МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет информационных технологий

Кафедра параллельных вычислений

ОТЧЕТ

О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ

“**Программирование многопоточных приложений. POSIX Threads**”

студента 2 курса, 20212 группы

Курбатова Максима Андреевича

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель: Кудинов А.Ю.

Новосибирск 2022

Вывод:

По итогу лабораторной работы получилась симуляция работы кластера.

Каждый из узлов(в моем случае процессов) выполняет следующий алгоритм:

1. Достает из очереди свое задание и выполняет его
2. Если собственные задания закончились, что происходит запрос заданий у другого процесса.
3. Если никакой из процессов не прислал заданий текущему, значит итерация закончена.

Все задания хранятся в очереди (std::queue). Задания другим выдаются по одному. Если выдавать задания сразу большими пачками, то может произойти простой некоторых процессов.

Чтобы сделать безопасным доступ к разделяемой памяти - я использую mutex. Может возникнуть такая проблема.

В очереди всего 1 элемент. В первом потоке идет проверка на то, что очередь не пуста, после этого второй поток может вытащить этот элемент, но первый поток попытается достать этот элемент, не зная, что его нет. По итогу мы пошлем nullptr.

Поэтому, нужно использовать мьютекс, чтобы второй поток не мог вытащить из очереди что-то, по первый процесс не закончит свои действия с ней.