Университет ИТМО

Факультет программной инженерии и компьютерной техники
Направление подготовки 09.03.04 Информатика и вычислительная техника
Дисциплина «Низкоуровневое программирование»

Отчет
По лабораторной работе №2
Вариант 1

Выполнил:
Степанов М.А.
Преподаватель:
Кореньков Ю. Д.

Санкт-Петербург, 2022 г.

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

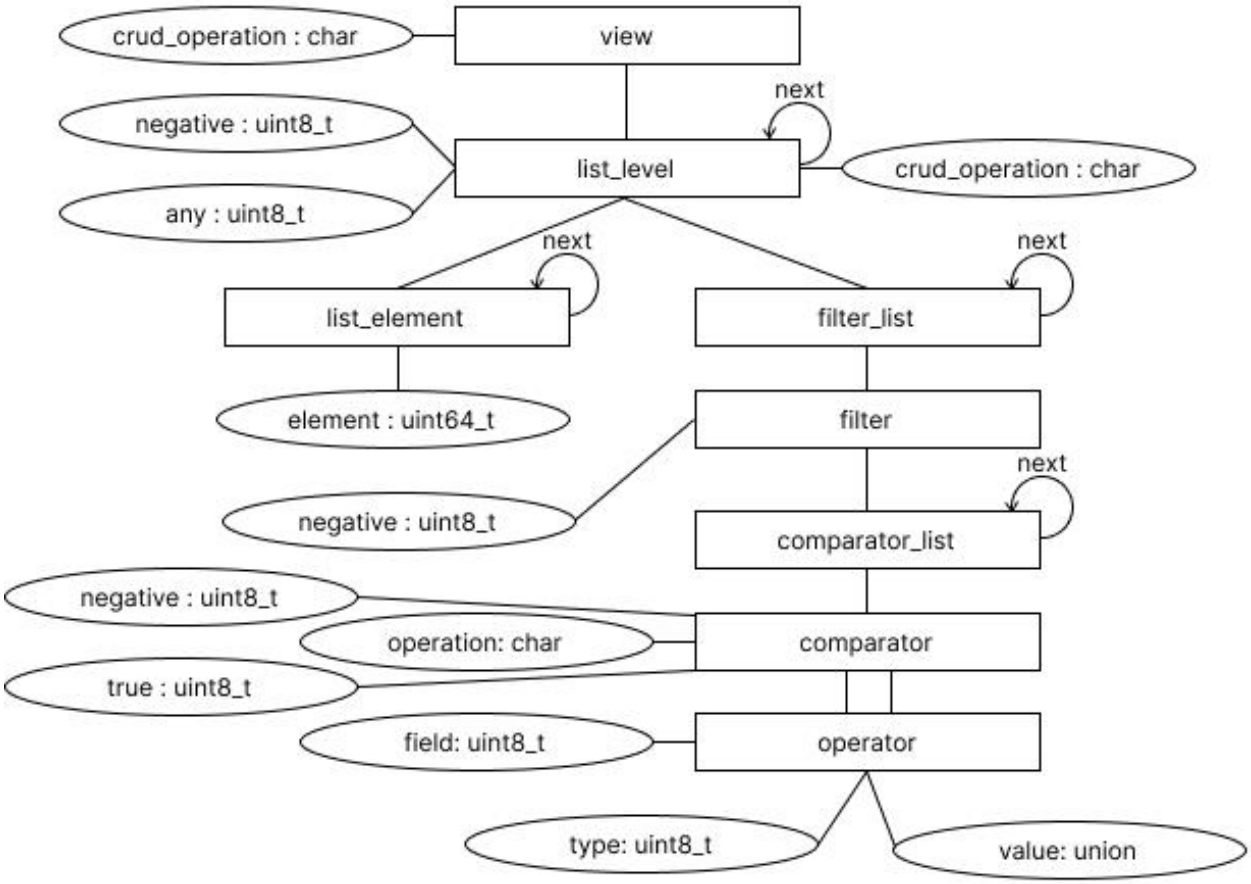
Задачи:

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:

Тестовая программа принимает на стандартные вход один запрос и выводит результат разбора. Программа состоит всего из двух модулей: *structure* – содержит сигнатуры используемых структур; *parser* – функции разбора строки на дерево.

Описание структур (сами структуры описаны в view.h):



Аспекты реализации:

Для выполнения данного задания синтаксис XPath был немного изменен: Добавлено указание операций, переработана система фильтров, чтобы избежать необходимости введения приоритезации операций.

