МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

## Факультет информационных технологий и робототехники

Кафедра программного обеспечения информационных систем

и технологий

**Отчет**

**по лабораторной работе № 11**

по дисциплине: ”Системное программирование”

на тему: ***”*** Сокеты и удаленное межпроцессное взаимодействие***”***

Выполнил**:** студент группы *10702121* Писарик А.С

Принял**:** пр. Давыденко Н.В.

Минск 2023

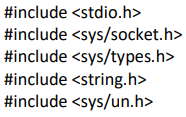
# Лабораторная работа №11.

**Цель работы:** Изучить механизм сокетного сетевого взаимодействия в LINUX

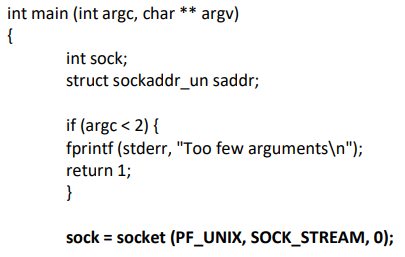
**Задание 1** Создать локальный сокет и назначить ему локальный адрес.

Решение

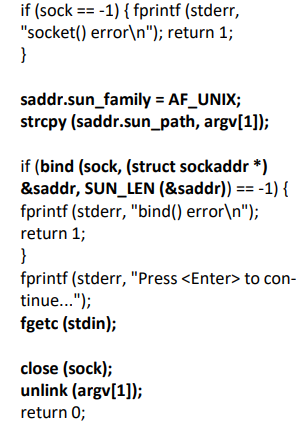
**Подключаем библиотеки**



**Создаем сокет**



**Назначаем ему локальный адрес**



**Проверяем:**



**Откроем другое терминальное окно и посмотрим на наш сокет:**



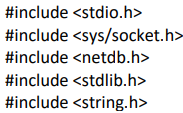
После нажатия клавиши в первом терминале сокет исчезнет.

**Какой консольной командой и каким системным вызовом в программах удаляется сокет из системы?**

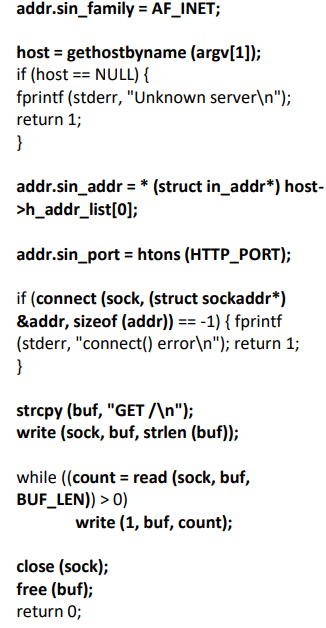
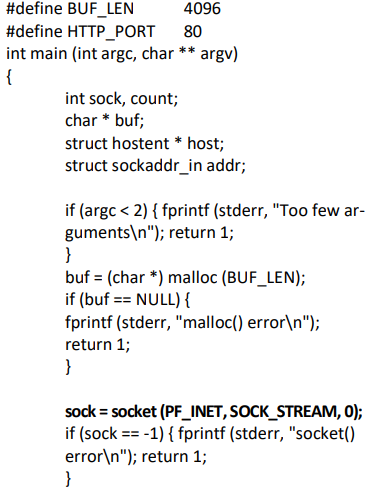
В ОС Linux консольной командой для удаления сокета является rm (remove). Однако, для удаления сокета из программы в ОС Linux часто используется системный вызов unlink, который удаляет файл из файловой системы по его имени. Сокеты в ОС Linux представлены в файловой системе как специальные файлы, поэтому удаление сокета в программе производится с помощью вызова unlink.

