**Отчет лабораторной работы №5**

**Тема «** **Технология OpenMP. Общие и частные переменные»**

**Султанов Шакир КА**

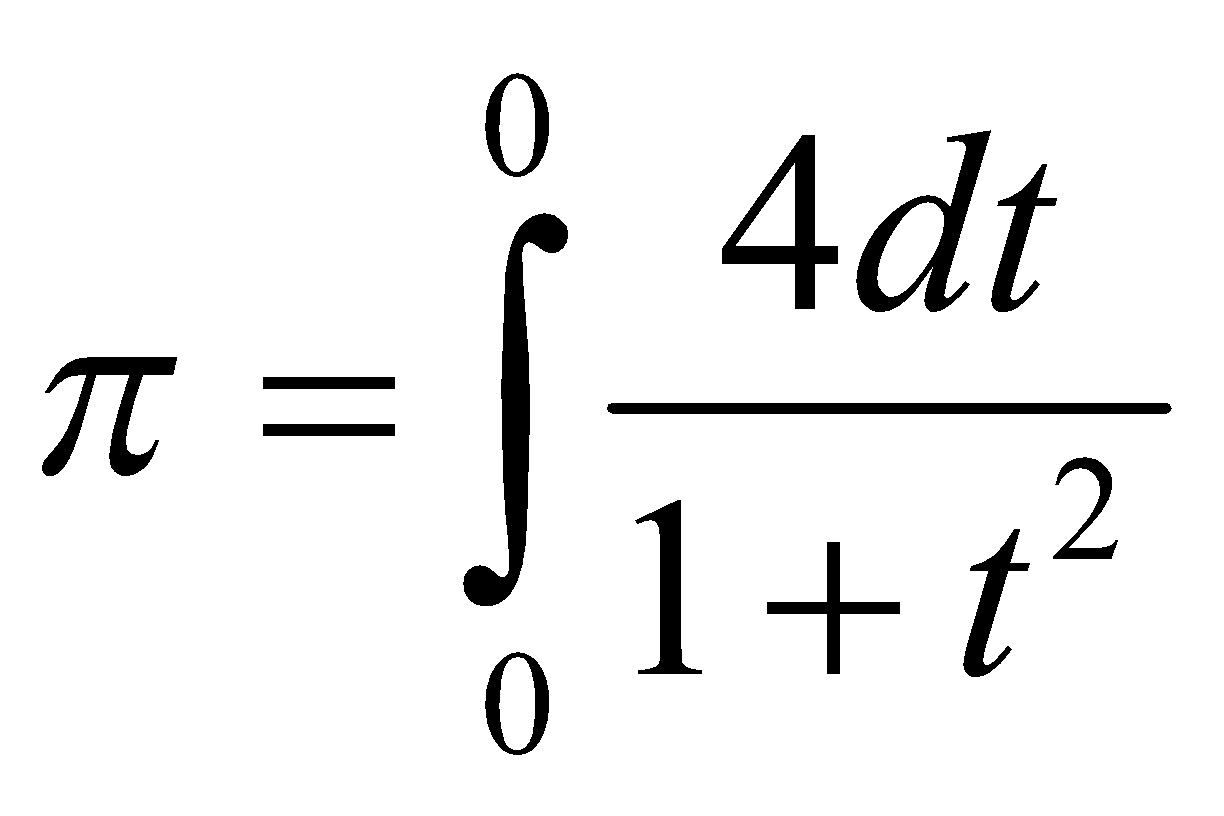
**Тема «Технология OpenMP. Общие и частные переменные»**

Студенты должны знать:

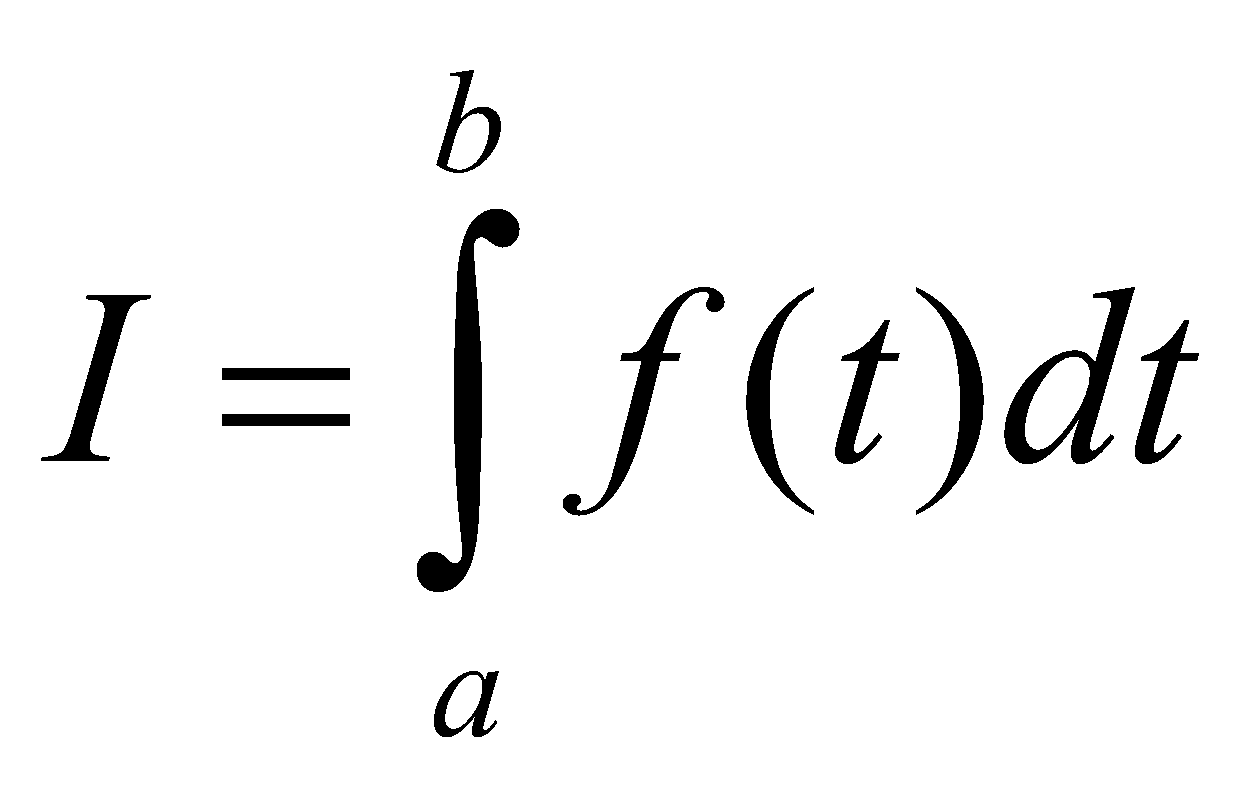
1. Директивы OpenMP.
2. Общие и частные переменные.
3. Функции работы с таймерами.

