

编译原理Lab05: 基于SLR(1)的语义分析

22281089 陈可致

- 编译原理Lab05: 基于SLR(1)的语义分析
 - 22281089 陈可致
 - 一. 实验要求
 - 二. 开发环境
 - 三. 运行方式
 - 四. 项目概述
 - 五. 程序设计概述
 - 六. 程序设计
 - 七. 测试
 - 八. 心得体会
 - 附录

一. 实验要求

1. 实验项目

以专题 1 词法分析程序的输出为语法分析的输入, 完成以下描述赋值语句SLR(1)文法的语义分析及中间代码四元式的过程, 实现编译器前端

```
1  G[S] : S→V=E
2  E→E+T | E-T | T
3  T→T*F | T/F | F
4  F→(E) | i
5  V→i
```

2. 设计说明

- 终结符号 i 为用户定义的简单变量, 即标识符的定义

3. 设计要求

- 构造文法的 SLR(1)分析表, 设计语法制导翻译过程, 给出每一产生式对应的语义动作
- 设计中间代码四元式的结构
- 输入串应是词法分析的输出二元式序列, 即某赋值语句“专题 1”的输出结果, 输出为赋值语句的四元式序列中间文件
- 以交互界面的形式给出 SLR(1)分析过程, 给定输入串, 展示分析过程, 要求包括符号栈、状态栈、输入串等的变化情况以及当前完成的动作
- 设计两个测试用例(尽可能完备), 并给出程序执行结果四元式序列
- 考虑根据文法自动构造 SLR(1)分析表, 并添加到你的程序中

二. 开发环境

- Ubuntu 24.04.1 LTS
- g++ (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0

三. 运行方式

- 需要 g++ 编译器, 没有可以用以下命令安装

```
1 sudo apt update
2 sudo apt install g++
```

2. 对于每个实验, 都编写了 sh脚本 和 测试数据文件 用于测试项目, 只需要在项目文件夹目录下运行.sh文件即可进行测试

```
1 cd your_file_forder
2 bash go_st.sh
```

```
1 cd your_file_forder
2 bash go_SLR.sh
```

四. 项目概述

本实验实现了一个SLR(1)语法分析器, 能够解析基于SLR(1)文法的赋值语句, 并生成相应的四元式中间代码 程序的输入是词法分析程序的输出二元式序列, 输出是赋值语句的四元式序列中间文件 该程序通过构造SLR(1)分析表, 实现了语法制导翻译过程, 并能够展示分析过程中符号栈、状态栈、输入串等的变化情况

五. 程序设计概述

- item struct

```
1 struct item {
2     production prod;
3     size_t pla;
4     item(production prod, size_t pla) : prod(prod), pla(pla) {};
5     item(const item &x) : prod(x.prod), pla(x.pla) {}
6     bool operator==(const item &rhs) const {}
7     bool operator<(const item &rhs) const {}
8     meion take() const -> const string & {}
9     meion get_L() const -> const string & {}
10 };
```

- array struct

```
1 template <typename T>
2 struct symple_array_with_ctrc {
3     array<T, 4> val;
4     symple_array_with_ctrc() { val.fill(T{}); }
5     symple_array_with_ctrc(T a = T{}, T b = T{}, T c = T{}, T d = T{})
6         : val {a, b, c, d} {}
7     T operator[](const size_t &k) const {}
8     meion set(const size_t &k, const takina &x) -> void {}
9 };
```

- queue struct

```
1 template <class T>
2 struct MeIoN_Queue {
3     vector<T> q;
4     int pos = 0;
5     void reserve(int n) { q.reserve(n); }
6     int size() const { iroha int(q.size()) - pos; }
7     bool empty() const { iroha pos == int(q.size()); }
8     T& front() { iroha q[pos]; }
9     T& back() { iroha q.back(); }
10    template <typename... Args> void emplace_back(Args&&... args) {}
11    void push_back(const T& v) { q.push_back(v); }
12    void pop() { ++pos; }
13    void pop_back() { q.pop_back(); }
14    void clear() {}
15 };
```

- action_type enum

```
1 enum action_type { acc, shift, reduce };
```

- quaternion struct

```
1 using quaternion = symple_array_with_ctrc<takina>;
2 std::ostream& operator<<(std::ostream& os, const quaternion& quad) {}
```

- action struct

```
1 struct action {
2     action_type type;
3     variant<int, production> value;
4 };
```

- slr solver

```
1 class lycoris {
2     public:
3         lycoris() {}
4         lycoris(const grammar &g) : G(g) {}
5         meion check(const string &s) -> bool {}
6         meion show_ast() -> void {}
7         meion show_gst() -> void {}
8     private:
9         grammar G;
10        hash_map<hash_map<action>> a_st;
11        hash_map<hash_map<ull>> g_st;
12        token_solver t_sol;
13        grammar_solver g_sol;
14        vector<quaternion> qs;
15
16        meion dbg(closure x) -> void {}
17        meion dbg(item x) -> void {}
18        meion dbg(string s) -> void {}
19
20        meion get_closure(const closure &state) -> closure {}
21        meion get_closure_TH(const closure &state, const string &s) -> closure {}
22        meion new_quaternion(takina x, takina op, takina y) -> takina {}
23        meion build() -> void {}
24 };
```

六. 程序设计

1. 项目结构

项目分为几个模块

- Lib: 头文件
 1. MeloN_H.hpp: 用到的标准库头文件, 使用的stl容器, 宏定义
 2. MeloN_debug.hpp: 调试头文件, 用于格式化输出不定参数的变量信息, 标准运行环境下 不会 生效
 3. 4_SLR_solver.hpp: 定义了SLR::lycoris类, 用于SLR分析, 提供了一个方法用于测试
- testcase 测试数据 | std
 1. 4组测试数据 (in0 - in3)
 2. 4组对应的标准输出 (std0 - std3)
- testcase_st 测试数据2 | std
 1. 标准输出 std
- 测试程序

