МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

ЛАБОРАТОРНАЯ РАБОТА №3

по дисциплине

“Низкоуровневое программирование”

Вариант № 18

Студент:

Рябоконь Архип
Борисович

Группа P33302

Преподаватель:

Кореньков Юрий Дмитриевич

Санкт-Петербург, 2024

Папка с видео работы:
https://drive.google.com/drive/folders/1zfvMmkWPz1FSVPunDRywQ6GwTCa3usrt?usp=sharing

Видео:
https://drive.google.com/file/d/1_CAMlsNzeASNErfs45xggbGlbtyJG7Ee/view?usp=sharing

Задание:
На базе данного транспортного формата описать схему протокола обмена информацией и воспользоваться существующей библиотекой по выбору для реализации модуля, обеспечивающего его функционирование. Протокол должен включать представление информации о командах создания, выборки, модификации и удаления данных в соответствии с данной формой, и результатах их выполнения. Используя созданные в результате выполнения заданий модули, разработать в виде консольного приложения две программы: клиентскую и серверную части. Серверная часть – получающая по сети запросы и операции описанного формата и последовательно выполняющая их над файлом данных с помощью модуля из первого задания. Имя фала данных для работы получать с аргументами командной строки, создавать новый в случае его отсутствия. Клиентская часть – в цикле получающая на стандартный ввод текст команд, извлекающая из него информацию о запрашиваемой операции с помощью модуля из второго задания и пересылающая её на сервер с помощью модуля для обмена информацией, получающая ответ и выводящая его в человеко-понятном виде в стандартный вывод.
Порядок выполнения:
1 Изучить выбранную библиотеку
    a. Библиотека должна обеспечивать сериализацию и десериализацию с валидацией в
    соответствии со схемой
    b. Предпочтителен выбор библиотек, поддерживающих кодогенерацию на основе схемы
    c. Библиотека может поддерживать передачу данных посредством TCP соединения
        Иначе, использовать сетевые сокеты посредством API ОС

d. Библиотека может обеспечивать диспетчеризацию удалённых вызовов

Иначе, реализовать диспетчеризацию вызовов на основе информации о виде команды

2 На основе существующей библиотеки реализовать модуль, обеспечивающий взаимодействие

a. Описать схему протокола в поддерживаемом библиотекой формате

Описание должно включать информацию о командах, их аргументах и результатах

Схема может включать дополнительные сущности (например, для итератора)

b. Подключить библиотеку к проекту и сформировать публичный интерфейс модуля с
использованием встроенных или сгенерированных структур данных используемой библиотеки

Поддержать установление соединения, отправку команд и получение их результатов

Поддержать приём входящих соединений, приём команд и отправку их результатов

c.Реализовать публичный интерфейс посредством библиотеки в соответствии с п1

3 Реализовать серверную часть в виде консольного приложения

a. В качестве аргументов командной строки приложение принимает:

Адрес локальной конечной точки для прослушивания входящих соединений

Имя файла данных, который необходимо открыть, если он существует, иначе создать

b. Работает с файлом данных посредством модуля из задания 1

c. Принимает входящие соединения и взаимодействует с клиентами посредством модуля из п2

d. Поступающая информация о запрашиваемых операциях преобразуется из структур данных модуля взаимодействия к структурам данных модуля управления данными и наоборот

4 Реализовать клиентскую часть в виде консольного приложения

a. В качестве аргументов командной строки приложение принимает адрес конечной точки для подключения

b. Подключается к серверу и взаимодействует с ним посредством модуля из п2

c.Читает со стандартного ввода текст команд и анализирует их посредством модуля из задания 2

d. Преобразует результат разбора команды к структурам данных модуля из п2, передаёт их для обработки на сервер, возвращаемые результаты выводит в стандартный поток вывода

5 Результаты тестирования представить в виде отчёта, в который включить:

d. В части 3 привести пример сеанса работы разработанных программ

e. В части 4 описать решение, реализованное в соответствии с пп.2-4

f.В часть 5 включить составленную схему п.2а

2. Общее описание решения.