**Задание**

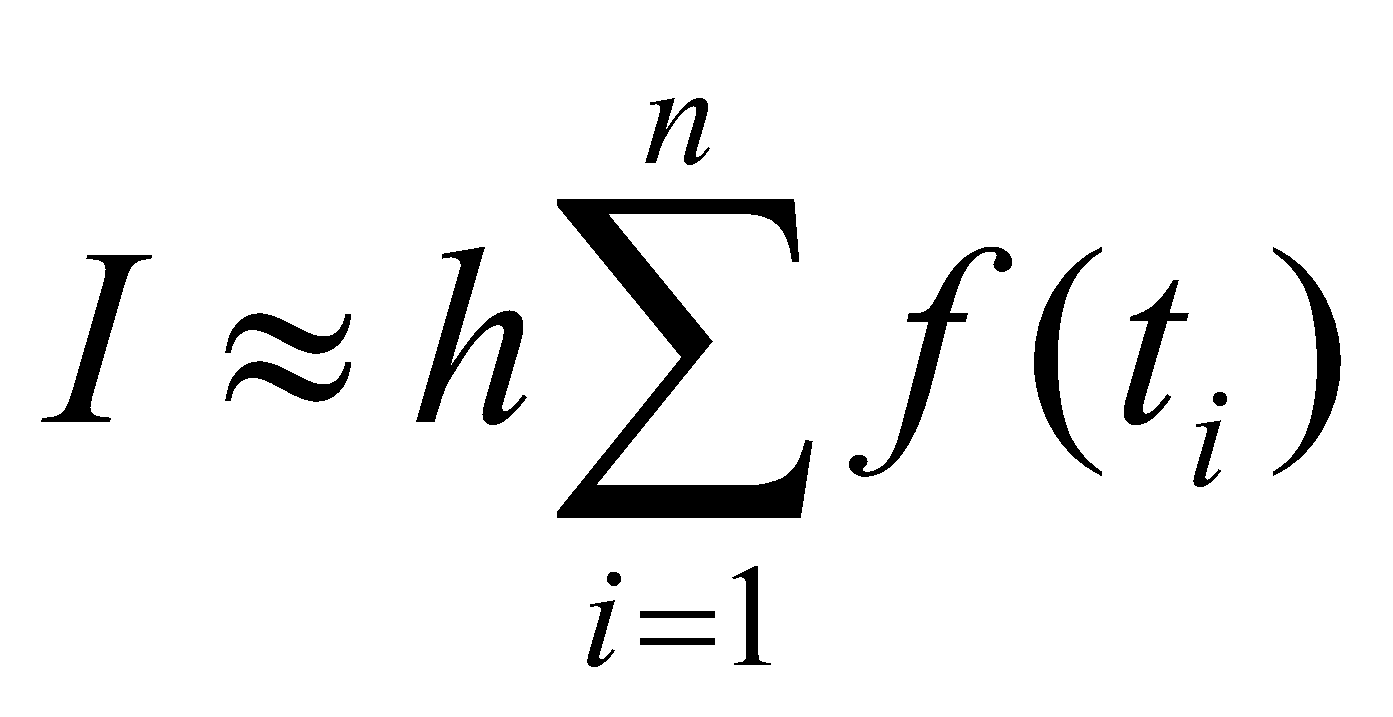
Известно, что