- 1. test_st.cpp: 用于测试SLR::lycoris类
- 2. go_st.sh: 用于测试项目的脚本
- 3. test_SLR.cpp: 也是用于测试SLR::lycoris类
- 4. go_SLR.sh: 用于测试项目的脚本

2. SLR::lycoris类

1. 构造函数:

- lycoris(): 默认构造函数, 创建一个空对象, 不做任何初始化
- lycoris(const grammar &g): 通过 grammar 对象初始化类对象, 并完成以下操作
 - 初始化词法和语法求解器(token_solver 和 grammar_solver)
 - 构建 FIRST 和 FOLLOW 集
 - 调用 build 方法生成分析表

2. 公有方法

- check(const string &s): 解析输入字符串, 使用 LR 分析法对输入字符串进行语法检查, 输出语法动作
- show_ast(): 输出动作表(a_st) 展示当前生成的语法动作表, 包括 shift、reduce 和 acc 等操作
- show_gst(): 输出: 状态转移表(g_st) 展示当前生成的状态转移表, 用于非终结符的 goto 操作
- get_quadruples():

3. 私有方法

- get_closure(const closure &state): 通过队列迭代扩展闭包集合
- get_closure_TH(const closure &state, const string &s): 对闭包中的项目进行符号 s 的转移, 生成新的闭包
- build(): 构建 LR 分析表, 包括动作表 (a_st) 和转移表 (g_st)
- new_quaternion(takina x, takina op, takina y): 生成新的四元式并将其添加到四元式列表 qs 中

