Министерство науки и высшего образования Российской Федерации

Ульяновский государственный Технический университет

Кафедра «Вычислительная техника»

Дисциплина «Операционные системы»

**Лабораторная работа №4**

**«****Проектирование пула потоков»**

**Выполнил**:

студент гр. ИВТАПбд-31

Сокольский Р. С.

**Проверил работу:**

Беляев К. С.

Ульяновск 2025

## Цель работы.

Изучить принципы многопоточного программирования, реализацию пула потоков и управление задачами с использованием POSIX Threads (PThreads), мьютексов и семафоров. Ознакомиться с механизмами синхронизации потоков для эффективного выполнения задач.

**Ход работы.**

В ходе работы был реализован пул потоков, обеспечивающий многопоточное выполнение задач с использованием POSIX Threads. Для управления задачами использовалась очередь, реализованная с применением связанного списка. Потоки взаимодействовали с очередью посредством мьютексов, предотвращающих одновременный доступ нескольких потоков к общей структуре данных, а также семафоров, обеспечивающих корректное распределение задач между потоками.

Программа состояла из трёх основных частей: реализации очереди задач, пула потоков и клиентской программы, демонстрирующей работу системы. Очередь позволяла динамически добавлять задачи и извлекать их в порядке поступления. Пул потоков включал в себя механизмы инициализации, запуска потоков и их синхронизации. Рабочие потоки ожидали поступления задач, извлекали их из очереди и выполняли переданную функцию. Клиентская программа демонстрировала процесс отправки нескольких задач в пул потоков и их параллельного выполнения.

После выполнения всех задач пул потоков корректно завершал свою работу, освобождая выделенные ресурсы и останавливая потоки. В результате удалось реализовать эффективное управление потоками, позволяющее распределять вычислительные задачи между ограниченным количеством потоков, обеспечивая оптимальную производительность программы.

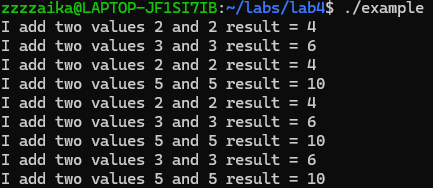


Рис. 1 вывод команды ./example.

**Вывод.**

В ходе работы был разработан пул потоков, который позволяет эффективно распределять задачи между потоками. Использование мьютексов предотвратило состояния гонки при доступе к очереди, а семафоры обеспечили синхронизацию выполнения потоков. Полученный код является примером многопоточной обработки задач и может применяться в серверных приложениях, распределённых вычислениях и других многозадачных системах.