Институт приборостроения, автоматизации и информационных технологий

Кафедра информационных систем и цифровых технологий

Дисциплина «Теория языков программирования и методы трансляции»

Отчет к лабораторной работе № 5

«Синтаксический разбор снизу вверх»

Выполнил:

Василения Иван Валерьевич

Мельниченко Артём Олегович

Принял:

Гордиенко А. П.

Орёл, 2024г

Замыкания:

I0: {

accept -> • goal eof

goal -> • 'SELECT' attr\_list 'FROM' string where\_case

attr\_list -> • string

attr\_list -> • string comma attr\_list

where\_case -> • 'WHERE' condition\_list

where\_case -> • э

condition\_list -> • condition

condition\_list -> • condition 'AND' condition\_list

condition\_list -> • condition 'OR' condition\_list

condition -> • string comparison num

}

I1: {

goal -> 'SELECT' • attr\_list 'FROM' string where\_case

attr\_list -> • string

attr\_list -> • string comma attr\_list

}

I2: {

accept -> goal • eof

}

I3: {

attr\_list -> string •

attr\_list -> string • comma attr\_list

}

I4: {

goal -> 'SELECT' attr\_list • 'FROM' string where\_case

}

I5: {

accept -> goal eof •

}

I6: {

attr\_list -> string comma • attr\_list

attr\_list -> • string

attr\_list -> • string comma attr\_list

}

I7: {

goal -> 'SELECT' attr\_list 'FROM' • string where\_case

}

I8: {

attr\_list -> string comma attr\_list •

}

I9: {

'SELECT' attr\_list 'FROM' string • where\_case

}

I10: {

where\_case -> 'WHERE' • condition\_list

condition\_list -> • condition

condition\_list -> • condition 'AND' condition\_list

condition\_list -> • condition 'OR' condition\_list

condition -> • string comparison num

}

I11: {

goal -> 'SELECT' attr\_list 'FROM' string where\_case •

}

I12: {

condition -> string • comparison num

}

I13: {

where\_case -> 'WHERE' condition\_list •

}

I14: {

condition\_list -> condition •

condition\_list -> condition • 'AND' condition\_list

condition\_list -> condition • 'OR' condition\_list

}

I15: {

condition -> string comparison • num

}

I16: {

condition\_list -> condition 'AND' • condition\_list

}

I17: {

condition\_list -> condition 'OR' • condition\_list

}

I18: {

condition -> string comparison num •

}

I19: {

condition\_list -> condition 'AND' condition\_list •

}

I20: {

condition\_list -> condition 'OR' condition\_list •

}

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Action | eof | SELECT | string | FROM | WHERE | comparison | num | comma | AND | OR |
| S0 |  | SHIFT S1 |  |  |  |  |  |  |  |  |
| S1 |  |  | SHIFT S3 |  |  |  |  |  |  |  |
| S2 | SHIFT S5 |  |  |  |  |  |  |  |  |  |
| S3 |  |  |  | **REDUCE 2** |  |  |  | SHIFT S6 |  |  |
| S4 |  |  |  | SHIFT S7 |  |  |  |  |  |  |
| S5 | ACCEPT | ACCEPT | ACCEPT | ACCEPT | ACCEPT | ACCEPT | ACCEPT | ACCEPT | ACCEPT | ACCEPT |
| S6 |  |  | SHIFT S3 |  |  |  |  |  |  |  |
| S7 |  |  | SHIFT S9 |  |  |  |  |  |  |  |
| S8 | **REDUCE 3** | **REDUCE 3** | **REDUCE 3** | **REDUCE 3** | **REDUCE 3** | **REDUCE 3** | **REDUCE 3** | **REDUCE 3** | **REDUCE 3** | **REDUCE 3** |
| S9 | **REDUCE 5** | **REDUCE 5** | **REDUCE 5** | **REDUCE 5** | SHIFT S10 | **REDUCE 5** | **REDUCE 5** | **REDUCE 5** | **REDUCE 5** | **REDUCE 5** |
| S10 |  |  | SHIFT S12 |  |  |  |  |  |  |  |
| S11 | **REDUCE 1** | **REDUCE 1** | **REDUCE 1** | **REDUCE 1** | **REDUCE 1** | **REDUCE 1** | **REDUCE 1** | **REDUCE 1** | **REDUCE 1** | **REDUCE 1** |
| S12 |  |  |  |  |  | SHIFT S15 |  |  |  |  |
| S13 | **REDUCE 4** | **REDUCE 4** | **REDUCE 4** | **REDUCE 4** | **REDUCE 4** | **REDUCE 4** | **REDUCE 4** | **REDUCE 4** | **REDUCE 4** | **REDUCE 4** |
| S14 | **REDUCE 6** | **REDUCE 6** | **REDUCE 6** | **REDUCE 6** | **REDUCE 6** | **REDUCE 6** | **REDUCE 6** | **REDUCE 6** | SHIFT S16 | SHIFT 17 |
| S15 |  |  |  |  |  |  | SHIFT S18 |  |  |  |
| S16 |  |  | SHIFT S12 |  |  |  |  |  |  |  |
| S17 |  |  | SHIFT S12 |  |  |  |  |  |  |  |
| S18 | **REDUCE 9** | **REDUCE 9** | **REDUCE 9** | **REDUCE 9** | **REDUCE 9** | **REDUCE 9** | **REDUCE 9** | **REDUCE 9** | **REDUCE 9** | **REDUCE 9** |
| S19 | **REDUCE 7** | **REDUCE 7** | **REDUCE 7** | **REDUCE 7** | **REDUCE 7** | **REDUCE 7** | **REDUCE 7** | **REDUCE 7** | **REDUCE 7** | **REDUCE 7** |
| S20 | **REDUCE 8** | **REDUCE 8** | **REDUCE 8** | **REDUCE 8** | **REDUCE 8** | **REDUCE 8** | **REDUCE 8** | **REDUCE 8** | **REDUCE 8** | **REDUCE 8** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| GOTO | goal | attr\_list | where\_case | condition\_list | condition |
| S0 | S2 |  |  |  |  |
| S1 |  | S4 |  |  |  |
| S2 |  |  |  |  |  |
| S3 |  |  |  |  |  |
| S4 |  |  |  |  |  |
| S5 |  |  |  |  |  |
| S6 |  | S8 |  |  |  |
| S7 |  |  |  |  |  |
| S8 |  |  |  |  |  |
| S9 |  |  | S11 |  |  |
| S10 |  |  |  | S13 | S14 |
| S11 |  |  |  |  |  |
| S12 |  |  |  |  |  |
| S13 |  |  |  |  |  |
| S14 |  |  |  |  |  |
| S15 |  |  |  |  |  |
| S16 |  |  |  | S19 | S14 |
| S17 |  |  |  | S20 | S14 |
| S18 |  |  |  |  |  |
| S19 |  |  |  |  |  |
| S20 |  |  |  |  |  |

