Санкт-Петербургский Национальный
Исследовательский Университет ИТМО

Факультет Программной Инженерии и Компьютерной Техники

Низкоуровневое программирование

Лабораторная работа №2

Вариант №1 «XPath»

Выполнила:

Руденко Мария, группа P33312

Преподаватель:

Кореньков Юрий Дмитриевич

Санкт-Петербург

2023

Задание

Использовать средство синтаксического анализа по выбору, реализовать модуль для разбора некоторого достаточного подмножества языка запросов по выбору в соответствии с вариантом формы данных. Должна быть обеспечена возможность описания команд создания, выборки, модификации и удаления элементов данных.

Порядок выполнения

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

Моя тестовая программа принимает на вход запрос и выводит результат его обработки. В модуле parser происходит разбор полученной строки на дерево. На выходе работы программы получается разбор запроса в авторском формате дерева.

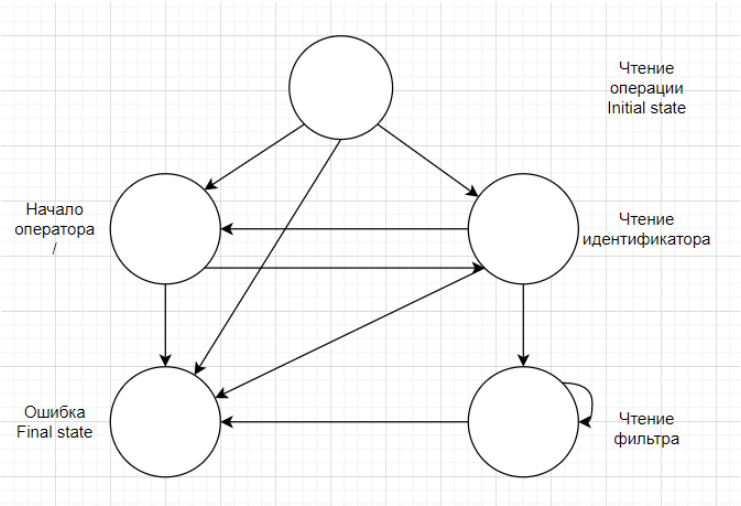
Особенности реализации. Операции над элементами:

1. + добавление элемента
2. – удаление элемента
3. ? поиск по элементам
4. = изменение имеющегося элемента

Предикаты операторов:

1. @ - true
2. !@ - false
3. < меньше
4. > больше
5. = равно
6. <field>:<value> поиск с подстрокой

Обработка термов реализована через последовательность состояний:



Структуры описаны в **structure.h** - директория **include**

```c
#ifndef LAB2_LLP_STRUCTURE_H
#define LAB2_LLP_STRUCTURE_H

#include <stdio.h>
#include <inttypes.h>
#include <stdlib.h>

enum parent {
    PARENT_ROOT = 0,
    PARENT_FREE,
    PARENT_RELATIVE
};

enum crud {
    CRUD_INSERT = '+',
    CRUD_DELETE = '-',
    CRUD_FIND = '?',
    CRUD_UPDATE = '='
};

enum compare {
    COMPARE_EQUAL = '=',
    COMPARE_LESS = '<',
    COMPARE_GREATER = '>',
    COMPARE_SUBSTR = ':'
};

enum field_types {
    STRING_TYPE = 0,
    INTEGER_TYPE,
    FLOAT_TYPE,
    BOOLEAN_TYPE
};

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

union types {
    struct field *string;
    int64_t integer;
    int64_t boolean;
    double real;
} ;

void *pr_malloc(size_t size_of);
```

```
void *print_memory();

struct operator {
    uint8_t field;
    enum field_types type;
    union types value;
};

struct comparator {
    uint8_t negative;
    uint8_t true;
    enum compare operation;
    struct operator *operation_1;
    struct operator *operation_2;
};

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

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

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

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

struct list_level {
    uint8_t negative;
    uint8_t any;
    enum parent place;
    struct list_level *next;
    struct list_element *value;
    struct filter_list *filters;
};

struct view {
    enum crud crud_operation;
    struct list_level *tree_lvl;
};

enum states{
    STATE_NEXT = 0,
    STATE_NAME,
    STATE_ATTRIBUTE,
    STATE_ERROR
};

struct view *create_view(enum crud crud_op);

struct list_level *create_list_level(uint8_t negative, uint8_t any, enum parent
place);

struct list_element *create_list_element(int64_t id);

struct filter_list *create_filter_list();

struct filter *create_filter(uint8_t negative);

struct comparator_list *create_comparator_list();

struct comparator *create_comparator();
```

```
struct operator *create_operator(uint8_t field, enum field_types type, union types
value);

struct field *create_field(size_t size, char *value);

#endif //LAB2_LLP_STRUCTURE_H
```

Пример работы программы

```
+/1[name=masha][type=1]
OPERATION: +
LEVEL: 1
ROOT RELATION
IS NEGATIVE: 0
ID: 1
 FILTERS
  FILTER: 1
  IS NEGATIVE: 0
   COMPARATORS
    COMPARATOR: 1
    IS NEGATIVE: 0
    OPERATOR 1: type (FIELD)
    OPERATION: =
    OPERATOR 2: 1
    END COMPARATOR
   END COMPARATORS
  END FILTER
  FILTER: 2
  IS NEGATIVE: 0
   COMPARATORS
    COMPARATOR: 1
    IS NEGATIVE: 0
    OPERATOR 1: name (FIELD)
    OPERATION: =
    OPERATOR 2: masha (FIELD)
    END COMPARATOR
   END COMPARATORS
  END FILTER
```

```
END FILTERS

memory usage: 225

Process finished with exit code 0
```

```
-/7[count<23|type>17][name=masha]
OPERATION: -
LEVEL: 1
ROOT RELATION
IS NEGATIVE: 0
ID: 7
 FILTERS
  FILTER: 1
  IS NEGATIVE: 0
   COMPARATORS
    COMPARATOR: 1
    IS NEGATIVE: 0
    OPERATOR 1: name (FIELD)
    OPERATION: =
    OPERATOR 2: masha (FIELD)
    END COMPARATOR
   END COMPARATORS
  END FILTER
  FILTER: 2
  IS NEGATIVE: 0
   COMPARATORS
    COMPARATOR: 1
    IS NEGATIVE: 0
    OPERATOR 1: type (FIELD)
    OPERATION: >
    OPERATOR 2: 17
    END COMPARATOR
    COMPARATOR: 2
    IS NEGATIVE: 0
    OPERATOR 1: count (FIELD)
    OPERATION: <
    OPERATOR 2: 23
    END COMPARATOR
   END COMPARATORS
  END FILTER
 END FILTERS

 memory usage: 294

Process finished with exit code 0
```

Пример некорректного запроса

```
+/5[name=masha][[year=2023]
--Syntax error--
```

Выводы:

- ✓ По использованию оперативной памяти: программа использует её только для хранения целевой структуры.
- ✓ В результате выполнения данной лабораторной работы был реализован синтаксический анализатор для запросов XPath в переработанной версии (упрощение запросов).
- ✓ Была реализована работа состояний без использования внешних библиотек, потому что с ними код был бы безосновательно перегружен.