Операции над элементами:

- + -- добавление элемента
- - -- удаление элемента
- ? – поиск элемента
- = -- изменение элемента

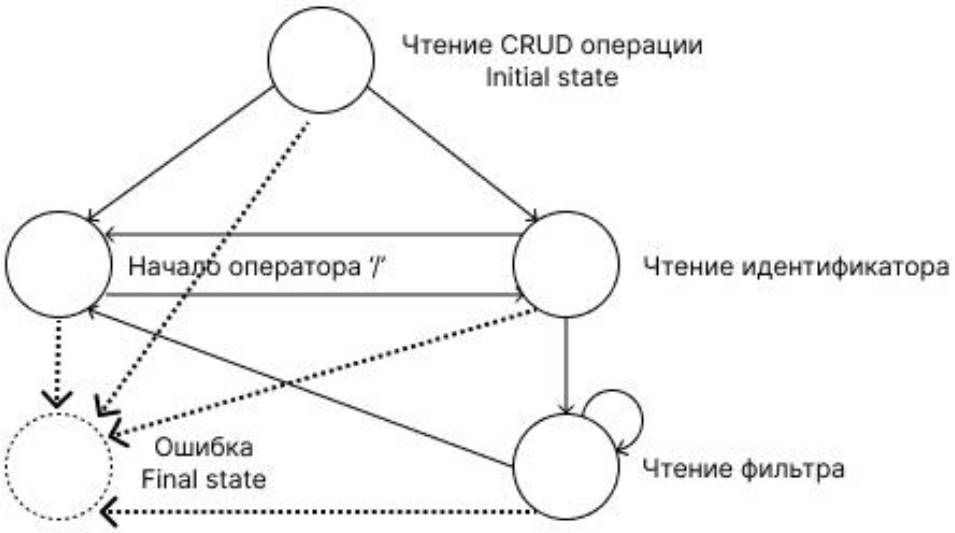
Выборка элементов:

- <tupleId> -- выборка элемента по id
- / -- следующий оператор является дочерним
- * -- указатель на любой элемент
- ! – инвертирует выборку

Предикаты для операторов:

- @ -- true
- ! – инвертировать предикат
- поддерживаются операторы сравнения (>, <, =)
- <field>:<substr> -- поиск по подстроке
- у одного оператора может быть несколько предикатов

Для реализации было принято решение не использовать библиотек синтаксического анализа текста (в виду того, что удалось упростить синтаксис языка до банальной последовательности термов), поэтому обработка ведется через состояния конечного автомата:



Результаты:

+/123[price>12.22|name:'mike][count<10][type=3]

```
------------------------------
OPERATION: +
------------------------------
LEVEL: 1
ROOT RELATION
IS NEGATIVE: 0
ID: 123
-FILTERS
--FILTER: 1
--IS NEGATIVE: 0
---COMPARATORS---
----COMPARATOR: 1
----IS NEGATIVE: 0
----OPERATOR 1: type (IS FIELD)
----OPERATION: =
----OPERATOR 2: 3
----END OF COMPARATOR
----COMPARATOR: 2
----IS NEGATIVE: 0
----OPERATOR 1: count (IS FIELD)
----OPERATION: <
----OPERATOR 2: 10
----END OF COMPARATOR
----COMPARATOR: 3
----IS NEGATIVE: 0
----OPERATOR 1: name (IS FIELD)
----OPERATION: :
----OPERATOR 2: mike
----END OF COMPARATOR
----COMPARATOR: 4
----IS NEGATIVE: 0
----OPERATOR 1: price (IS FIELD)
----OPERATION: >
----OPERATOR 2: 12.220000
----END OF COMPARATOR
---END OF COMPARATORS
--END OF FILTER
-END OF FILTERS
```

View ram: 486
Full ram: 486

Как можно заметить, программа использует оперативную память только для хранение целевой структуры.

Примеры запросов:

Выводы:

- Был реализован модуль производящий синтаксический анализ и разбор запроса XPath (его измененной версии для упрощения запросов).
- Изначально планировалось использовать YACC & LEX для разбора запроса, но после нескольких попыток разобраться в работе YACC и преобразований языка XPath, оказалось, что реализовать собственную машину состоянии намного проще.
- Для хранения неопределенных по размеру данных проще оказалось удобнее использовать непрозрачные типы данных (структура view)/