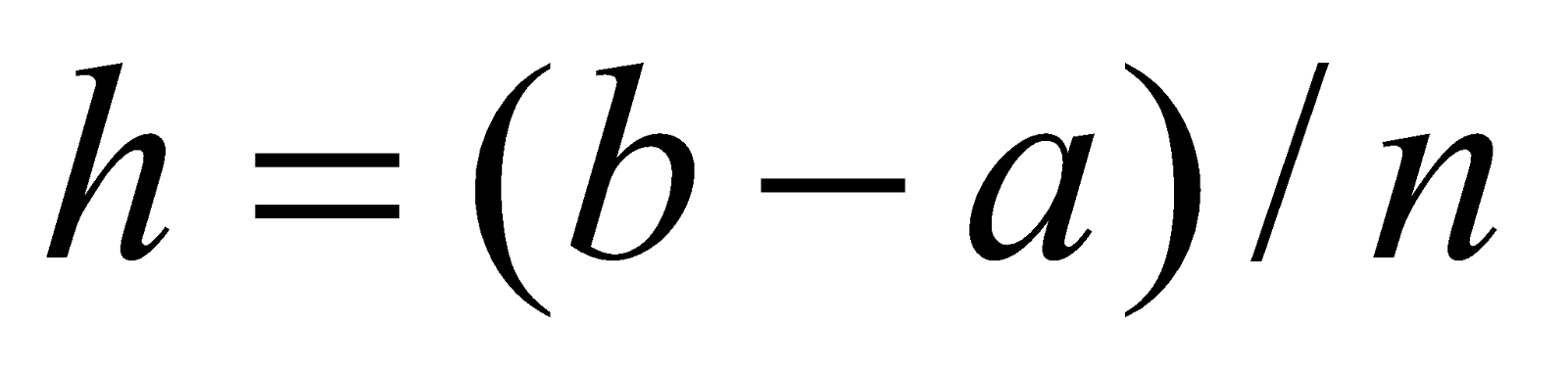