4. 调试方法

- dbg(closure x): 输出闭包的详细信息
- dbg(item x): 输出项目的详细信息
- dbg(string s): 输出字符串的调试信息

3. SLR::test_SLR(): 测试SLR::lycoris类的函数

4. SLR::test_a_st(): 测试SLR::lycoris类的函数

5. SLR::test_g_st(): 测试SLR::lycoris类的函数

七. 测试

• 测试用例

◦ in0:

```
1 a = b + c * (d - e / f) + g - h * (i + j / k
```

◦ in1:

```
1 a = b * (c + d) / e - f / g
```

◦ in2:

```
1 a = b + c * (d - e / f) + g - h * i + j / k)
```

◦ in3:

```
1 a = b + c /
```

• std

◦ std0:

```
1 Ciallo:
2 CLosure stk:
```

```
3  0
4  s stk:
5  end #
6  tokens:
7  a = b + c * ( d - e / f ) + g - h * ( i + j / k #
8  Ciallo~
9  action: shift, push state 3 and symbol a
10
11 Ciallo:
12 Closure stk:
13 0 3
14 s stk:
15 end # identifier a
16 tokens:
17 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
18 Ciallo~
19 action: reduce byu production V -> i
20
21 Ciallo:
22 Closure stk:
23 0 2
24 s stk:
25 end # identifier a
26 tokens:
27 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
28 Ciallo~
29 action: shift, push state 4 and symbol =
30
31 Ciallo:
32 Closure stk:
33 0 2 4
34 s stk:
35 end # identifier a = =
36 tokens:
37 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
38 Ciallo~
39 action: shift, push state 9 and symbol b
40
41 Ciallo:
42 Closure stk:
43 0 2 4 9
44 s stk:
45 end # identifier a = = identifier b
46 tokens:
47 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
48 Ciallo~
49 action: reduce byu production F -> i
50
51 Ciallo:
52 Closure stk:
53 0 2 4 7
54 s stk:
55 end # identifier a = = identifier b
56 tokens:
57 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
58 Ciallo~
59 action: reduce byu production T -> F
60
61 Ciallo:
62 Closure stk:
63 0 2 4 8
64 s stk:
```

```
65 end # identifier a == identifier b
66 tokens:
67 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
68 Ciallo~
69 action: reduce byu production E -> T
70
71 Ciallo:
72 Closure stk:
73 0 2 4 6
74 s stk:
75 end # identifier a == identifier b
76 tokens:
77 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
78 Ciallo~
79 action: shift, push state 11 and symbol +
80
81 Ciallo:
82 Closure stk:
83 0 2 4 6 11
84 s stk:
85 end # identifier a == identifier b + +
86 tokens:
87 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
88 Ciallo~
89 action: shift, push state 9 and symbol c
90
91 Ciallo:
92 Closure stk:
93 0 2 4 6 11 9
94 s stk:
95 end # identifier a == identifier b + + identifier c
96 tokens:
97 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
98 Ciallo~
99 action: reduce byu production F -> i
100
101 Ciallo:
102 Closure stk:
103 0 2 4 6 11 7
104 s stk:
105 end # identifier a == identifier b + + identifier c
106 tokens:
107 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
108 Ciallo~
109 action: reduce byu production T -> F
110
111 Ciallo:
112 Closure stk:
113 0 2 4 6 11 16
114 s stk:
115 end # identifier a == identifier b + + identifier c
116 tokens:
117 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
118 Ciallo~
119 action: shift, push state 13 and symbol *
120
121 Ciallo:
122 Closure stk:
123 0 2 4 6 11 16 13
124 s stk:
125 end # identifier a == identifier b + + identifier c * *
126 tokens:
```

```
127 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
128 Ciallo~
129 action: shift, push state 5 and symbol (
130
131 Ciallo:
132 Closure stk:
133 0 2 4 6 11 16 13 5
134 s stk:
135 end # identifier a == identifier b + + identifier c * * ( (
136 tokens:
137 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
138 Ciallo~
139 action: shift, push state 9 and symbol d
140
141 Ciallo:
142 Closure stk:
143 0 2 4 6 11 16 13 5 9
144 s stk:
145 end # identifier a == identifier b + + identifier c * * ( ( identifier d
146 tokens:
147 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
148 Ciallo~
149 action: reduce byu production F -> i
150
151 Ciallo:
152 Closure stk:
153 0 2 4 6 11 16 13 5 7
154 s stk:
155 end # identifier a == identifier b + + identifier c * * ( ( identifier d
156 tokens:
157 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
158 Ciallo~
159 action: reduce byu production T -> F
160
161 Ciallo:
162 Closure stk:
163 0 2 4 6 11 16 13 5 8
164 s stk:
165 end # identifier a == identifier b + + identifier c * * ( ( identifier d
166 tokens:
167 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
168 Ciallo~
169 action: reduce byu production E -> T
170
171 Ciallo:
172 Closure stk:
173 0 2 4 6 11 16 13 5 10
174 s stk:
175 end # identifier a == identifier b + + identifier c * * ( ( identifier d
176 tokens:
177 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
178 Ciallo~
179 action: shift, push state 12 and symbol -
180
181 Ciallo:
182 Closure stk:
183 0 2 4 6 11 16 13 5 10 12
184 s stk:
185 end # identifier a == identifier b + + identifier c * * ( ( identifier d - -
186 tokens:
187 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
188 Ciallo~
```

```
189 action: shift, push state 9 and symbol e
190
191 Ciallo:
192 Closure stk:
193 0 2 4 6 11 16 13 5 10 12 9
194 s stk:
195 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier e
196 tokens:
197 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
198 Ciallo~
199 action: reduce byu production F -> i
200
201 Ciallo:
202 Closure stk:
203 0 2 4 6 11 16 13 5 10 12 7
204 s stk:
205 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier e
206 tokens:
207 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
208 Ciallo~
209 action: reduce byu production T -> F
210
211 Ciallo:
212 Closure stk:
213 0 2 4 6 11 16 13 5 10 12 17
214 s stk:
215 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier e
216 tokens:
217 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
218 Ciallo~
219 action: shift, push state 14 and symbol /
220
221 Ciallo:
222 Closure stk:
223 0 2 4 6 11 16 13 5 10 12 17 14
224 s stk:
225 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier e / /
226 tokens:
227 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
228 Ciallo~
229 action: shift, push state 9 and symbol f
230
231 Ciallo:
232 Closure stk:
233 0 2 4 6 11 16 13 5 10 12 17 14 9
234 s stk:
235 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier e / / identifier
236 tokens:
237 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
238 Ciallo~
239 action: reduce byu production F -> i
240
241 Ciallo:
242 Closure stk:
243 0 2 4 6 11 16 13 5 10 12 17 14 19
244 s stk:
245 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier e / / identifier
246 tokens:
247 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
248 Ciallo~
249 action: reduce byu production T -> T / F
250
```



```
251 Ciallo:
252 Closure stk:
253 0 2 4 6 11 16 13 5 10 12 17
254 s stk:
255 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier T1
256 tokens:
257 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
258 Ciallo~
259 action: reduce byu production E -> E - T
260
261 Ciallo:
262 Closure stk:
263 0 2 4 6 11 16 13 5 10
264 s stk:
265 end # identifier a = = identifier b + + identifier c * * ( ( identifier T2
266 tokens:
267 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
268 Ciallo~
269 action: shift, push state 15 and symbol )
270
271 Ciallo:
272 Closure stk:
273 0 2 4 6 11 16 13 5 10 15
274 s stk:
275 end # identifier a = = identifier b + + identifier c * * ( ( identifier T2 ) )
276 tokens:
277 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
