Технологии параллельного программирования

Модуль 1. Введение. Технология OpenMP

Семинар 3

**Общие задания**

**Задание 1. Директива OpenMP parallel , опции private и num\_threads. Поэкспериментировать с разными значениями num\_threads**

#include <stdio.h>

#include <omp.h>

#include <locale>

int main()

{

setlocale(LC\_ALL, "Russian");

int n = 1;

printf("n в последовательной области (начало): %d\n", n);

#pragma omp parallel private(n) num\_threads(6)

{

printf("Значение n в потоке (на входе): %d\n", n);

n = omp\_get\_thread\_num(); // Присваеваем n номер текущего потока (нити)

printf("Значение n в потоке (на выходе): %d\n", n);

}

n = 3;

printf("n в последовательной области (конец): %d\n", n);

return 0;

}

**Задание 2. Директива OpenMP parallel, опция firstprivate. Поэкспериментировать с разными значениями инициализации n**

#include <stdio.h>

#include <omp.h>

#include <locale>

int main()

{

setlocale(LC\_ALL, "Russian");

int n = 1;

printf("Значение n в начале: %d\n", n);

#pragma omp parallel firstprivate(n)

{

printf("Значение n в потоке (на входе): %d\n", n);

n = omp\_get\_thread\_num(); // Присвоим переменной n порядковый номер потока

printf("Значение n в потоке (на выходе): %d\n", n);

}

printf("Значение n в конце: %d\n", n);

}

**Задание 3. Директива OpenMP parallel, опция reduction. Поэкспериментировать с разными значениями оператора опции, разным начальным значением x и разным значением операции над переменной x.**

#include <iostream>

#include "omp.h"

#include <string>

#include <locale>

using namespace std;

int main()

{

setlocale(LC\_ALL, "Russian");

int x = 0;

printf("x в последовательной области (начало): %d\n", x);

#pragma omp parallel reduction(+:x) num\_threads(30)

{

printf("Значение x в потоке (на входе): %d\n", x);

x += 1;

printf("Значение x в потоке (на выходе): %d\n", x);

}

cout << "x = " << x << endl;

return 0;

}

**Задания для самостоятельного выполнения**

1. Выполнить сложение двух векторов, содержащих от 50 млн. вещественных значений (предварительно заполнить векторы случайными значениями, предусмотреть выделение и освобождение областей памяти для векторов с помощью функций стандартной библиотеки С++). Вычислить и вывести на экран время выполнения сложения, с использованием функций библиотеки OpenMP. Вывести на экран размер и любой элемент результирующего вектора.
2. Выполнить задание 1, используя директиву Parallel с опцией Shared (для векторов). Сравнить полученное время с временем в задаче 1, сделать выводы.