Лабораторная состоит из 2ух модулей.

1 — Сервер, основан на лаб1

2 — Клиент, основан на лаб2

Для связи между клиентом и сервером используются Unix sockets.

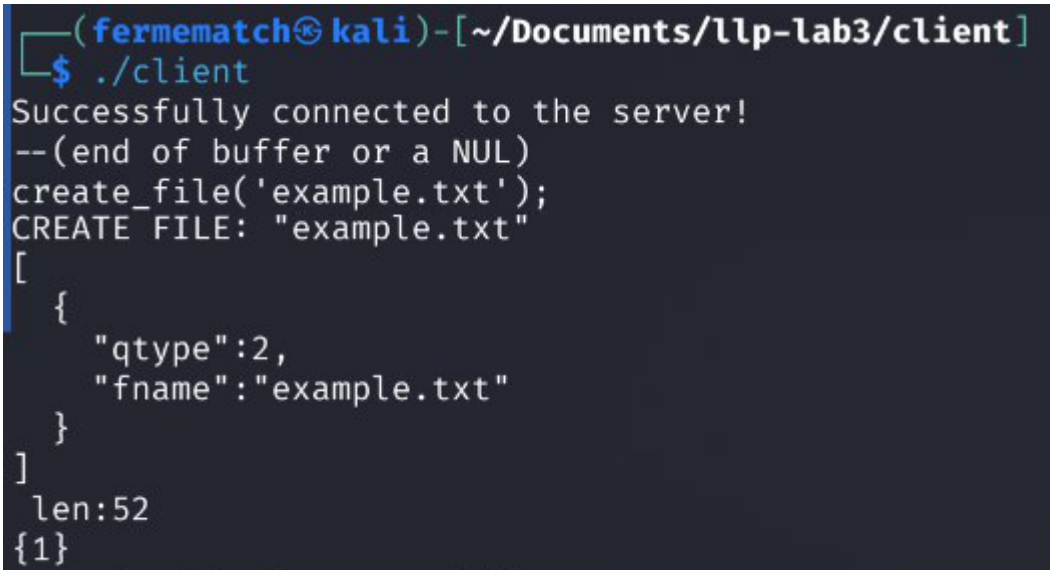
Для сериализации и десериализации данных используется библиотека json-c.

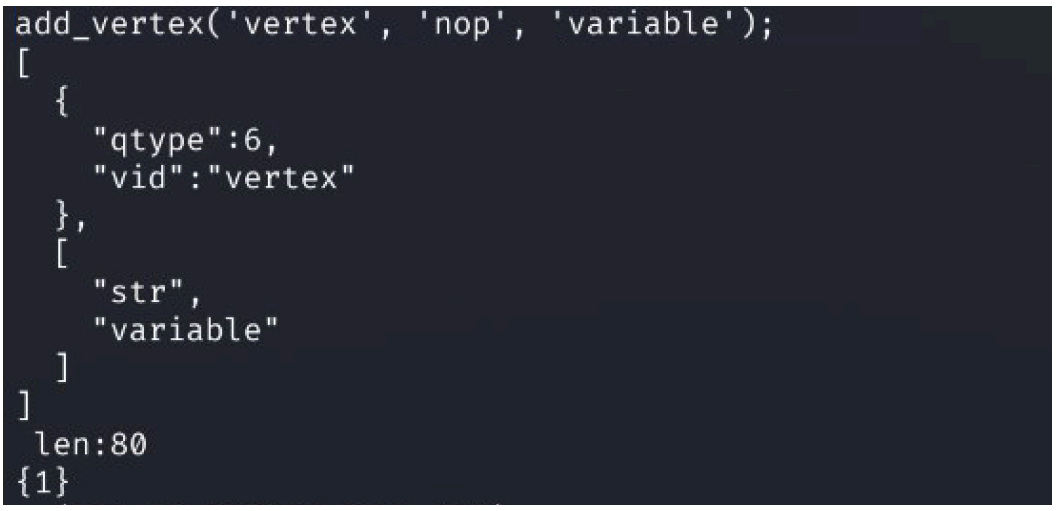
Схема протокола работы с данными может быть описана так: struct ast_tree, полученный из lab2 передаётся функции ast2json(), которая возвращает json,
полученный из ast_tree. Потом этот json приводится к текстовому виду и отправляется на сервер. На сервере функция handle_client() принимает json и
парсит содержащийся в нём запрос, после чего выполняет его. Потом функция
отправляет ответ клиенту, который тот выводит в консоль. Ответ может иметь
вид json’а в случае возвращения данных, или простой текстовой строки в случае
возвращения простого сообщения.