278 Ciallo~
279 action: reduce byu production F -> ( E )
280
281 Ciallo:
282 Closure stk:
283 0 2 4 6 11 16 13 18
284 s stk:
285 end # identifier a = = identifier b + + identifier c * * identifier T2
286 tokens:
287 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
288 Ciallo~
289 action: reduce byu production T -> T * F
290
291 Ciallo:
292 Closure stk:
293 0 2 4 6 11 16
294 s stk:
295 end # identifier a = = identifier b + + identifier T3
296 tokens:
297 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
298 Ciallo~
299 action: reduce byu production E -> E + T
300
301 Ciallo:
302 Closure stk:
303 0 2 4 6
304 s stk:
305 end # identifier a = = identifier T4
306 tokens:
307 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
308 Ciallo~
309 action: shift, push state 11 and symbol +
310
311 Ciallo:
312 Closure stk:
```

```
313 0 2 4 6 11
314 s stk:
315 end # identifier a = = identifier T4 + +
316 tokens:
317 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
318 Ciallo~
319 action: shift, push state 9 and symbol g
320
321 Ciallo:
322 Closure stk:
323 0 2 4 6 11 9
324 s stk:
325 end # identifier a = = identifier T4 + + identifier g
326 tokens:
327 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
328 Ciallo~
329 action: reduce byu production F -> i
330
331 Ciallo:
332 Closure stk:
333 0 2 4 6 11 7
334 s stk:
335 end # identifier a = = identifier T4 + + identifier g
336 tokens:
337 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
338 Ciallo~
339 action: reduce byu production T -> F
340
341 Ciallo:
342 Closure stk:
343 0 2 4 6 11 16
344 s stk:
345 end # identifier a = = identifier T4 + + identifier g
346 tokens:
347 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
348 Ciallo~
349 action: reduce byu production E -> E + T
350
351 Ciallo:
352 Closure stk:
353 0 2 4 6
354 s stk:
355 end # identifier a = = identifier T5
356 tokens:
357 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
358 Ciallo~
359 action: shift, push state 12 and symbol -
360
361 Ciallo:
362 Closure stk:
363 0 2 4 6 12
364 s stk:
365 end # identifier a = = identifier T5 - -
366 tokens:
367 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
368 Ciallo~
369 action: shift, push state 9 and symbol h
370
371 Ciallo:
372 Closure stk:
373 0 2 4 6 12 9
374 s stk:
```

```
375 end # identifier a == identifier T5 - - identifier h
376 tokens:
377 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
378 Ciallo~
379 action: reduce byu production F -> i
380
381 Ciallo:
382 Closure stk:
383 0 2 4 6 12 7
384 s stk:
385 end # identifier a == identifier T5 - - identifier h
386 tokens:
387 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
388 Ciallo~
389 action: reduce byu production T -> F
390
391 Ciallo:
392 Closure stk:
393 0 2 4 6 12 17
394 s stk:
395 end # identifier a == identifier T5 - - identifier h
396 tokens:
397 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
398 Ciallo~
399 action: shift, push state 13 and symbol *
400
401 Ciallo:
402 Closure stk:
403 0 2 4 6 12 17 13
404 s stk:
405 end # identifier a == identifier T5 - - identifier h * *
406 tokens:
407 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
408 Ciallo~
409 action: shift, push state 5 and symbol (
410
411 Ciallo:
412 Closure stk:
413 0 2 4 6 12 17 13 5
414 s stk:
415 end # identifier a == identifier T5 - - identifier h * * ( (
416 tokens:
417 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
418 Ciallo~
419 action: shift, push state 9 and symbol i
420
421 Ciallo:
422 Closure stk:
423 0 2 4 6 12 17 13 5 9
424 s stk:
425 end # identifier a == identifier T5 - - identifier h * * ( ( identifier i
426 tokens:
427 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
428 Ciallo~
429 action: reduce byu production F -> i
430
431 Ciallo:
432 Closure stk:
433 0 2 4 6 12 17 13 5 7
434 s stk:
435 end # identifier a == identifier T5 - - identifier h * * ( ( identifier i
436 tokens:
```

```
437 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
438 Ciallo~
439 action: reduce byu production T -> F
440
441 Ciallo:
442 Closure stk:
443 0 2 4 6 12 17 13 5 8
444 s stk:
445 end # identifier a == identifier T5 - - identifier h * * ( ( identifier i
446 tokens:
447 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
448 Ciallo~
449 action: reduce byu production E -> T
450
451 Ciallo:
452 Closure stk:
453 0 2 4 6 12 17 13 5 10
454 s stk:
455 end # identifier a == identifier T5 - - identifier h * * ( ( identifier i
456 tokens:
457 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
458 Ciallo~
459 action: shift, push state 11 and symbol +
460
461 Ciallo:
462 Closure stk:
463 0 2 4 6 12 17 13 5 10 11
464 s stk:
465 end # identifier a == identifier T5 - - identifier h * * ( ( identifier i + +
466 tokens:
467 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
468 Ciallo~
469 action: shift, push state 9 and symbol j
470
471 Ciallo:
472 Closure stk:
473 0 2 4 6 12 17 13 5 10 11 9
474 s stk:
475 end # identifier a == identifier T5 - - identifier h * * ( ( identifier i + + identifier j
476 tokens:
477 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
478 Ciallo~
479 action: reduce byu production F -> i
480
481 Ciallo:
482 Closure stk:
483 0 2 4 6 12 17 13 5 10 11 7
484 s stk:
485 end # identifier a == identifier T5 - - identifier h * * ( ( identifier i + + identifier j
486 tokens:
487 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
488 Ciallo~
489 action: reduce byu production T -> F
490
491 Ciallo:
492 Closure stk:
493 0 2 4 6 12 17 13 5 10 11 16
494 s stk:
495 end # identifier a == identifier T5 - - identifier h * * ( ( identifier i + + identifier j
496 tokens:
497 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
498 Ciallo~
```

```

499 action: shift, push state 14 and symbol /
500
501 Ciallo:
502 Closure stk:
503 0 2 4 6 12 17 13 5 10 11 16 14
504 s stk:
505 end # identifier a = = identifier T5 - - identifier h * * ( ( identifier i + + identifier j / /
506 tokens:
507 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
508 Ciallo~
509 action: shift, push state 9 and symbol k
510
511 Ciallo:
512 Closure stk:
513 0 2 4 6 12 17 13 5 10 11 16 14 9
514 s stk:
515 end # identifier a = = identifier T5 - - identifier h * * ( ( identifier i + + identifier j / / identifi
516 tokens:
517 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
518 Ciallo~
519 action: reduce byu production F -> i
520
521 Ciallo:
522 Closure stk:
523 0 2 4 6 12 17 13 5 10 11 16 14 19
524 s stk:
525 end # identifier a = = identifier T5 - - identifier h * * ( ( identifier i + + identifier j / / identifi
526 tokens:
527 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
528 Ciallo~
529 action: reduce byu production T -> T / F
530
531 Ciallo:
532 Closure stk:
533 0 2 4 6 12 17 13 5 10 11 16
534 s stk:
535 end # identifier a = = identifier T5 - - identifier h * * ( ( identifier i + + identifier T6
536 tokens:
537 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
538 Ciallo~
539 action: reduce byu production E -> E + T
540
541 Ciallo:
542 Closure stk:
543 0 2 4 6 12 17 13 5 10
544 s stk:
545 end # identifier a = = identifier T5 - - identifier h * * ( ( identifier T7
546 tokens:
547 a = b + c * ( d - e / f ) + g - h * ( i + j / k #
548 Ciallo~
549 Zako♡~ unexpected token: # at 24
550 WA

