**Лабораторная работа №1**

**Распараллеливание по данным для циклов**

**Цель:** изучить основные особенности распараллеливания данных по циклам в OpenMP на примере использования в рамках языка С++.

**Лабораторные задания** (№ варианта = 1)

**Задание 1.**

* Написать на C++ программу в соответствии с вариантом задания, при этом программы должны оптимально использовать предлагающиеся вычислительные ресурсы. Для реализации многопоточности использовать технологию OpenMP.
* Измерять время работы программы для различных значений параметров. **Результаты занести в отчёт.**
* Также, при запуске программ, запустить диспетчер задач Windows, в нём установить счётчик потоков, и просмотреть сколько потоков выполнения использует процесс, соответствующий запущенной программе. **Результаты также занести в отчёт.**

|  |  |
| --- | --- |
| **Вариант** | **Задание: написать программы, демонстрирующие работу следующей функции:** |
| **1** | Дана матрица из MxN натуральных (ненулевых) элементов (задаются случайно).  Написать программу, считающую количество семёрок в десятеричной записи числа всех попарных сумм элементов для каждой строки. |

**Задание 2.**

* Модифицировать программу, составленную по Заданию 1, убрав возможность параллельного использования вычислительных ресурсов.
* Измерять время работы программы для тех же значений параметров, что были использованы при выполнении Задания 1. **Результаты сравнить и занести в отчёт.**

/\*

Дана матрица из MxN натуральных (ненулевых) элементов (задаются случайно).

Написать программу, считающую количество семёрок в десятеричной записи числа всех попарных сумм элементов

для каждой строки.

\*/

#include <omp.h>

#include <iostream>

#include <ctime>

#include <string>

#include <algorithm>

#include <iomanip>

int count7(int n, int m, int\*\* arr) {

int i, j, k;

int count = 0;

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

for (j = 0; j < m - 1; j++) {

for (k = j + 1; k < m; k++) {

std::string twosum = std::to\_string(arr[i][j] + arr[i][k]);

count = count + std::count(twosum.begin(), twosum.end(), '7');

}

}

}

return count;

}

int count7\_omp(int n, int m, int\*\* arr) {

int i, j, k;

int count = 0;

#pragma omp parallel for shared(arr) private(i, j, k) reduction(+:count)

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

for (j = 0; j < m - 1; j++) {

for (k = j + 1; k < m; k++) {

std::string twosum = std::to\_string(arr[i][j] + arr[i][k]);

count = count + std::count(twosum.begin(), twosum.end(), '7');

}

}

}

return count;

}

void test(int n, int m, int\*\* arr) {

double t1, t2, count;

//Расчет попарных сумм и расчет в их записи количества семёрок

t1 = omp\_get\_wtime();

count = count7(n, m, arr);

t2 = omp\_get\_wtime();

std::cout << "Тест без использования OpenMP" << std::endl;

std::cout << "Количество '7' в записи сумм: " << count << std::endl;

std::cout << "Время исполнения основного вычислительного блока: " << t2 - t1 << std::endl;

}

void test\_omp(int n, int m, int\*\* arr) {

double t1, t2, count;

//Расчет попарных сумм и расчет в их записи количества семёрок

t1 = omp\_get\_wtime();

count = count7\_omp(n, m, arr);

t2 = omp\_get\_wtime();

std::cout << "Тест с использованием OpenMP" << std::endl;

std::cout << "Количество '7' в записи сумм: " << count << std::endl;

std::cout << "Время исполнения основного вычислительного блока: " << t2 - t1 << std::endl;

}

int main() {

srand((unsigned int)time(NULL)); // автоматическая рандомизация

system("chcp 1251");

double t1, t2;

int m, n, count;

//Ввод количества строк и столбцов массива

std::cout << "Введите количество строк (n) и столбцов (m):" << std::endl;

std::cin >> n >> m;

//Создание рандомного массива

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

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

arr[i] = new int[m];

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

arr[i][j] = rand()%100;

}

}

//Вывод массива

std::cout << "Рандомный массив m на n" << std::endl;

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

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

std::cout << std::setw(4) << arr[i][j] << " ";

}

std::cout << "\n";

}

//Вывод попарных сумм каждой строки

std::cout << "Попарные суммы в каждой строке" << std::endl;

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

for (int j = 0; j < m - 1; j++) {

for (int k = j + 1; k < m; k++) {

std::string twosum = std::to\_string(arr[i][j] + arr[i][k]);

std::cout << std::setw(4) << twosum << " ";

}

}

std::cout << "\n";

}

//Проведение тестов без и с использованием OpenMP

test(n, m, arr);

test\_omp(n, m, arr);

return 0;

}

|  |  |  |
| --- | --- | --- |
| n m | Время без OpenMP, сек | Время с OpenMP, сек |
| 4 5 | 0.0002927 | 0.0001482 |
| 10 10 | 0.002773 | 0.0014936 |
| 30 50 | 0.15055 | 0.048433 |
| 60 70 | 0.477769 | 0.165911 |
| 100 100 | 1.65845 | 0.575269 |