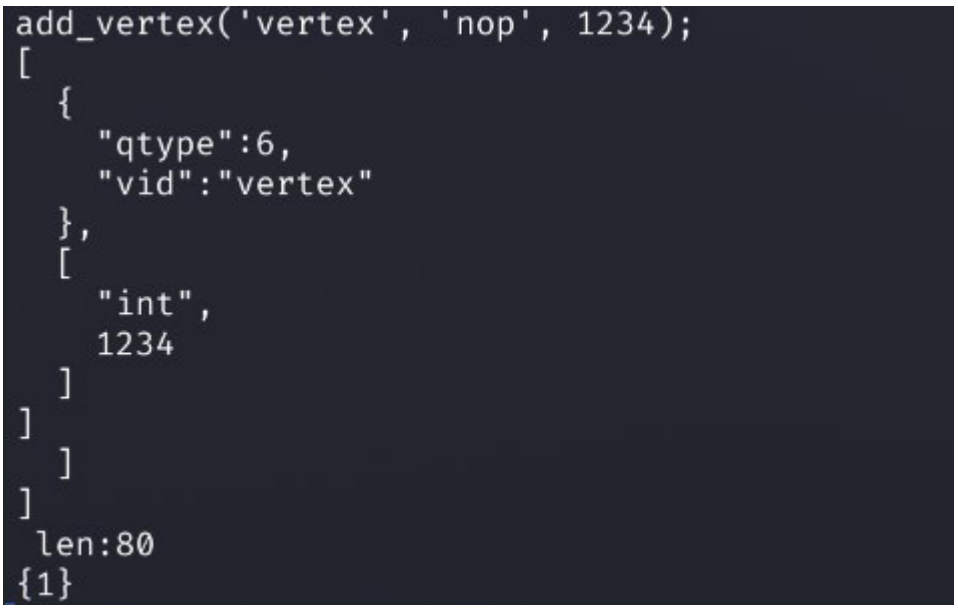
Примеры выполнения:

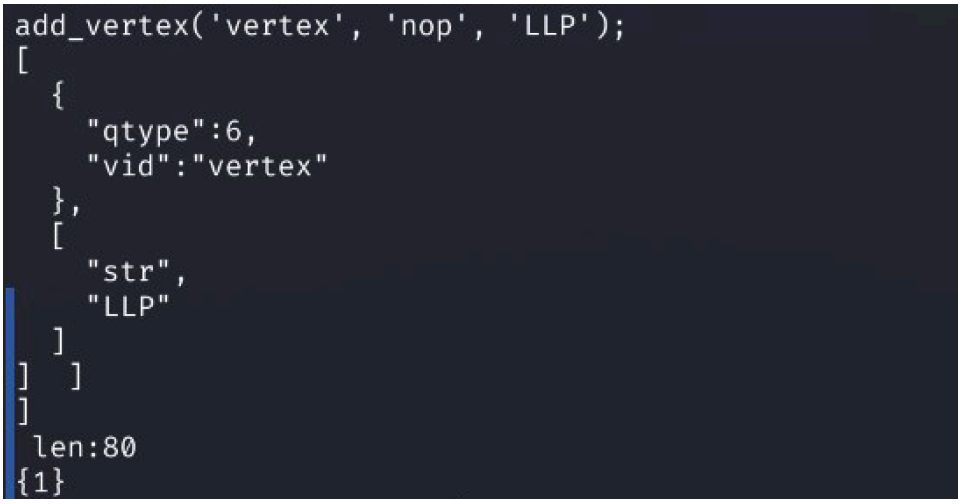
Начало файла по загрузке данных:
create_file('nanan.txt');
add_vertex('vertex', 'smth', 'AFG', 'smth', 4326, 'smth', 65.0, 'smth', 33.0, 'smth', 'Afghanistan', 'smth', 'AF');
add_vertex('vertex', 'smth', 'ALB', 'smth', 4326, 'smth', 20.0, 'smth', 41.0, 'smth', 'Albania', 'smth', 'AL');
add_vertex('vertex', 'smth', 'DZA', 'smth', 4326, 'smth', 3.0, 'smth', 28.0, 'smth', 'Algeria', 'smth', 'DZ');
add_vertex('vertex', 'smth', 'ASM', 'smth', 4326, 'smth', 170.0, 'smth', 14.3333, 'smth', 'AmericanSamoa', 'smth', 'AS');
add_vertex('vertex', 'smth', 'AND', 'smth', 4326, 'smth', 1.6, 'smth', 42.5, 'smth', 'Andorra', 'smth', 'AD');
add_vertex('vertex', 'smth', 'AGO', 'smth', 4326, 'smth', 18.5, 'smth', 12.5, 'smth', 'Angola', 'smth', 'AO');
add_vertex('vertex', 'smth', 'AIA', 'smth', 4326, 'smth', 63.1667, 'smth', 18.25, 'smth', 'Anguilla', 'smth', 'AI');
add_vertex('vertex', 'smth', 'ATA', 'smth', 4326, 'smth', 0.0, 'smth', 90.0, 'smth', 'Antarctica', 'smth', 'AQ');
add_vertex('vertex', 'smth', 'ATG', 'smth', 4326, 'smth', 61.8, 'smth', 17.05, 'smth', 'AntiguaandBarbuda', 'smth', 'AG');
add_vertex('vertex', 'smth', 'ARG', 'smth', 4326, 'smth', 64.0, 'smth', 34.0, 'smth', 'Argentina', 'smth', 'AR');
add_vertex('vertex', 'smth', 'ARM', 'smth', 4326, 'smth', 45.0, 'smth', 40.0, 'smth', 'Armenia', 'smth', 'AM');
add_vertex('vertex', 'smth', 'ABW', 'smth', 4326, 'smth', 69.9667, 'smth', 12.5, 'smth', 'Aruba', 'smth', 'AW');
add_vertex('vertex', 'smth', 'AUS', 'smth', 4326, 'smth', 133.0, 'smth', 27.0, 'smth', 'Australia', 'smth', 'AU');
add_vertex('vertex', 'smth', 'AUT', 'smth', 4326, 'smth', 13.3333, 'smth', 47.3333, 'smth', 'Austria', 'smth', 'AT');
add_vertex('vertex', 'smth', 'AZE', 'smth', 4326, 'smth', 47.5, 'smth', 40.5, 'smth', 'Azerbaijan', 'smth', 'AZ');
add_vertex('vertex', 'smth', 'BHS', 'smth', 4326, 'smth', 76.0, 'smth', 24.25, 'smth', 'Bahamas', 'smth', 'BS');
Загрузка:
./client < dataset/out.txt

Примеры запросов к БД:
Открытие файла и создание в нём вершин:









Селект элемента:

```
V('d').has('str0', eq('LLP'));
[
  {
    "vid":2,
    "n_con":0,
    "n_int":0,
    "n_flt":0,
    "n_str":1,
    "conn":[],
    "ints":[],
    "flts":[],
    "strs":[
      "LLP"
    ]
  }
]
```

```
V('d').has('str0', contains('ari'));
[
  {
    "vid":0,
    "n_con":0,
    "n_int":0,
    "n_flt":0,
    "n_str":1,
    "conn":[],
    "ints":[],
    "flts":[],
    "strs":[
      "variable"
    ]
  }
]
```