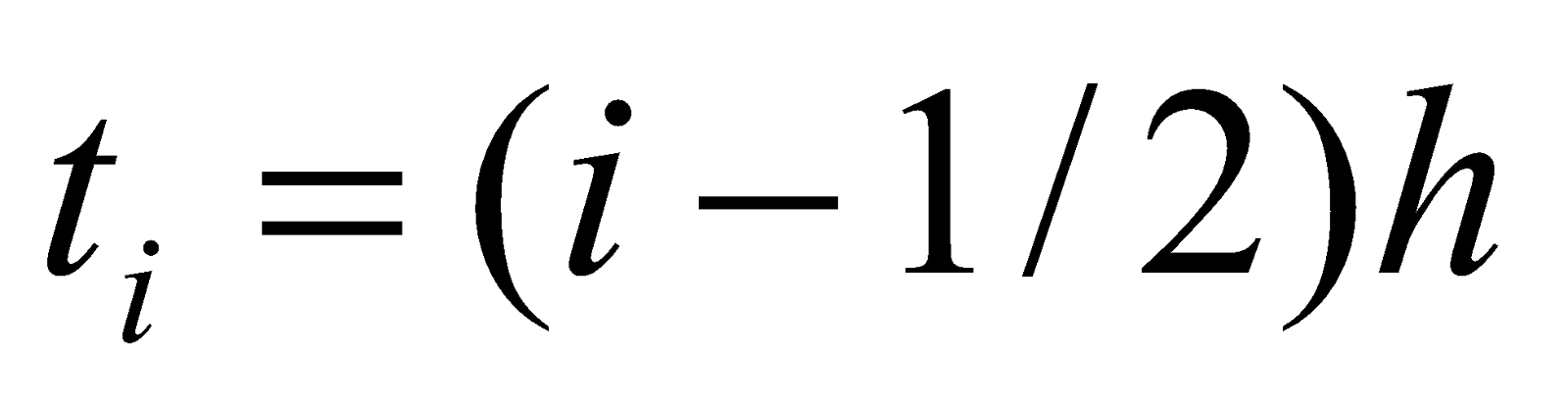
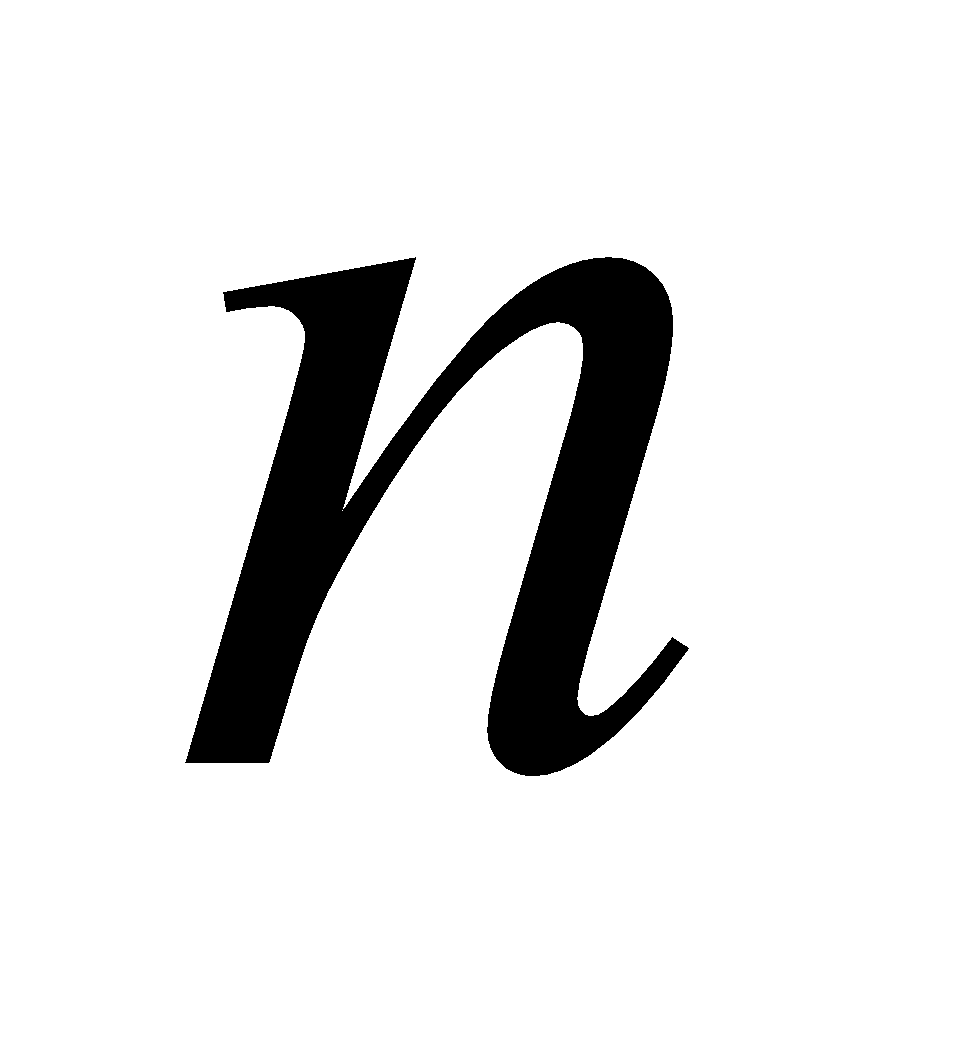
.

