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 № 10**

по дисциплине: «Системное программирование»

на тему: «Сокеты и удаленное межпроцессное взаимодействие»

Выполнила: ст. гр. 10702221

Багапов Д.М.

Матвейчук К.А.

Яцук З.Р.

Проверил: Давыденко Н.В.

Минск 2023

**ЛАБОРАТОРНАЯ РАБОТА # 10**

**Сокеты и удаленное межпроцессное взаимодействие  
Цель работы :**

Изучить механизм сокетного сетевого взаимодействия в LINUX.

**Задания лабораторной:**

**Задание 1:**

Напишите программы с клиент-серверной архитектурой по следующим требованиям:

1. Обменяться числами. Сервер возводит получаемое от клиента число в

квадрат и возвращает ответ клиенту.

2. Создать два клиента на один и тот же сокет. Используют один и тот же

сокет. Первый клиент получает результат счета от 1 до 10, второй – результат счета от 100 до 110 (подумайте, как распознать, какой клиент

обратился к серверу).

Примечание. Если количество студентов команды, выполняющих задание больше двух, то

число процессов-клиентов должно соответствовать числу студентов команды.

Если клиентов больше двух, то в этом случае процессы должны дополнительно выводить

фамилии студентов, выполнивших задания. Третий клиент получает результат счета от 200

до 210. Каждому последующему клиенту сообщение увеличивается на 100.

19

СОКЕТЫ И УДАЛЕННОЕ МЕЖПРОЦЕССНОЕ ВЗАИМОДЕЙСТВИЕ СЕРГЕЙ СТАНКЕВИЧ, МИНСК

3. Создать два клиента на один и тот же сервер. Сокеты должны быть разными (именованное соединение в каждом случае должно быть различным). Первый клиент получает результат счета от 1 до 10, второй – от

100 до 110.

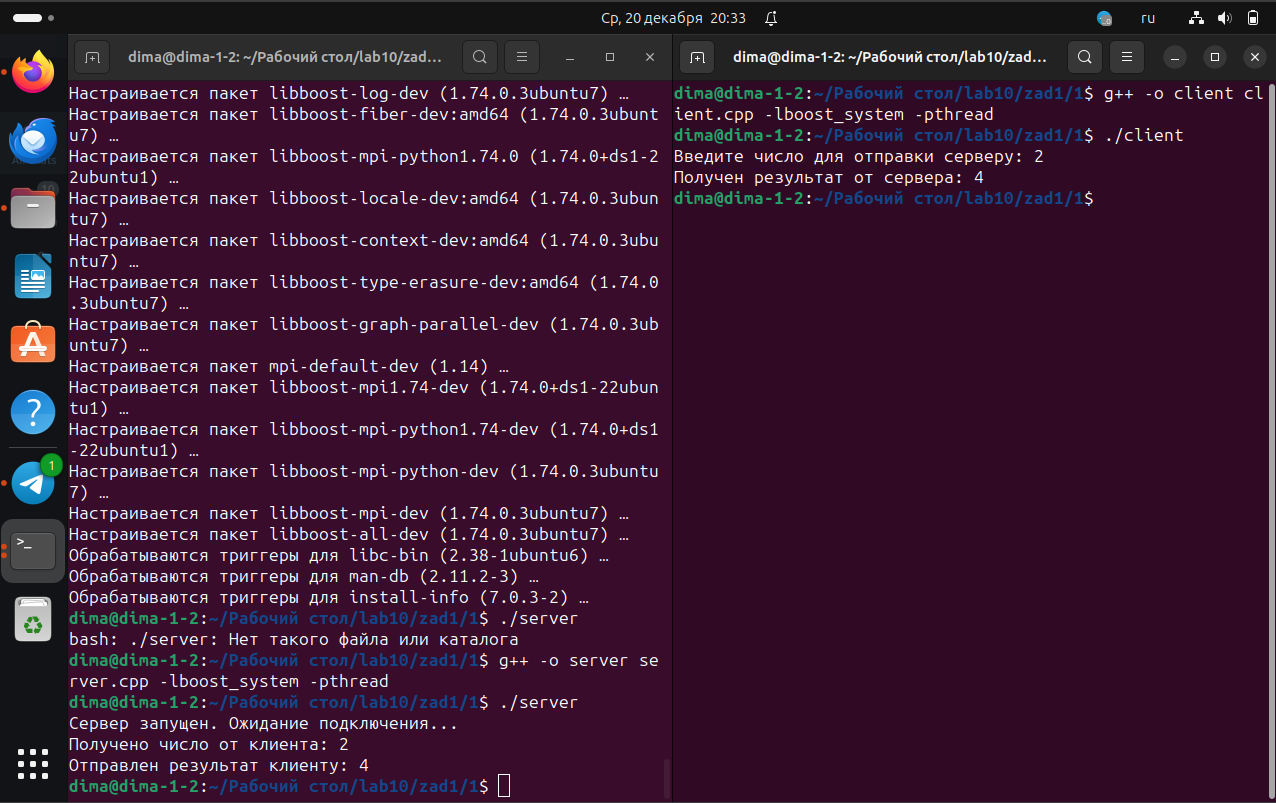
Примечание. Число процессов-клиентов и сокетов должно соответствовать числу студентов команды, выполняющей задание.