Удаление элемента:

```
V('d').has('int0', eq(1234));
[
  {
    "vid":1,
    "n_con":0,
    "n_int":1,
    "n_flt":0,
    "n_str":0,
    "conn":[],
    "ints":[
      1234
    ],
    "flts":[],
    "strs":[]
  }
]
```

```
V('d').has('int0', eq(1234)).delete;
[]
```

```
V('d').has('int0', eq(1234));
[]
```

Запросы ке загруженному датасету:

Получение элемента, у которого строка равна TUR

V('d').has('str0', eq('TUR'));

```
Validating JSON schema...
**key:    $schema
**key:    type
**key:    items
Valid schema!
Validating JSON file...
          KEYWORD $schema 0
          KEYWORD type 0
          KEYWORD items 0
ALL OK

 Valid JSON file!
0: 230
```

```
V('d').has('str0', eq('TUR'));
[
  {
    "vid":230,
    "n_con":0,
    "n_int":3,
    "n_flt":0,
    "n_str":3,
    "conn":[],
    "ints":[
      4326,
      35,
      39
    ],
    "flts":[],
    "strs":[
      "TUR",
      "Turkey",
      "TR"
    ]
  }
]
```

Получение элемента, у которого строка равна HabibBankAGZurich
V(‘d’).has(‘str1’, eq(‘HabibBankAGZurich’));

```
Validating JSON file...
        KEYWORD $schema 0
        KEYWORD type 0
        KEYWORD items 0
ALL OK

 Valid JSON file!
0: 406
1: 402
2: 391
```

```
V('d').has('str1', eq('HabibBankAGZurich'));
[
  {
    "vid":406,
    "n_con":1,
    "n_int":3,
    "n_flt":0,
    "n_str":2,
    "conn":[
      238
    ],
    "ints":[
      4326,
      54,
      24
    ],
    "flts":[],
    "strs":[
      "habibbankagzurichdubaiunitedarabemiratesare",
      "HabibBankAGZurich"
    ]
  },
  {
    "vid":402,
    "n_con":1,
    "n_int":3,
    "n_flt":0,
    "n_str":2,
    "conn":[
      238
    ],
    "ints":[
      4326,
      54,
      24
    ],
    "flts":[],
    "strs":[
      "habibbankagzurichdubaiuaeare",
      "HabibBankAGZurich"
    ]
  },
  {
    "vid":391,
    "n_con":1,
    "n_int":3,
    "n_flt":0,
    "n_str":2,
    "conn":[
      238
    ],
    "ints":[
      4326,
      54,
      24
    ],
    "flts":[],
    "strs":[
      "habibbankagzurichdubaiunitedarabemiratesare",
      "HabibBankAGZurich"
    ]
  }
]
```

Запрос на то чтобы достать дочерний элемент:
Получение дочернего элемента связанного с элементом с айди 406
V('d').has('sid', eq(406)).in('conn').otherV()

```
V('d').has('sid', eq(406)).in('conn').otherV();
[
  {
    "vid":238,
    "n_con":16,
    "n_int":3,
    "n_flt":0,
    "n_str":3,
    "conn":[
      261,
      274,
      301,
      302,
      313,
      314,
      362,
      369,
      371,
      372,
      390,
      391,
      392,
      402,
      406,
      417
    ],
    "ints":[
      4326,
      97,
      38
    ],
    "flts":[],
    "strs":[
      "USA",
      "UnitedStates",
      "US"
    ]
  }
]
```

Вывод: В данной лабораторной работе я ознакомился с написание клиент-серверного приложения на C, а так же с сериализацией и десериализацией данных в форматах JSON с помощью библиотеки JSON-C