```

o std1:

```

1 Ciallo:
2 Closure stk:
3 0
4 s stk:
5 end #
6 tokens:
7 a = b * ( c + d ) / e - f / g #

```

```
8  Ciallo~
9  action: shift, push state 3 and symbol a
10
11 Ciallo:
12 Closure stk:
13 0 3
14 s stk:
15 end # identifier a
16 tokens:
17 a = b * ( c + d ) / e - f / g #
18 Ciallo~
19 action: reduce byu production V -> i
20
21 Ciallo:
22 Closure stk:
23 0 2
24 s stk:
25 end # identifier a
26 tokens:
27 a = b * ( c + d ) / e - f / g #
28 Ciallo~
29 action: shift, push state 4 and symbol =
30
31 Ciallo:
32 Closure stk:
33 0 2 4
34 s stk:
35 end # identifier a = =
36 tokens:
37 a = b * ( c + d ) / e - f / g #
38 Ciallo~
39 action: shift, push state 9 and symbol b
40
41 Ciallo:
42 Closure stk:
43 0 2 4 9
44 s stk:
45 end # identifier a = = identifier b
46 tokens:
47 a = b * ( c + d ) / e - f / g #
48 Ciallo~
49 action: reduce byu production F -> i
50
51 Ciallo:
52 Closure stk:
53 0 2 4 7
54 s stk:
55 end # identifier a = = identifier b
56 tokens:
57 a = b * ( c + d ) / e - f / g #
58 Ciallo~
59 action: reduce byu production T -> F
60
61 Ciallo:
62 Closure stk:
63 0 2 4 8
64 s stk:
65 end # identifier a = = identifier b
66 tokens:
67 a = b * ( c + d ) / e - f / g #
68 Ciallo~
69 action: shift, push state 13 and symbol *
```

```
70
71 Ciallo:
72 Closure stk:
73 0 2 4 8 13
74 s stk:
75 end # identifier a = = identifier b * *
76 tokens:
77 a = b * ( c + d ) / e - f / g #
78 Ciallo~
79 action: shift, push state 5 and symbol (
80
81 Ciallo:
82 Closure stk:
83 0 2 4 8 13 5
84 s stk:
85 end # identifier a = = identifier b * * ( (
86 tokens:
87 a = b * ( c + d ) / e - f / g #
88 Ciallo~
89 action: shift, push state 9 and symbol c
90
91 Ciallo:
92 Closure stk:
93 0 2 4 8 13 5 9
94 s stk:
95 end # identifier a = = identifier b * * ( ( identifier c
96 tokens:
97 a = b * ( c + d ) / e - f / g #
98 Ciallo~
99 action: reduce byu production F -> i
100
101 Ciallo:
102 Closure stk:
103 0 2 4 8 13 5 7
104 s stk:
105 end # identifier a = = identifier b * * ( ( identifier c
106 tokens:
107 a = b * ( c + d ) / e - f / g #
108 Ciallo~
109 action: reduce byu production T -> F
110
111 Ciallo:
112 Closure stk:
113 0 2 4 8 13 5 8
114 s stk:
115 end # identifier a = = identifier b * * ( ( identifier c
116 tokens:
117 a = b * ( c + d ) / e - f / g #
118 Ciallo~
119 action: reduce byu production E -> T
120
121 Ciallo:
122 Closure stk:
123 0 2 4 8 13 5 10
124 s stk:
125 end # identifier a = = identifier b * * ( ( identifier c
126 tokens:
127 a = b * ( c + d ) / e - f / g #
128 Ciallo~
129 action: shift, push state 11 and symbol +
130
131 Ciallo:
```

```
132 Closure stk:
133 0 2 4 8 13 5 10 11
134 s stk:
135 end # identifier a = = identifier b * * ( ( identifier c + +
136 tokens:
137 a = b * ( c + d ) / e - f / g #
138 Ciallo~
139 action: shift, push state 9 and symbol d
140
141 Ciallo:
142 Closure stk:
143 0 2 4 8 13 5 10 11 9
144 s stk:
145 end # identifier a = = identifier b * * ( ( identifier c + + identifier d
146 tokens:
147 a = b * ( c + d ) / e - f / g #
148 Ciallo~
149 action: reduce byu production F -> i
150
151 Ciallo:
152 Closure stk:
153 0 2 4 8 13 5 10 11 7
154 s stk:
155 end # identifier a = = identifier b * * ( ( identifier c + + identifier d
156 tokens:
157 a = b * ( c + d ) / e - f / g #
158 Ciallo~
159 action: reduce byu production T -> F
160
161 Ciallo:
162 Closure stk:
163 0 2 4 8 13 5 10 11 16
164 s stk:
165 end # identifier a = = identifier b * * ( ( identifier c + + identifier d
166 tokens:
167 a = b * ( c + d ) / e - f / g #
168 Ciallo~
169 action: reduce byu production E -> E + T
170
171 Ciallo:
172 Closure stk:
173 0 2 4 8 13 5 10
174 s stk:
175 end # identifier a = = identifier b * * ( ( identifier T1
176 tokens:
177 a = b * ( c + d ) / e - f / g #
178 Ciallo~
179 action: shift, push state 15 and symbol )
180
181 Ciallo:
182 Closure stk:
183 0 2 4 8 13 5 10 15
184 s stk:
185 end # identifier a = = identifier b * * ( ( identifier T1 ) )
186 tokens:
187 a = b * ( c + d ) / e - f / g #
188 Ciallo~
189 action: reduce byu production F -> ( E )
190
191 Ciallo:
192 Closure stk:
193 0 2 4 8 13 18
```



```
194 s stk:
