

**Отчет по**

**Лабораторной работе № 1. (2021, весна)**

**Базовые представления об OpenMP**

**Выполнил:**

Гломадов Антон – студент 3 курса Прикладной математики и информатики 1 подгруппа

2021г.

**Цель работы**: научиться пользоваться простейшими методами OpenMP и написать простейшие программы.

**Задачи**:

1. Задайте количество нитей, выполняющих параллельную область, равную 5.
2. Выведите номер текущей нити из общего количества используемых нитей в формате: I am <Номер нити> thread from <Количество нитей> threads!
3. Модифицируйте программу таким образом, чтобы строку

I am <Номер нити> thread from <Количество нитей> threads!

выводили только нити с четным номером.

1. Изучите  конструкции  для  управления  работой  с  данными  shared  и private.  Напишите программу,  в  которой  создается k нитей,  и каждая  нечетная нить выводит на экран свой номер через переменную rank, где rank = omp\_get\_thread\_num();

Экспериментами определите, общей или частной должна быть переменная rank.

**Листинг программы:**

#include <iostream>

//Библиотека для работы с OpenMP

#include <omp.h>

**int** **main**() {

//выставление кол-ва потоков

omp\_set\_num\_threads(**5**);

/\*

\* Задача 1 - Вывести номер потока и их кол-во

\*/

std::cout<<"Задача 1 - Вывести номер потока и их кол-во:**\n**";

//Определяет параллельную область, которая является кодом, который будет выполняться несколькими потоками параллельно.

#pragma omp parallel **default**(none)

{

//Указывает, что код должен выполняться только в одном потоке за раз.

#pragma omp critical

std::cout << "I am " << omp\_get\_thread\_num() << " thread from " << omp\_get\_num\_threads() << " threads"

<< std::endl;

}

/\*

\* Задача 2 - Вывести номер потока с четными номерами и кол-во потоков

\*/

std::cout<<"**\n**Задача 2 - Вывести номер потока с четными номерами и кол-во потоков:**\n**";

//Определяет параллельную область, которая является кодом, который будет выполняться несколькими потоками параллельно.

#pragma omp parallel **default**(none)

{

//Указывает, что код должен выполняться только в одном потоке за раз.

#pragma omp critical

**if** (omp\_get\_thread\_num()%**2**==**0**)

std::cout << "I am " << omp\_get\_thread\_num() << " thread from " << omp\_get\_num\_threads() << " threads"

<< std::endl;

}

/\*

\* Задача 3 - Создать K потоков. Создать переменную rank. C помощью переменной rank выести номера нечетных потоков;

\*/

**int** rank;

**int** numOfThreads;

std::cout<<"**\n**Задача 3 - Создать K потоков. Создать переменную rank. C помощью переменной rank выести номера нечетных потоков;**\n**";

std::cout<<"Введите кол-во потоков:**\n**";

std::cin>>numOfThreads;

//выставление кол-ва потоков

omp\_set\_num\_threads(numOfThreads);

//Определяет параллельную область, которая является кодом, который будет выполняться несколькими потоками параллельно.

#pragma omp parallel **private** (rank)

{

//присваеваем пременной rank номер потока

rank = omp\_get\_thread\_num();

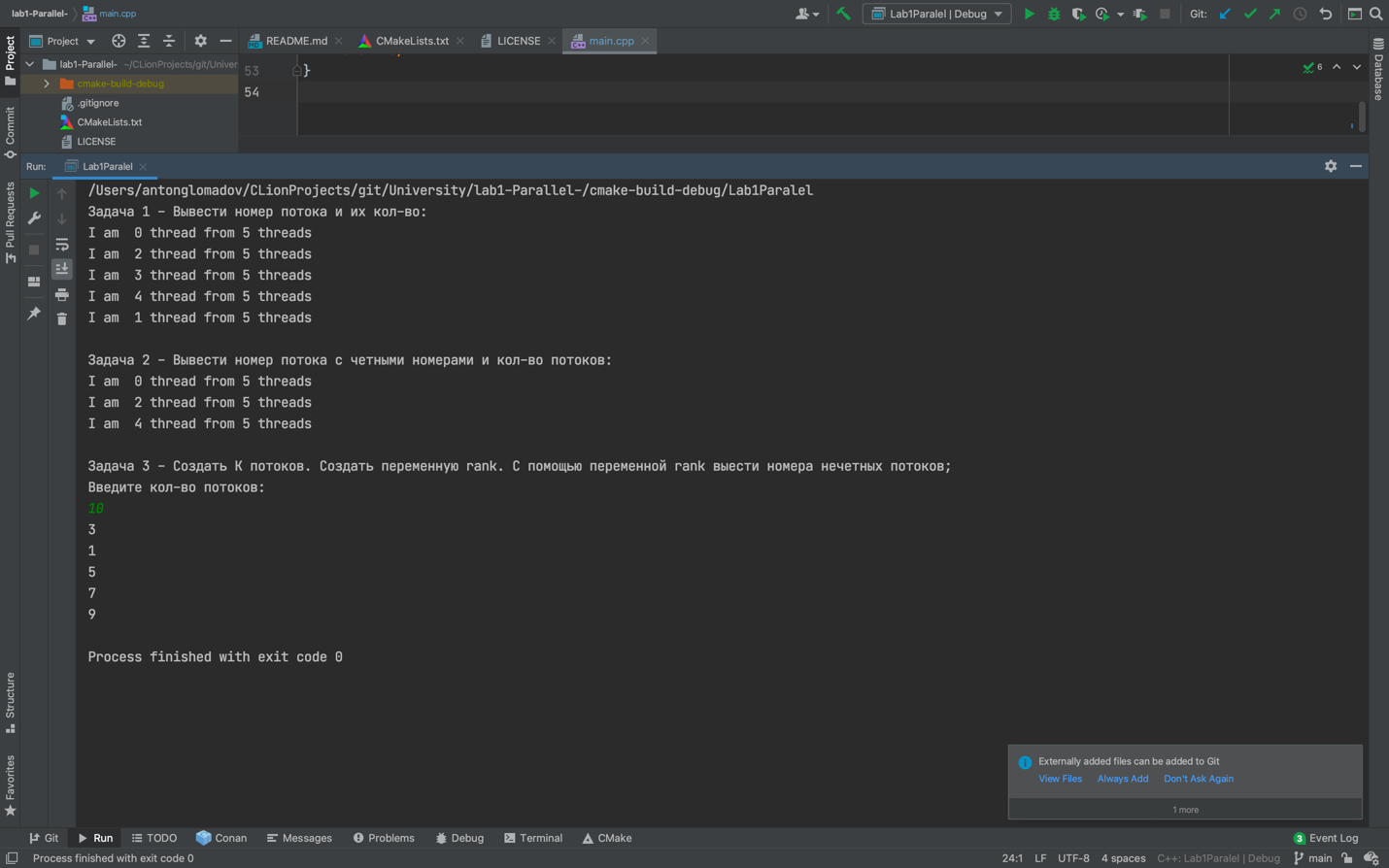
**if** (rank%**2**==**1**)

std::cout << rank<<std::endl;

}

**return** **0**;

}

**Результат работы программы:**

**Ссылка на репозитрий с данной программой:** <https://github.com/AntonGlomadov/lab1-Parallel-.git>

**Вывод:** в результате выполнения лабораторной работы №1 я освоил базовые навыки работы с OpenMP и. написал простейшую программу.