Если клиентов больше двух, то в этом случае процессы должны дополнительно выводить

фамилии студентов, выполнивших задания. Каждому последующему клиенту сообщение

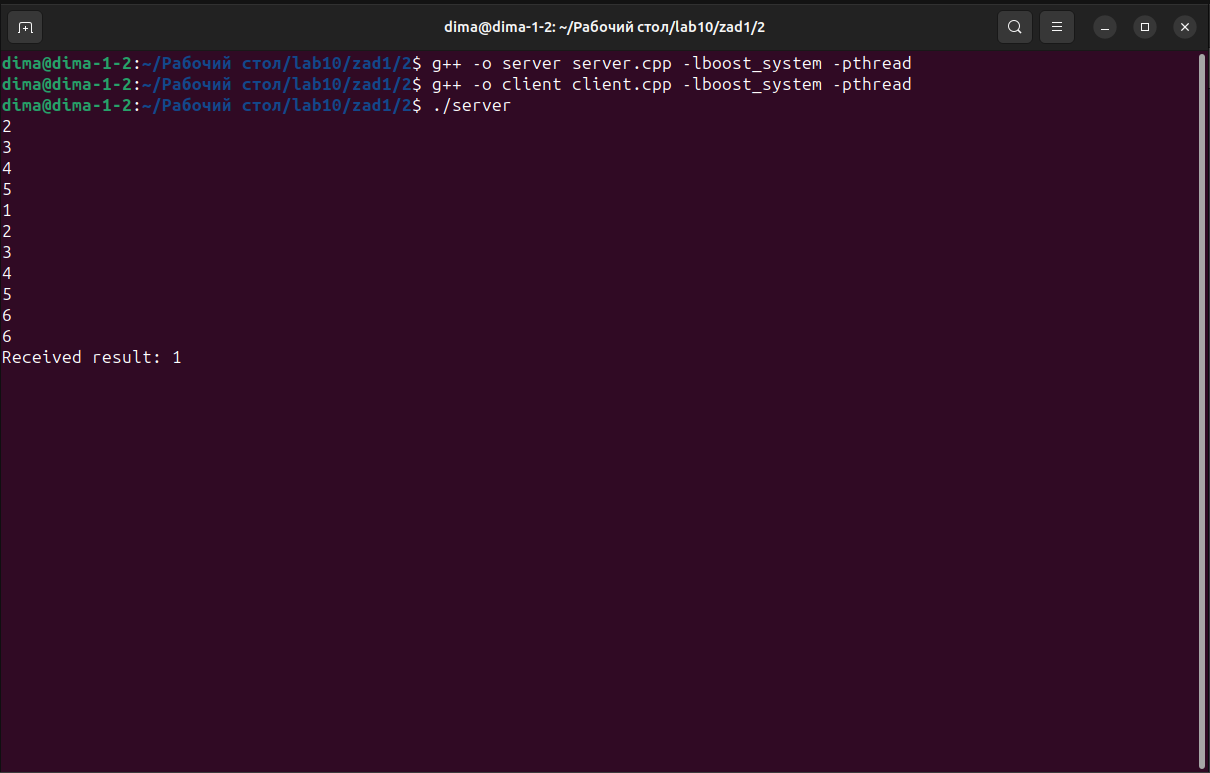
увеличивается на 100.