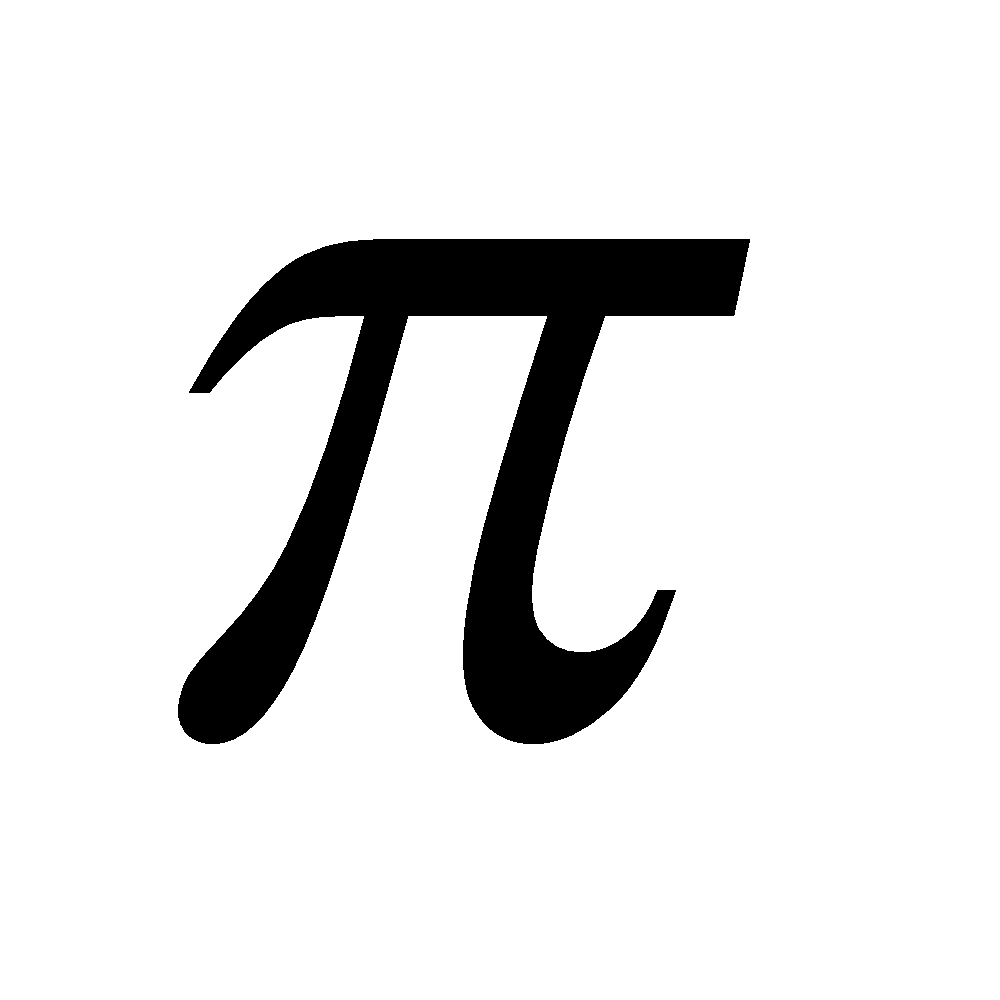
Определенный интеграл



Приближенно вычисляется, например, с помощью составной формулы трапеций:

,

где , . Чем дольше  , тем точнее вычисляется интеграл.

Ниже приведена последовательная программа вычисления числа  с помощью численного интегрирования.

