**Лабораторная работа №3**

**Использование редукции**

**Цель:** изучить основные особенности использования редукции в OpenMP на примере использования в рамках языка С++.

**Контрольные вопросы**

1. Как определяются общие и локальные переменные потоков?

Параметр shared определяет переменные, которые будут общими для всех потоков. Параметр private указывает переменные, для которых в каждом потоке будут созданы локальные копии – они будут доступны только внутри каждого потока в отдельности (значения локальных переменных потока недоступны для других потоков)

По умолчанию, все параметры являются общими

1. Что понимается под операцией редукции?

Редукция - Задание коллективной операции

#include <omp.h>

#include <iostream>

#include <time.h>

using namespace std;

const int N = 10000000;

#define PRINT\_MATRIX false

#define PRINT\_RESULT false

#define SHOW\_ITERATIONS false

void sequence\_func()

{

int i;

int\* A;

int\* B;

int\* C;

double result = 0.1;

A = new int[N];

B = new int[N];

C = new int[N];

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

A[i] = rand() % 10 + 1;

B[i] = rand() % 10 + 1;

C[i] = rand() % 10 + 1;

}

#pragma region prints

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

if (PRINT\_MATRIX)

cout << A[i] << " ";

}

if (PRINT\_MATRIX) {

cout << " \n";

}

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

if (PRINT\_MATRIX)

cout << B[i] << " ";

}

if (PRINT\_MATRIX)

cout << " \n";

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

if (PRINT\_MATRIX)

cout << C[i] << " ";

}

if (PRINT\_MATRIX)

cout << " \n";

#pragma endregion

#pragma region code

clock\_t start1 = clock();

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

if (A[i] % 2 == 0) {

result += (double)B[i] / (double)C[i]; // здесь и далее использую “+” (а не \*) для наглядного отображения результата (не inf)

if (SHOW\_ITERATIONS)

cout << result << endl;

}

else

{

result += (double)A[i] + B[i];

if (SHOW\_ITERATIONS)

cout << result << endl;

}

}

#pragma endregion

clock\_t end1 = clock();

printf("\nTime of operating without MP: %f", difftime(end1, start1));

cout << "\nResult: " << result << endl;

result = 0.1;

clock\_t start = clock();

#pragma omp parallel shared(A,B,C)

{

#pragma omp for private(i) reduction(+:result)

for (i = 0; i < N; i++)

{

if (A[i] % 2 == 0)

{

result += (double)B[i] / (double)C[i];

if (SHOW\_ITERATIONS)

cout << result << endl;

}

else

{

result += (double)A[i] + B[i];

if (SHOW\_ITERATIONS)

cout << result << endl;

}

}

}

clock\_t end = clock();

printf("\nTime of operating with MP: %f", difftime(end, start));

cout << "\nResult: " << result << endl;

}

int main() {

int i;

sequence\_func();

return 0;

}

Результат выполненения:

Time of operating without MP: 47.000000

Result: 6.0538e+07

Time of operating with MP: 16.000000

Result: 6.0538e+07