**Задание 2: Реализуйте чтение главное страницы из указанного web-сервиса**

**Подключаем библиотеки**



**Реализуем чтение**



**Подключаемся к интернету**



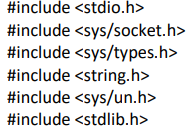
**Заносим инфу в файл**



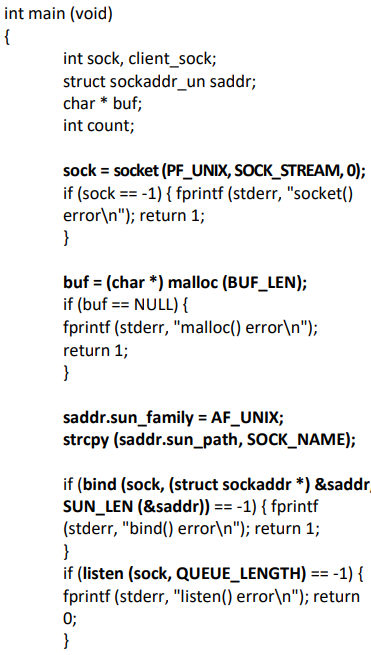
**Задание 3 Реализовать межпроцессное взаимодействие с использованием локальных потоковых сокетов.**

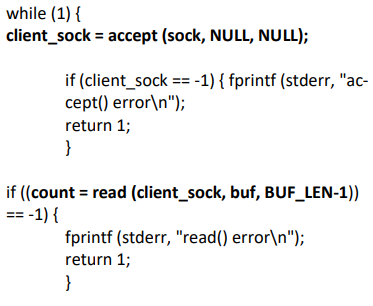
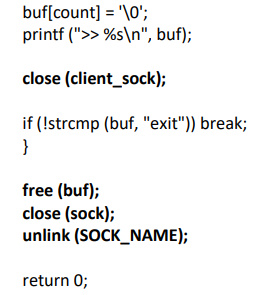
Создадим локальное клиент-серверное приложение.

Подключим библиотеки

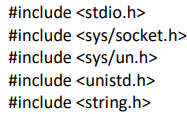


**Создадим сервер**

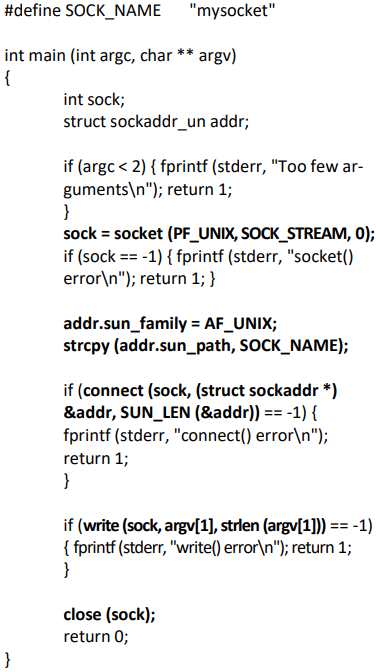


**Подключим библиотеки**



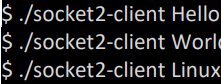
**Создадим клиента**



**После создания сервера переходим в режим прослушивания сокета**

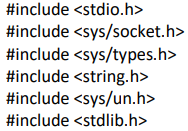
**Откроем другое терминальное окно и передадим серверу запросы:**

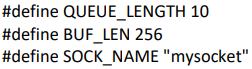
**Задание 4 Реализовать прием и передачу данных через дейтаграммные сокеты**

