На **удовлетворительно** - Лабораторные работы: любые две из №1-4, изучить особенности одного из протоколов: POP3, SMTP, FTP, TFTP, IPv4, TCP и UDP, SCTP.

На **хорошо** - Лабораторные работы: любые три из №1-4 изучить особенности протоколов: POP3 и SMTP; FTP и TFTP; IPv4 и IPv6 и сравнение; TCP, UDP и SCTP и их сравнение.

На **отлично** 3 Лабораторные работы: любые две из №1-4 из №1 - 4 и реализовать сетевое приложение на основе протокола прикладного уровня:

1) POP3-клиент (RFC1939)
2) SMTP-клиент (RFC2821)
3) IMAP4-клиент (RFC3501)
4) FTP-клиент (RFC959) (passiv)
5) FTP-клиент (RFC959) (active)
6) TFTP-клиент (RFC1350)
7) HTTP-сервер (RFC2616)
8) Сетевой чат (SCTP IPv4), параллельный сервер
9) Сетевой чат (SCTP IPv4), параллельный сервер

Лабораторная работа № 1

Тема: Параллельный (мультипроцессный) сервер.

Задание:

1. Написать программу обеспечивающую параллельную работу севера, принимающего информацию от клиента по сети. Условие: мультипроцессная организация на основе функции **fork**, транспортный протокол – TCP [1-3].

   Обеспечить в сервере завершение «зомби-процессов» !!!
2. Написать клиентскую программу, передающую заданное число ***i*** в цикле (определенное число раз с задержкой в ***i*** сек) на сервер. Соответсвующий процесс сервера выводит полученную информацию на экран.
3. Реализация - на языке C/C++, консольные приложения.

Серверная программа должна находить номер свободного порта и выводить его на экран (см. [1-3], стр. 338-342, функции bind, getsockname). При запуске клиентской программы задавать со строки IP адрес сервера и порт.

Лабораторная работа № 2

Тема: Параллельный (многопоточный) сервер.

Задание:

1. Написать программу обеспечивающую параллельную работу севера, принимающего информацию от клиента по сети. Информацию получаемую от клиента сохранять в одном общем файле (обеспечить целостность данных). Условие: мультипоточная организация на основе функций библиотеки **pthread**, транспортный протокол – TCP.
2. Написать клиентскую программу, передающую заданное число ***i*** в цикле (определенное число раз с задержкой в ***i*** сек) на сервер. Соответсвующий процесс сервера выводит полученную информацию на экран.
3. Продемонстрировать реализованные возможности программ согласно заданию.

4. Реализация - на языке C/C++, консольные приложения.

Серверная программа должна находить номер свободного порта и выводить его на экран (см. [1-3], функции bind, getsockname). При запуске клиентской программы задавать со строки IP адрес сервера и порт.

Лабораторная работа № 3

Тема: Псевдопараллельный сервер

Задание:

1. Разработать программу однопотокового сервера, использующую асинхронный ввод/вывод (организованный с помощью системного вызова **select**) обеспечивающую псевдопараллельную работу клиентов.
2. Написать клиентскую программу, передающую сообщения на сервер.
3. Продемонстрировать асинхронную работу сервера. Например, при запуске клиента пользователь задает число *i* от 1 до 10. Клиент передает серверу в цикле это число с задержкой в *i* секунд между передачей. Сервер отображает на экран полученную от клиентов информацию.
   Например:
   1-й клиент посылает число 1 в цикле с задержкой в 1 сек.
   2-ой клиент посылает число 2 с задержкой в 2 сек.
   3-й клиент посылает число 3 в цикле с задержкой в 3 сек.
   Сервер отображает информацию полученную от клиентов. Если у Вас правильно организован асинхронный ввод/вывод, то на экран со стороны сервера будет выводиться с чередованием числа 1, 2 и 3. Причем частота появления определенного числа будет зависеть от задержки по времени его передачи.
4. Реализация - на языке C/C++, консольные приложения.

Серверная программа должна находить номер свободного порта и выводить его на экран. При запуске клиентской программы задавать со строки IP адрес сервера и порт.

Лабораторная работа № 4

Тема: Мультипротокольный сервер.

Задание:

1. Написать программу, реализующую работу сервера по двум протоколам (TCP и UDP) параллельно с помощью системного вызова select.
2. Написать две клиентские программы, передающие на сервер файлы по протоколам TCP и UDP соответственно.
3. Реализация - на языке C/C++, консольные приложения.

Серверная программа должна находить номер свободного порта и выводить его на экран (см. [1-3], функции bind, getsockname). При запуске клиентской программы задавать со строки IP адрес сервера и порт.

Полезные ссылки:

1. Фейт С. TCP/IP: Архитектура, протоколы, реализация (включая IP версии 6 и IP Security). – М.: Лори, 2000. – 424 с.
2. Павский К. В., Ефимов А. В. Разработка сетевых приложений (протоколы TCP/IP, клиент-сервер, PCAP, Boost.ASIO) : Учебное пособие / Сибирский государственный университет телекоммуникаций и информатики. – Новосибирск, 2018. – 80 с.
3. Протоколы TCP/IP и разработка сетевых приложений : учеб. пособие / К.В. Павский ; Сиб. гос. ун-т телекоммуникаций и информатики. - Новосибирск : СибГУТИ, 2013. – 130с.