**#include <stdio.h>**

**// Вычисление числа pi**

**double f(double t) {return(4.0/(1.0+t\*t));}**

**int main()**

**{**

**double t, t, sum, pi;**

**int i;**

**long int n = 100000;**

**h = 1.0/n;**

**sum = 0.0;**

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

**{**

**t = h\*(i-0.5);**

**sum = sum + f(t);**

**}**

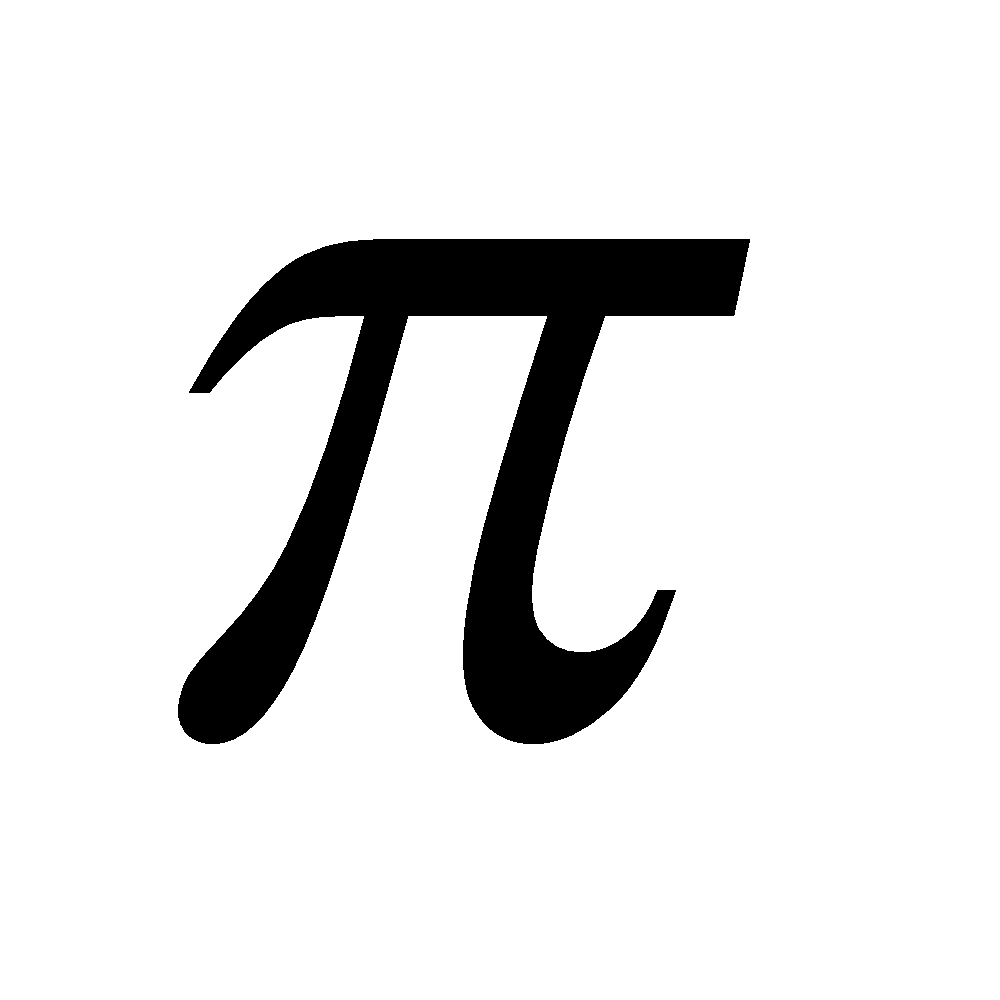
**pi = h\*sum;**

**printf("pi = %f\n", pi);**

**}**

Используя технологию OpenMP, написать параллельную версию этой программы.

Провести вычислительный эксперимент и заполнить таблицу 1.

Таблица 1. Время вычисления числа 

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n | Последовательная  программа | Параллельная программа (количество потоков) | | |
| 2 | 4 | 6 |
| 10^4 |  |  |  |  |
| 10^6 |  |  |  |  |
| 10^8 |  |  |  |  |
| 10^10 |  |  |  |  |
| 10^12 |  |  |  |  |

**КОД**

**#include <stdio.h>**

**#include "omp.h"**

**// Вычисление числа pi**

**double f(double y) {return(4.0/(1.0+y\*y));}**

**int main()**

**{**

**int \_threads;**

**printf("Введите кол-во потоков:");**

**scanf("%d", &\_threads);**

**long int n = 10000;**

**printf("Введите n:");**

**scanf("%ld", &n);**

**double w, x, sum, pi;**

**long int i;**

**w = 1.0/n;**

**sum = 0.0;**

**double f\_time,s\_time;**

**s\_time = omp\_get\_wtime();**

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

**{**

**x = w\*(i-0.5);**

**sum = sum + f(x);**

**}**

**pi = w\*sum;**

**f\_time = omp\_get\_wtime();**

**printf("n= %ld pi = %f time = %f \n", n,pi,f\_time - s\_time);**

**s\_time = omp\_get\_wtime();**

**sum = 0.0;**

**#pragma omp parallel for private(x) shared(w) reduction(+:sum) num\_threads(\_threads)**

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

**{**

**x = w\*(i-0.5);**

**sum = sum + f(x);**

**}**

**pi = w\*sum;**

**f\_time = omp\_get\_wtime();**

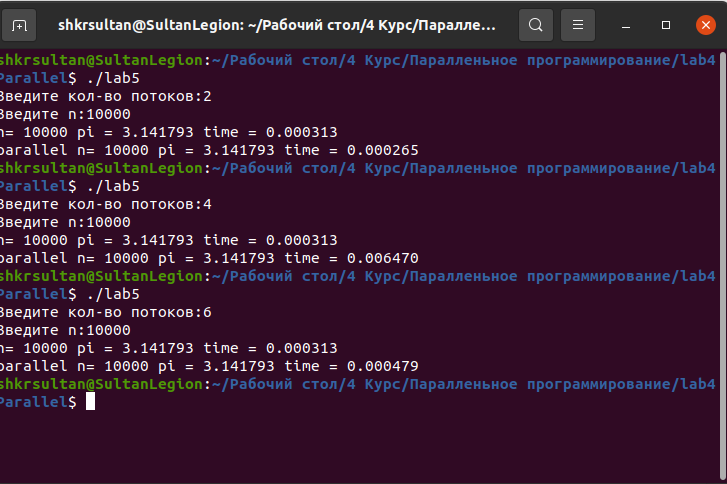
**printf("parallel n= %ld pi = %f time = %f \n", n,pi,f\_time - s\_time);**