**Выполнение:**

1. 

**СРАБОТАЛО!!!!!!**

**2.**

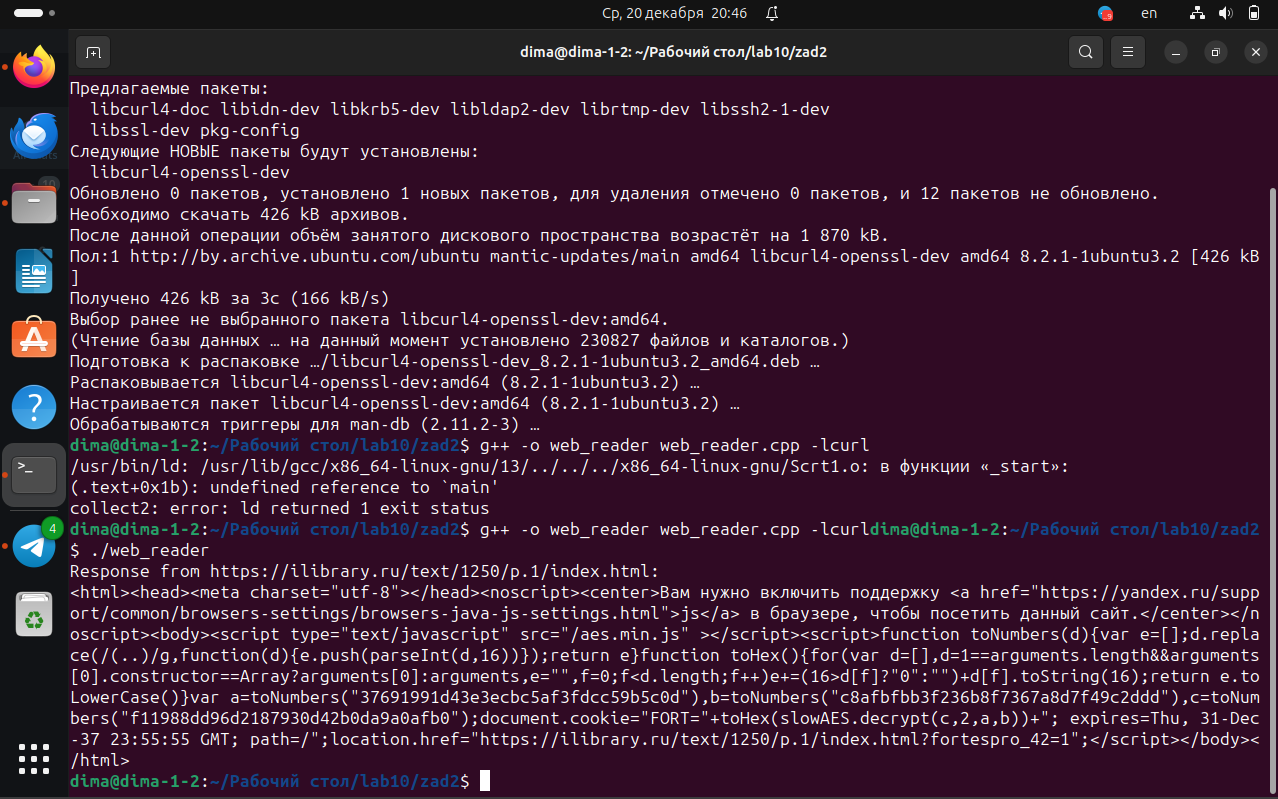


**Успешное выполнение**

**3.**

**Задание 2:**

Напишите программу, которая читает главную страницу из какого-нибудь Web-сервера. Запуск программы должен производиться с консоли.

