

A[k][i] = 0;

}

});

}

// Обратный ход

for (int i = n - 1; i >= 0; --i) {

for (int j = i - 1; j >= 0; --j) {

b[j] -= A[j][i] \* b[i];

A[j][i] = 0;

}

}

}

int main() {

int n = 1000; // Размер СЛАУ

vector<vector<double>> A(n, vector<double>(n));

vector<double> b(n);

// Генерация СЛАУ

generate\_system(A, b, n);

// Вывод изначальных данных

//cout << "Исходная СЛАУ:\n";

//print\_system(A, b);

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

// Решение СЛАУ

solve\_system(A, b);

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

auto duration = std::chrono::duration\_cast<std::chrono::milliseconds>(t2 - t1);

// Вывод результата

//cout << "\nРешение СЛАУ:\n";

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

// cout << "x" << i << " = " << b[i] << "\n";

//}

cout << "Время выполнения: " << duration.count() << " мс" << endl;

return 0;

}