**}**

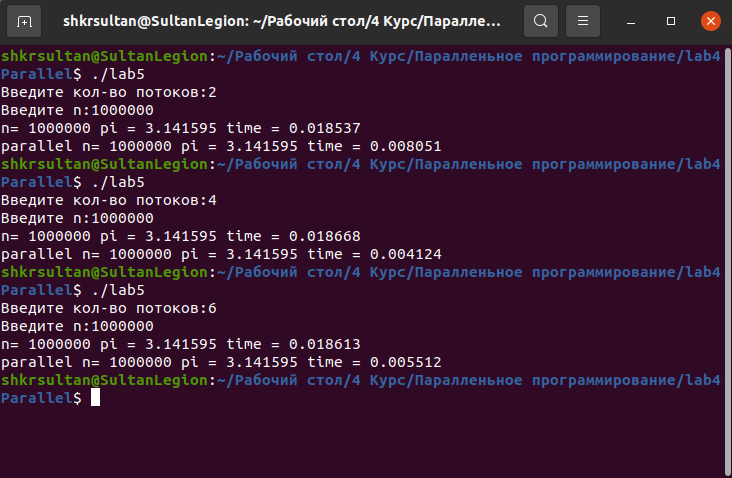
**Результат**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n | Последовательная программа | Параллельная программа (количество потоков) | | |
| 2 | 4 | 6 |
| 10^4 | 0.000313 | 0.000265 | 0.006470 | 0.000479 |
| 10^6 | 0.018537 | 0.008051 | 0.004124 | 0.005512 |
| 10^8 | 1.596 | 0.797078 | 0.404954 | 0.418996 |
| 10^10 | 159.4 | 79.682467 | 40.144519 | 44.952758 |

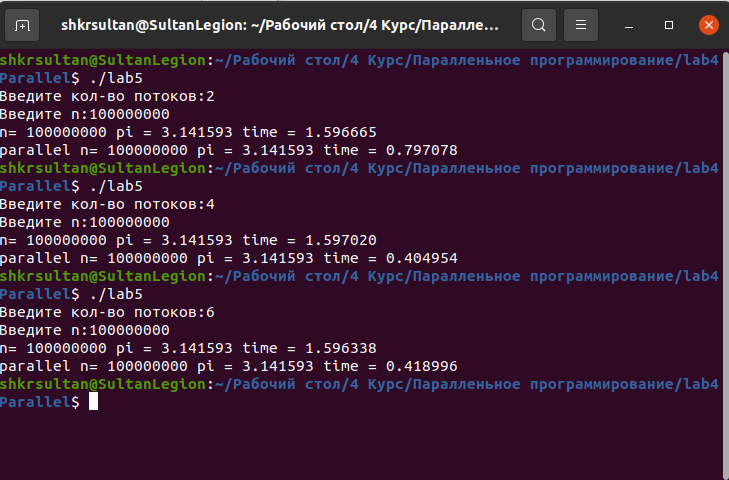
**10^4**

****

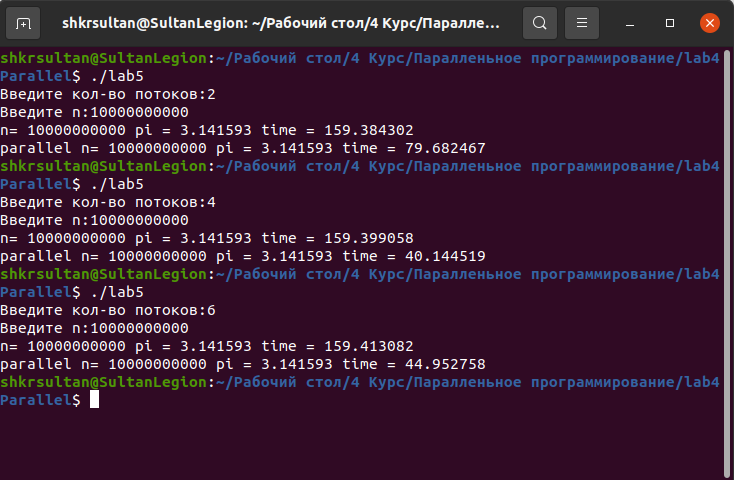
**10^6**

****

**10^8**

****

**10^10**

****