Листинг анализатора:

#ifndef ANALYZER\_H

#define ANALYZER\_H

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#include "tokens.h"

#include "stack.h"

#include "parse\_tables.h"

#define DEBUG\_MODE true

#define \_EOF 0

extern int yylex();

extern char\* yytext;

pair\_stack \*parse\_stack;

void reduce(); void shift();

void drop\_parse\_error() {

    printf("\033[31mParse error!!!!!!\033[0m\n");

    exit(EXIT\_FAILURE);

}

void show\_pair\_stack(pair\_stack\* s) {

    if (!DEBUG\_MODE) return;

    printf("Stack content: ");

    for (int i = 0; i <= s->head\_num; i++) {

        printf("\033[35m[St: %d, El: %d], \033[0m", s->pair[i].state, s->pair[i].token);

    }

    printf("\n");

}

bool is\_terminal(int tok) { return tok < NON\_TERMINAL\_TYPE\_BASE; }

bool is\_non\_terminal(int tok) { return tok >= NON\_TERMINAL\_TYPE\_BASE && tok < STATES\_TYPE\_BASE; }

bool is\_states\_type(int tok) { return tok >= STATES\_TYPE\_BASE && tok < REDUCE\_TYPE\_BASE; }

bool is\_reduce\_type(int tok) { return tok >= REDUCE\_TYPE\_BASE && tok < ACC; }

bool is\_accept(int tok) { return tok == ACC; }

void init() {

    parse\_stack = pair\_stack\_alloc();

}

void update\_curr\_state() {

}

int curr\_token;

void parse\_start() {

    init();

    pair\_stack\_push\_pair(parse\_stack, S0, \_EOF);

    curr\_token = yylex();

    while (1) {

        int curr\_state = pair\_stack\_get\_top\_state(parse\_stack);

        int action = action\_table[curr\_state - STATES\_TYPE\_BASE][curr\_token];

        show\_pair\_stack(parse\_stack);

        printf("Current action: %d\n", action);

        printf("Action cell: %d, %d\n", curr\_state - STATES\_TYPE\_BASE, curr\_token);

        printf("lex: %s\n", yytext);

        if (is\_accept(action)) {

            printf("\033[32mР’С‹СЂР°Р¶РµРЅРёРµ РїСЂРёРЅР°РґР»РµР¶РёС‚ СЏР·С‹РєСѓ!!!!!!!!!!!!!!!!!!!!!!!\033[0m\n");

            break;

        }

        else if (is\_states\_type(action)) {

            pair\_stack\_push\_pair(parse\_stack, action, curr\_token);

            curr\_token = yylex();

        }

        else if (is\_reduce\_type(action)) {

            int rule = action - REDUCE\_TYPE\_BASE;

            printf("rule %d\n", rule);

            for (int i = 0; i < rules[rule][RULE\_LEN]; i++) {

                pair\_stack\_pop\_pair(parse\_stack);

            }

            int curr\_state = pair\_stack\_get\_top\_state(parse\_stack);

            printf("Cell %d, %d\n", curr\_state - STATES\_TYPE\_BASE, rules[rule][RULE\_RES] - NON\_TERMINAL\_TYPE\_BASE);

            int new\_state = goto\_table[curr\_state - STATES\_TYPE\_BASE][rules[rule][RULE\_RES] - NON\_TERMINAL\_TYPE\_BASE];

            pair\_stack\_push\_pair(parse\_stack, new\_state, rules[rule][RULE\_RES]);

        }

        else {

            drop\_parse\_error();

        }

    }

}

void shift() {

    curr\_token = yylex();

}

void reduce() {

}

#endif