

Задание 1

Необходимые знания

1. TCP и TCP/IP
2. TCP vs UDP
3. Системный вызов `socket`
4. Системный вызов `bind`
5. Системный вызов `listen`
6. Системный вызов `accept`
7. Системный вызов `recv`
8. Системный вызов `send`
9. Системный вызов `close`
10. Системный вызов `connect`

В предыдущей лабораторной работе вы распараллеливали вычисление факториала по модулю с помощью потоков. В этой работе вы пойдете еще дальше: вы распараллелите эту работу еще и между серверами.

Необходимо закончить `client.c` и `server.c`:

Клиент в качестве аргументов командной строки получает `k`, `mod`, `servers`, где `k` это факториал, который необходимо вычислить (`k! % mod`), `servers` это путь до файла, который содержит сервера (`ip:port`), между которыми клиент будет распараллеливать соединения.

Сервер получает от клиента "кусоч" своих вычислений и `mod`, в ответ отправляет клиенту результат этих вычислений.

Заметка

Обратите внимание, что клиент сейчас дожидается завершения работы с каждым сервером в цикле, т.е. последовательно. Это значит, что вам нужно подумать, как распараллелить работу с каждым сервером, чтобы сервера могли работать параллельно. Есть несколько вариантов, как это можно сделать, опираясь на уже пройденный вами материал, никто вас не ограничивает в выборе способа, но выбранный вариант вам необходимо объяснить.

TCP и TCP/IP

- **TCP/IP** — это набор сетевых протоколов, используемых для передачи данных в интернете.
- **TCP (Transmission Control Protocol)** — обеспечивает надёжную, упорядоченную и гарантированную доставку данных.
- **IP (Internet Protocol)** — отвечает за адресацию и маршрутизацию пакетов по сети.

TCP vs UDP

Свойство	TCP	UDP
Надёжность	Гарантирует доставку	Нет гарантии доставки
Порядок данных	Сохраняется	Не сохраняется
Скорость	Медленнее (из-за контроля)	Быстрее (без проверок)
Используется где	Веб, почта, файлы	Видео, игры, VoIP

Основные системные вызовы для сокетов (на стороне сервера)

socket

- Создаёт сокет (точку обмена данными).
- Указывается тип (например, TCP) и протокол.

bind

- Привязывает сокет к IP-адресу и порту.

listen

- Переводит сокет в режим ожидания входящих подключений (для TCP-сервера).

accept

- Принимает входящее соединение от клиента и возвращает новый сокет для общения.

recv

- Получает данные из сокета (от клиента).
- Работает с подключённым сокетом (например, после `accept`).

send

- Отправляет данные через сокет клиенту.

close

- Закрывает сокет, освобождая ресурсы.

Системный вызов для клиента

connect

- Устанавливает соединение с сервером (по IP и порту).
- Применяется на стороне клиента к ранее созданному сокету.

Сделано



Задание 2

Создать `makefile` для программ клиента и сервера

Сделано

Задание 3

Найти дублирующийся код в двух приложениях и вынести его в библиотеку. Добавить изменения в `makefile`.

Готово