Лабораторная работа №2 по дисциплине "Низкоуровневое программирование"

Зависимости

- Clang
- CMake 3.24

Цель:

- Реализовать модуль, использующий средство синтаксического анализа для разбора языка запросов

Задачи:

- Изучить выбранное средство синтаксического анализа
- Изучить синтаксис языка запросов и записать спецификацию для средства синтаксического анализа
- Реализовать модуль, использующий средство синтаксического анализа для разбора языка запросов
- Реализовать тестовую программу для демонстрации работоспособности созданного модуля
- Результаты тестирования представить в виде отчёта

Описание работы:

Тестовая программа принимает в стандартный поток ввода один запрос и выводит его синтаксический разбор.

Описание структур:

- Comparator

```
struct comparator {
  uint8_t negative;
  uint8_t true_flag;
  compare operation;
  op *op1;
  op *op2;

  comparator();
};
```

- Comparator list

```
struct comparator_list {
 comparator_list *next;
 comparator *value;

 comparator_list();
};
```

- Field

```
  struct field {
   size_t size;
   char *value;

   field(size_t size, char *value);
  };
```

- Filter

```
  struct filter {
   uint8_t negative;
   comparator_list *comparators;

   explicit filter(uint8_t negative);
  };
```

- Filter list

```
struct filter_list {
 filter_list *next;
 filter *value;

 filter_list();
};
```

- Form

```
struct form {
 enum crud crud_operation;
 struct list_level *tree;

 explicit form(crud crud_operation);
};
```

- List elements

```
struct list_element {
 list_element *next;
 int64_t element;

 explicit list_element(int64_t id);
};
```

- List level

```
struct list_level {
 uint8_t negative;
 uint8_t any;
 parent place;
 list_level *next = nullptr;
 list_element value = list_element(0);
 filter_list *filters = nullptr;

 list_level(uint8_t negative, uint8_t any, parent parent);
};
```

- Operation

```
struct op {
 uint8_t field;
 field_types type;
 types value{};

 op(uint8_t field, enum field_types type, union types value);
};
```

- States

```
enum states{
 S_NEXT = 0,
 S_NAME,
 S_ATTRIBUTE,
 S_ERROR
};
```

Аспекты реализации:

Для выполнения данной лабораторной работы был изменен синтаксис языка запросов XPath:

- Добавление элемента `+`
- Удаление элемента `-`
- Изменение элемента `=`
- Поиск элемента `?`
- Указатель на любой элемент `*`
- Инвертирование выборки `!`
- Истина `@`
- Инвертирование предиката `!`
- У одного оператора может быть много предикатов
- Поддерживаемые операторы сравнения `<`, `>`, `=`
- А так же, поиск по подстроке: `<field>:<substr>`

По причине переработки языка запросов XPath было принято решение не использовать библиотеки синтаксического анализа.

Результаты:

```
+/123[capasity>100|name=Audi][count<25][price=10000]
--------------------------------
OPERATION: +
--------------------------------
LEVEL: 1
ROOT RELATION
IS NEGATIVE: 0
ID: 123
-FILTERS
--FILTER: 1
--IS NEGATIVE: 0
---COMPARATORS---
----COMPARATOR: 1
----IS NEGATIVE: 0
----OPERATOR 1: price (IS FIELD)
----OPERATION: =
----OPERATOR 2: 10000
----END OF COMPARATOR
---END OF COMPARATORS
--END OF FILTER
--FILTER: 2
--IS NEGATIVE: 0
---COMPARATORS---
----COMPARATOR: 1
----IS NEGATIVE: 0
----OPERATOR 1: count (IS FIELD)
----OPERATION: <
----OPERATOR 2: 25
----END OF COMPARATOR
---END OF COMPARATORS
--END OF FILTER
--FILTER: 3
--IS NEGATIVE: 0
---COMPARATORS---
----COMPARATOR: 1
----IS NEGATIVE: 0
----OPERATOR 1: name (IS FIELD)
----OPERATION: =
----OPERATOR 2: Audi (IS FIELD)
----END OF COMPARATOR
----COMPARATOR: 2
----IS NEGATIVE: 0
----OPERATOR 1: capasity (IS FIELD)
```

```
----OPERATION: >
----OPERATOR 2: 100
----END OF COMPARATOR
---END OF COMPARATORS
--END OF FILTER
-END OF FILTERS
--------------------------------
```

Вывод:

В результате выполнения данной лабораторной работы был разработан модуль, реализующий синтаксический анализ и разбор запроса языка XPath без сторонних библиотек, по скольку последоватьность членов быстрее и проще парсить без них.