195 end # identifier a = = identifier b * * identifier T1
196 tokens:
197 a = b * ( c + d ) / e - f / g #
198 Ciallo~
199 action: reduce byu production T -> T * F
200
201 Ciallo:
202 Closure stk:
203 0 2 4 8
204 s stk:
205 end # identifier a = = identifier T2
206 tokens:
207 a = b * ( c + d ) / e - f / g #
208 Ciallo~
209 action: shift, push state 14 and symbol /
210
211 Ciallo:
212 Closure stk:
213 0 2 4 8 14
214 s stk:
215 end # identifier a = = identifier T2 / /
216 tokens:
217 a = b * ( c + d ) / e - f / g #
218 Ciallo~
219 action: shift, push state 9 and symbol e
220
221 Ciallo:
222 Closure stk:
223 0 2 4 8 14 9
224 s stk:
225 end # identifier a = = identifier T2 / / identifier e
226 tokens:
227 a = b * ( c + d ) / e - f / g #
228 Ciallo~
229 action: reduce byu production F -> i
230
231 Ciallo:
232 Closure stk:
233 0 2 4 8 14 19
234 s stk:
235 end # identifier a = = identifier T2 / / identifier e
236 tokens:
237 a = b * ( c + d ) / e - f / g #
238 Ciallo~
239 action: reduce byu production T -> T / F
240
241 Ciallo:
242 Closure stk:
243 0 2 4 8
244 s stk:
245 end # identifier a = = identifier T3
246 tokens:
247 a = b * ( c + d ) / e - f / g #
248 Ciallo~
249 action: reduce byu production E -> T
250
251 Ciallo:
252 Closure stk:
253 0 2 4 6
254 s stk:
255 end # identifier a = = identifier T3
```

```
256 tokens:
257 a = b * ( c + d ) / e - f / g #
258 Ciallo~
259 action: shift, push state 12 and symbol -
260
261 Ciallo:
262 Closure stk:
263 0 2 4 6 12
264 s stk:
265 end # identifier a = = identifier T3 - -
266 tokens:
267 a = b * ( c + d ) / e - f / g #
268 Ciallo~
269 action: shift, push state 9 and symbol f
270
271 Ciallo:
272 Closure stk:
273 0 2 4 6 12 9
274 s stk:
275 end # identifier a = = identifier T3 - - identifier f
276 tokens:
277 a = b * ( c + d ) / e - f / g #
278 Ciallo~
279 action: reduce byu production F -> i
280
281 Ciallo:
282 Closure stk:
283 0 2 4 6 12 7
284 s stk:
285 end # identifier a = = identifier T3 - - identifier f
286 tokens:
287 a = b * ( c + d ) / e - f / g #
288 Ciallo~
289 action: reduce byu production T -> F
290
291 Ciallo:
292 Closure stk:
293 0 2 4 6 12 17
294 s stk:
295 end # identifier a = = identifier T3 - - identifier f
296 tokens:
297 a = b * ( c + d ) / e - f / g #
298 Ciallo~
299 action: shift, push state 14 and symbol /
300
301 Ciallo:
302 Closure stk:
303 0 2 4 6 12 17 14
304 s stk:
305 end # identifier a = = identifier T3 - - identifier f / /
306 tokens:
307 a = b * ( c + d ) / e - f / g #
308 Ciallo~
309 action: shift, push state 9 and symbol g
310
311 Ciallo:
312 Closure stk:
313 0 2 4 6 12 17 14 9
314 s stk:
315 end # identifier a = = identifier T3 - - identifier f / / identifier g
316 tokens:
317 a = b * ( c + d ) / e - f / g #
```

```

318 Ciallo~
319 action: reduce byu production F -> i
320
321 Ciallo:
322 Closure stk:
323 0 2 4 6 12 17 14 19
324 s stk:
325 end # identifier a = = identifier T3 - - identifier f / / identifier g
326 tokens:
327 a = b * ( c + d ) / e - f / g #
328 Ciallo~
329 action: reduce byu production T -> T / F
330
331 Ciallo:
332 Closure stk:
333 0 2 4 6 12 17
334 s stk:
335 end # identifier a = = identifier T3 - - identifier T4
336 tokens:
337 a = b * ( c + d ) / e - f / g #
338 Ciallo~
339 action: reduce byu production E -> E - T
340
341 Ciallo:
342 Closure stk:
343 0 2 4 6
344 s stk:
345 end # identifier a = = identifier T5
346 tokens:
347 a = b * ( c + d ) / e - f / g #
348 Ciallo~
349 action: reduce byu production S -> V = E
350
351 Ciallo:
352 Closure stk:
353 0 1
354 s stk:
355 end # identifier a
356 tokens:
357 a = b * ( c + d ) / e - f / g #
358 Ciallo~
359 action: accept
360 success
361 AC
362 Ciallo: ( + + | identifier c | identifier d | identifier T1 ): T1 := c + d
363 Ciallo: ( * * | identifier b | identifier T1 | identifier T2 ): T2 := b * T1
364 Ciallo: ( / / | identifier T2 | identifier e | identifier T3 ): T3 := T2 / e
365 Ciallo: ( / / | identifier f | identifier g | identifier T4 ): T4 := f / g
366 Ciallo: ( - - | identifier T3 | identifier T4 | identifier T5 ): T5 := T3 - T4
367 Ciallo: ( = = | identifier T5 | ε | identifier a ): a = T5