 **Успешное выполнение**

**Контрольные вопросы:**

**1. Что такое межпроцессное взаимодействие?**

- Межпроцессное взаимодействие (IPC) — это механизм обмена данными между процессами в операционной системе. Он позволяет процессам совместно использовать информацию и координировать свою работу.

**2. Что такое каналы?**

- В контексте межпроцессного взаимодействия, каналы представляют собой один из механизмов передачи данных между процессами. Каналы позволяют установить связь между процессами таким образом, что один процесс может отправлять данные, а другой — принимать их.

**3. Какие типы каналов существуют?**

- Существуют два основных типа каналов: анонимные (или неименованные) и именованные. Анонимные каналы создаются динамически для обмена данными между родственными процессами, а именованные каналы имеют идентификатор в файловой системе и могут использоваться для обмена данными между процессами, даже если они не родственны.

**4. Что такое именованные каналы?**

- Именованные каналы — это механизм межпроцессного взаимодействия, при котором создается файл в файловой системе, который становится именованным каналом. Процессы могут открывать этот файл для чтения или записи, обеспечивая тем самым обмен данными.

**5. К какому виду межпроцессного взаимодействия относятся каналы?**

- Каналы относятся к однонаправленному, неименованному межпроцессному взаимодействию. По широте охвата — к ограниченному, по направлению передачи данных — к однонаправленному, по характеру доступа — к неявному (без использования прямых указателей).

**6. Команды, системные вызовы и библиотечные функции, относящиеся к каналам, и библиотеки их подключения.**

- Некоторые команды и функции:

- Команды: `pipe`, `mkfifo`.

- Системные вызовы: `pipe`, `fork`, `close`, `read`, `write`.

- Библиотечные функции: `popen`, `pclose`.

- Библиотеки: `<unistd.h>`, `<fcntl.h>`, `<stdio.h>`.

**7. Команды и системные вызовы, относящиеся к именованным каналам.**

- Системные вызовы: `mkfifo`, `open`.

- Команды: `mkfifo`.

- Библиотеки: `<sys/types.h>`, `<sys/stat.h>`, `<fcntl.h>`.

**8. Атрибуты системных вызовов.**

- Некоторые атрибуты системных вызовов:

- `pipe`: Создает анонимный канал.

- `mkfifo`: Создает именованный канал (FIFO).

- `open`: Открывает файл или канал.

**9. Какие задачи выполняют библиотечные функции `popen()`, `pclose()`.**

- `popen()`: Открывает канал для выполнения команды в дочернем процессе и возвращает указатель на поток для чтения или записи.

- `pclose()`: Закрывает канал, созданный `popen()`, и возвращает статус завершения дочернего процесса.

**10. Что такое однонаправленное межпроцессное взаимодействие?**

- Однонаправленное межпроцессное взаимодействие — это сценарий, при котором данные передаются только в одном направлении, например, от родительского процесса к дочернему, или наоборот.

**11. Что такое архитектура приложений клиент/сервер, и в чем ее отличие от архитектуры взаимодействия процессов «источник» - «приёмник».**

- Архитектура клиент/сервер: Это модель взаимодействия, где сервер предоставляет услуги или ресурсы, а клиент запрашивает их. Клиент и сервер могут быть выполнены как отдельные процессы или даже на разных устройствах.

- Архитектура взаимодействия процессов "источник" - "приёмник": Процессы взаимодействуют напрямую друг с другом, обмениваясь данными или отправляя сигналы. В этом случае, каждый процесс может быть источником данных, а другой — их приёмником, и наоборот. В отличие от клиент/сервер, здесь нет явного разделения на роли клиента и сервера.