Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет Программной Инженерии и Компьютерной Техники

**Лабораторная работа № 2**

по дисциплине

«Низкоуровневое программирование» Вариант: 9 (Cypher)

Выполнил:

Скориков Родион

Группа: P33312

Преподаватель:

Кореньков Юрий Дмитриевич

Санкт-Петербург, 2024г

# Цель

Реализовать модуль для разбора некоторого достаточного подмножества языка запросов по выбору в соответствии с вариантом формы данных.

# Задачи

1. Изучить выбранное средство синтаксического анализа.
2. Изучить синтаксис языка запросов и записать спецификацию для средства синтаксического анализа.
3. Реализовать модуль, использующий средство синтаксического анализа для разбора языка запросов.
4. Реализовать тестовую программу для демонстрации работоспособности созданного модуля, принимающую на стандартный ввод текст запроса и выводящую на стандартный вывод результирующее дерево разбора или сообщение об ошибке.

# Описание работы

Программа состоит из следующих модулей:

* + parser.y - описание грамматики языка запросов на языке Bison
  + lexer.l - описание лексического анализатора на языке Flex
  + main.c - точка входа в программу
  + query.c - описание структур и методы для их формирования в ходе работы парсера
  + printer.c - вывод получившейся структуры в консоль

# Аспекты реализации

Поддерживаются операции - create, match-set, match-delete, match - для работы с узлами.

**struct query {**

**char var\_name[MAX\_STRING\_SIZE]; char label[MAX\_STRING\_SIZE];**

**enum query\_type type; union {**

**struct create\_query as\_create; struct match\_query as\_match;**

**struct set\_query as\_set;**

**struct delete\_query as\_delete;**

**};**

**};**

Основная структура для описания запроса.

struct match\_query {

struct filter\* filter;

char return\_value[MAX\_STRING\_SIZE];

};

struct delete\_query {

struct filter\* filter;

char delete\_value[MAX\_STRING\_SIZE];

};

struct set\_query {

struct filter\* filter; struct property\* prop;

};

struct create\_query {

struct property\* prop;

};

Описывают более конкретную структуру каждого запроса.

**struct property {**

**char name[MAX\_STRING\_SIZE]; struct value value;**

**struct property\* next\_prop;**

**};**

Описывает поле узла с возможными значениями - bool, int, double, string.

enum property\_operation { EQUALS,

NOT\_EQUALS, LESS\_THAN,

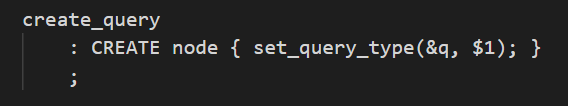
GREATER\_THAN, CONTAINS

};

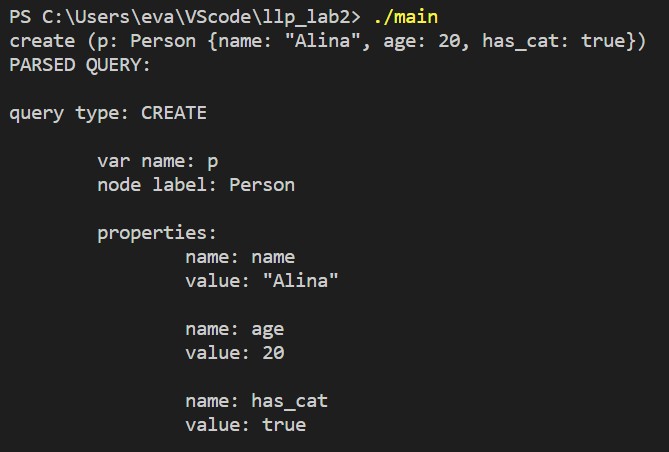
Поддерживаются операции: =, !=, >, <, CONTAINS.

В файле lexer.l описаны правила разбора запроса на токены.

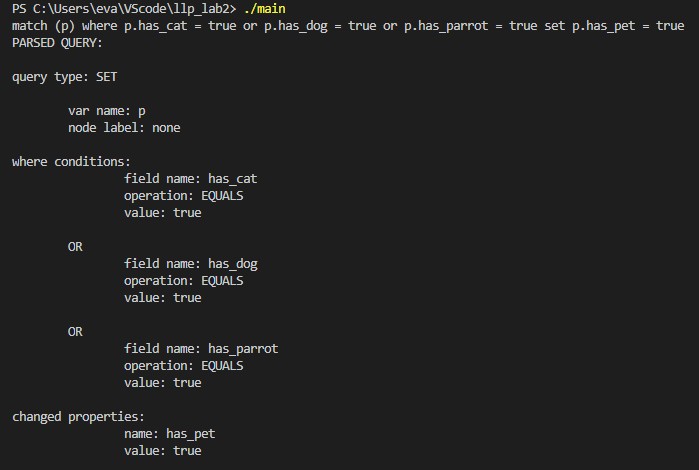
Файл parser.y содержит описание синтаксиса запросов и логику их обработки.

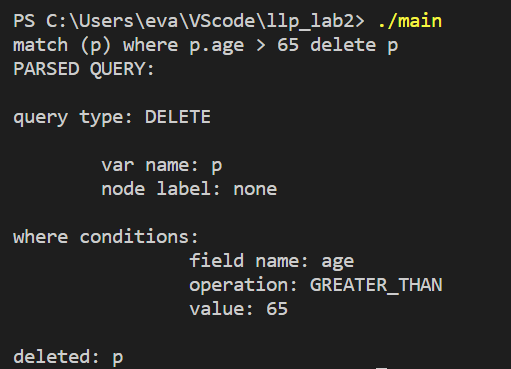
Определенные правила добавляют в структуру query поля. Например:

# Примеры работы программы

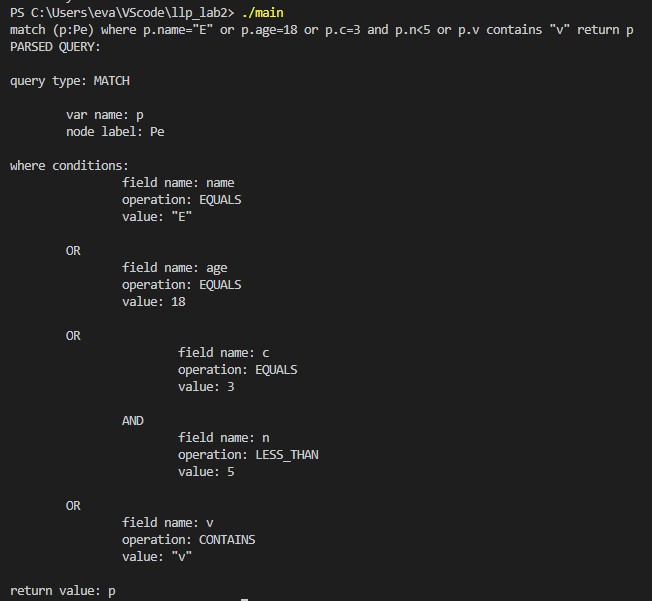
1. **Создание элемента**

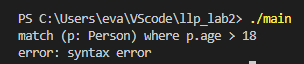
# Обновление элемента



1. **Удаление элемента**

# Выборка элементов



1. **Пример ошибки синтаксиса**

# Выводы

В ходе выполнения лабораторной работы изучены Bison и Flex, а также был реализован модуль для разбора запросов Cypher.