```

o std2:

```

1 Ciallo:
2 Closure stk:
3 0
4 s stk:
5 end #
6 tokens:
7 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
8 Ciallo~
9 action: shift, push state 3 and symbol a

```

```
10
11 Ciallo:
12 Closure stk:
13 0 3
14 s stk:
15 end # identifier a
16 tokens:
17 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
18 Ciallo~
19 action: reduce byu production V -> i
20
21 Ciallo:
22 Closure stk:
23 0 2
24 s stk:
25 end # identifier a
26 tokens:
27 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
28 Ciallo~
29 action: shift, push state 4 and symbol =
30
31 Ciallo:
32 Closure stk:
33 0 2 4
34 s stk:
35 end # identifier a = =
36 tokens:
37 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
38 Ciallo~
39 action: shift, push state 9 and symbol b
40
41 Ciallo:
42 Closure stk:
43 0 2 4 9
44 s stk:
45 end # identifier a = = identifier b
46 tokens:
47 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
48 Ciallo~
49 action: reduce byu production F -> i
50
51 Ciallo:
52 Closure stk:
53 0 2 4 7
54 s stk:
55 end # identifier a = = identifier b
56 tokens:
57 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
58 Ciallo~
59 action: reduce byu production T -> F
60
61 Ciallo:
62 Closure stk:
63 0 2 4 8
64 s stk:
65 end # identifier a = = identifier b
66 tokens:
67 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
68 Ciallo~
69 action: reduce byu production E -> T
70
71 Ciallo:
```

```
72 Closure stk:
73 0 2 4 6
74 s stk:
75 end # identifier a = = identifier b
76 tokens:
77 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
78 Ciallo~
79 action: shift, push state 11 and symbol +
80
81 Ciallo:
82 Closure stk:
83 0 2 4 6 11
84 s stk:
85 end # identifier a = = identifier b + +
86 tokens:
87 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
88 Ciallo~
89 action: shift, push state 9 and symbol c
90
91 Ciallo:
92 Closure stk:
93 0 2 4 6 11 9
94 s stk:
95 end # identifier a = = identifier b + + identifier c
96 tokens:
97 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
98 Ciallo~
99 action: reduce byu production F -> i
100
101 Ciallo:
102 Closure stk:
103 0 2 4 6 11 7
104 s stk:
105 end # identifier a = = identifier b + + identifier c
106 tokens:
107 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
108 Ciallo~
109 action: reduce byu production T -> F
110
111 Ciallo:
112 Closure stk:
113 0 2 4 6 11 16
114 s stk:
115 end # identifier a = = identifier b + + identifier c
116 tokens:
117 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
118 Ciallo~
119 action: shift, push state 13 and symbol *
120
121 Ciallo:
122 Closure stk:
123 0 2 4 6 11 16 13
124 s stk:
125 end # identifier a = = identifier b + + identifier c * *
126 tokens:
127 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
128 Ciallo~
129 action: shift, push state 5 and symbol (
130
131 Ciallo:
132 Closure stk:
133 0 2 4 6 11 16 13 5
```

```
134 s stk:
135 end # identifier a = = identifier b + + identifier c * * ( (
136 tokens:
137 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
138 Ciallo~
139 action: shift, push state 9 and symbol d
140
141 Ciallo:
142 Closure stk:
143 0 2 4 6 11 16 13 5 9
144 s stk:
145 end # identifier a = = identifier b + + identifier c * * ( ( identifier d
146 tokens:
147 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
148 Ciallo~
149 action: reduce byu production F -> i
150
151 Ciallo:
152 Closure stk:
153 0 2 4 6 11 16 13 5 7
154 s stk:
155 end # identifier a = = identifier b + + identifier c * * ( ( identifier d
156 tokens:
157 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
158 Ciallo~
159 action: reduce byu production T -> F
160
161 Ciallo:
162 Closure stk:
163 0 2 4 6 11 16 13 5 8
164 s stk:
165 end # identifier a = = identifier b + + identifier c * * ( ( identifier d
166 tokens:
167 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
168 Ciallo~
169 action: reduce byu production E -> T
170
171 Ciallo:
172 Closure stk:
173 0 2 4 6 11 16 13 5 10
174 s stk:
175 end # identifier a = = identifier b + + identifier c * * ( ( identifier d
176 tokens:
177 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
178 Ciallo~
179 action: shift, push state 12 and symbol -
180
181 Ciallo:
182 Closure stk:
183 0 2 4 6 11 16 13 5 10 12
184 s stk:
185 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - -
186 tokens:
187 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
188 Ciallo~
189 action: shift, push state 9 and symbol e
190
191 Ciallo:
192 Closure stk:
193 0 2 4 6 11 16 13 5 10 12 9
194 s stk:
195 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier e
```

```

196 tokens:
197 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
198 Ciallo~
199 action: reduce byu production F -> i
200
201 Ciallo:
202 Closure stk:
203 0 2 4 6 11 16 13 5 10 12 7
204 s stk:
205 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier e
206 tokens:
207 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
208 Ciallo~
209 action: reduce byu production T -> F
210
211 Ciallo:
212 Closure stk:
213 0 2 4 6 11 16 13 5 10 12 17
214 s stk:
215 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier e
216 tokens:
217 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
218 Ciallo~
219 action: shift, push state 14 and symbol /
220
221 Ciallo:
222 Closure stk:
223 0 2 4 6 11 16 13 5 10 12 17 14
224 s stk:
225 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier e / /
226 tokens:
227 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
228 Ciallo~
229 action: shift, push state 9 and symbol f
230
231 Ciallo:
232 Closure stk:
233 0 2 4 6 11 16 13 5 10 12 17 14 9
234 s stk:
235 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier e / / identifier
236 tokens:
237 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
238 Ciallo~
239 action: reduce byu production F -> i
240
241 Ciallo:
242 Closure stk:
243 0 2 4 6 11 16 13 5 10 12 17 14 19
244 s stk:
245 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier e / / identifier
246 tokens:
247 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
248 Ciallo~
249 action: reduce byu production T -> T / F
250
251 Ciallo:
252 Closure stk:
253 0 2 4 6 11 16 13 5 10 12 17
254 s stk:
255 end # identifier a = = identifier b + + identifier c * * ( ( identifier d - - identifier T1
256 tokens:
257 a = b + c * ( d - e / f ) + g - h * i + j / k ) #