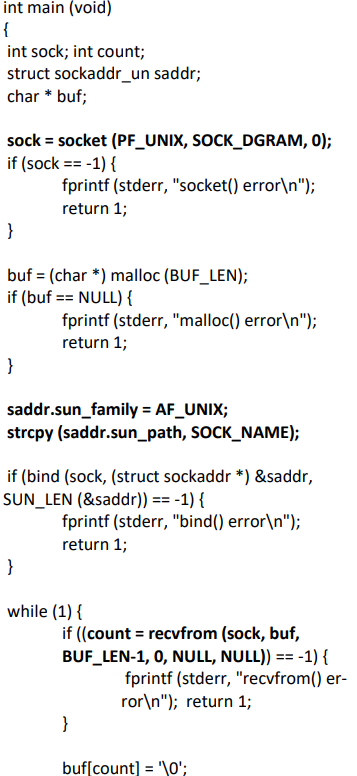
**Создаем сервер**

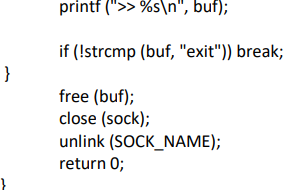
**Подключаем библиотеки**



**Реализуем**

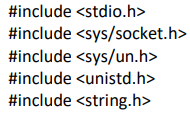






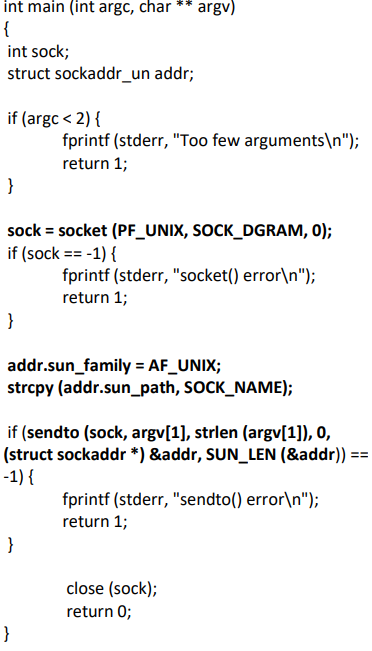
**Создаем клиента**

**Подключаем библиотеки**



**Реализуем**





**Контрольные**

3) **Что такое сокет? Как он представлен в операционной системе?**

Сокет - это абстрактное понятие для реализации механизма межпроцессного взаимодействия (IPC) в операционной системе. Он представляет собой интерфейс для обмена данными между процессами, работающими на одной машине или на разных машинах в сети.

В операционной системе сокет представлен в виде файлового дескриптора, что позволяет программам использовать стандартные файловые операции для взаимодействия с сокетом. Когда сокет создается, операционная система выделяет соответствующий файловый дескриптор и связывает его с сетевыми ресурсами, такими как IP-адрес и порт. Это позволяет процессам читать и записывать данные через сокет, а также выполнять другие операции, такие как установка соединения и прослушивание запросов от клиентов.

Сокеты позволяют процессам различных программ обмениваться данными через сеть или между различными частями одной программы, что делает их мощным инструментом для разработки сетевых приложений и клиент-серверных систем.

4) **Какие операции можно производить над сокетом?**

Сокет предоставляет различные операции для взаимодействия с протоколами в сети. Вот некоторые основные операции, которые можно производить над сокетом:

1. Создание сокета: Сокет можно создать с помощью функции socket(), указав тип сокета (например, SOCKSTREAM для TCP или SOCKDGRAM для UDP) и домен (например, AFINET для IPv4 или AFINET6 для IPv6).

2. Связывание сокета: С помощью функции bind() сокет можно связать с конкретным сетевым адресом и портом. Это позволяет сокету прослушивать входящие соединения на определенном адресе и порту.

3. Установка в режим прослушивания: Для сокета TCP можно установить режим прослушивания (listening) с помощью функции listen(). Это позволяет сокету ожидать входящих подключений от клиентов.

4. Принятие соединения: Для сокета TCP функция accept() позволяет принять входящее подключение от клиента. Она создает новый сокет для этого соединения и возвращает файловый дескриптор этого сокета.

5. Чтение и запись данных: Сокеты позволяют производить операции чтения и записи данных. Для сокета TCP функции read() и write() используются для чтения данных из сокета и записи данных в сокет соответственно. Для сокета UDP используется функция sendto() для отправки данных и recvfrom() для приема данных.

6. Закрытие сокета: Когда работа с сокетом завершена, его следует закрыть с помощью функции close(). Это освободит ресурсы, связанные с сокетом, и закроет соединение, если оно было установлено.

Кроме этих основных операций, также существуют дополнительные операции, такие как настройка параметров сокета, установка и разрыв соединений, установка таймаутов и т.д., которые зависят от конкретного протокола и типа сокета.