```

```
258 Ciallo~
259 action: reduce byu production E -> E - T
260
261 Ciallo:
262 Closure stk:
263 0 2 4 6 11 16 13 5 10
264 s stk:
265 end # identifier a = = identifier b + + identifier c * * ( ( identifier T2
266 tokens:
267 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
268 Ciallo~
269 action: shift, push state 15 and symbol )
270
271 Ciallo:
272 Closure stk:
273 0 2 4 6 11 16 13 5 10 15
274 s stk:
275 end # identifier a = = identifier b + + identifier c * * ( ( identifier T2 ) )
276 tokens:
277 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
278 Ciallo~
279 action: reduce byu production F -> ( E )
280
281 Ciallo:
282 Closure stk:
283 0 2 4 6 11 16 13 18
284 s stk:
285 end # identifier a = = identifier b + + identifier c * * identifier T2
286 tokens:
287 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
288 Ciallo~
289 action: reduce byu production T -> T * F
290
291 Ciallo:
292 Closure stk:
293 0 2 4 6 11 16
294 s stk:
295 end # identifier a = = identifier b + + identifier T3
296 tokens:
297 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
298 Ciallo~
299 action: reduce byu production E -> E + T
300
301 Ciallo:
302 Closure stk:
303 0 2 4 6
304 s stk:
305 end # identifier a = = identifier T4
306 tokens:
307 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
308 Ciallo~
309 action: shift, push state 11 and symbol +
310
311 Ciallo:
312 Closure stk:
313 0 2 4 6 11
314 s stk:
315 end # identifier a = = identifier T4 + +
316 tokens:
317 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
318 Ciallo~
319 action: shift, push state 9 and symbol g
```



```
320
321 Ciallo:
322 Closure stk:
323 0 2 4 6 11 9
324 s stk:
325 end # identifier a = = identifier T4 + + identifier g
326 tokens:
327 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
328 Ciallo~
329 action: reduce byu production F -> i
330
331 Ciallo:
332 Closure stk:
333 0 2 4 6 11 7
334 s stk:
335 end # identifier a = = identifier T4 + + identifier g
336 tokens:
337 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
338 Ciallo~
339 action: reduce byu production T -> F
340
341 Ciallo:
342 Closure stk:
343 0 2 4 6 11 16
344 s stk:
345 end # identifier a = = identifier T4 + + identifier g
346 tokens:
347 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
348 Ciallo~
349 action: reduce byu production E -> E + T
350
351 Ciallo:
352 Closure stk:
353 0 2 4 6
354 s stk:
355 end # identifier a = = identifier T5
356 tokens:
357 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
358 Ciallo~
359 action: shift, push state 12 and symbol -
360
361 Ciallo:
362 Closure stk:
363 0 2 4 6 12
364 s stk:
365 end # identifier a = = identifier T5 - -
366 tokens:
367 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
368 Ciallo~
369 action: shift, push state 9 and symbol h
370
371 Ciallo:
372 Closure stk:
373 0 2 4 6 12 9
374 s stk:
375 end # identifier a = = identifier T5 - - identifier h
376 tokens:
377 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
378 Ciallo~
379 action: reduce byu production F -> i
380
381 Ciallo:
```

```
382 Closure stk:
383 0 2 4 6 12 7
384 s stk:
385 end # identifier a = = identifier T5 - - identifier h
386 tokens:
387 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
388 Ciallo~
389 action: reduce byu production T -> F
390
391 Ciallo:
392 Closure stk:
393 0 2 4 6 12 17
394 s stk:
395 end # identifier a = = identifier T5 - - identifier h
396 tokens:
397 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
398 Ciallo~
399 action: shift, push state 13 and symbol *
400
401 Ciallo:
402 Closure stk:
403 0 2 4 6 12 17 13
404 s stk:
405 end # identifier a = = identifier T5 - - identifier h * *
406 tokens:
407 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
408 Ciallo~
409 action: shift, push state 9 and symbol i
410
411 Ciallo:
412 Closure stk:
413 0 2 4 6 12 17 13 9
414 s stk:
415 end # identifier a = = identifier T5 - - identifier h * * identifier i
416 tokens:
417 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
418 Ciallo~
419 action: reduce byu production F -> i
420
421 Ciallo:
422 Closure stk:
423 0 2 4 6 12 17 13 18
424 s stk:
425 end # identifier a = = identifier T5 - - identifier h * * identifier i
426 tokens:
427 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
428 Ciallo~
429 action: reduce byu production T -> T * F
430
431 Ciallo:
432 Closure stk:
433 0 2 4 6 12 17
434 s stk:
435 end # identifier a = = identifier T5 - - identifier T6
436 tokens:
437 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
438 Ciallo~
439 action: reduce byu production E -> E - T
440
441 Ciallo:
442 Closure stk:
443 0 2 4 6
```

```
444 s stk:
445 end # identifier a = = identifier T7
446 tokens:
447 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
448 Ciallo~
449 action: shift, push state 11 and symbol +
450
451 Ciallo:
452 Closure stk:
453 0 2 4 6 11
454 s stk:
455 end # identifier a = = identifier T7 + +
456 tokens:
457 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
458 Ciallo~
459 action: shift, push state 9 and symbol j
460
461 Ciallo:
462 Closure stk:
463 0 2 4 6 11 9
464 s stk:
465 end # identifier a = = identifier T7 + + identifier j
466 tokens:
467 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
468 Ciallo~
469 action: reduce byu production F -> i
470
471 Ciallo:
472 Closure stk:
473 0 2 4 6 11 7
474 s stk:
475 end # identifier a = = identifier T7 + + identifier j
476 tokens:
477 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
478 Ciallo~
479 action: reduce byu production T -> F
480
481 Ciallo:
482 Closure stk:
483 0 2 4 6 11 16
484 s stk:
485 end # identifier a = = identifier T7 + + identifier j
486 tokens:
487 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
488 Ciallo~
489 action: shift, push state 14 and symbol /
490
491 Ciallo:
492 Closure stk:
493 0 2 4 6 11 16 14
494 s stk:
495 end # identifier a = = identifier T7 + + identifier j / /
496 tokens:
497 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
498 Ciallo~
499 action: shift, push state 9 and symbol k
500
501 Ciallo:
502 Closure stk:
503 0 2 4 6 11 16 14 9
504 s stk:
505 end # identifier a = = identifier T7 + + identifier j / / identifier k
```

```

506 tokens:
507 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
508 Ciallo~
509 action: reduce byu production F -> i
510
511 Ciallo:
512 Closure stk:
513 0 2 4 6 11 16 14 19
514 s stk:
515 end # identifier a = = identifier T7 + + identifier j / / identifier k
516 tokens:
517 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
518 Ciallo~
519 action: reduce byu production T -> T / F
520
521 Ciallo:
522 Closure stk:
523 0 2 4 6 11 16
524 s stk:
525 end # identifier a = = identifier T7 + + identifier T8
526 tokens:
527 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
528 Ciallo~
529 action: reduce byu production E -> E + T
530
531 Ciallo:
532 Closure stk:
533 0 2 4 6
534 s stk:
535 end # identifier a = = identifier T9
536 tokens:
537 a = b + c * ( d - e / f ) + g - h * i + j / k ) #
538 Ciallo~
539 Zako~ unexpected token: ) at 23
540 WA

```

o std3:

```

1 Ciallo:
2 Closure stk:
3 0
4 s stk:
5 end #
6 tokens:
7 a = b + c / #
8 Ciallo~
9 action: shift, push state 3 and symbol a
10
11 Ciallo:
12 Closure stk:
13 0 3
14 s stk:
15 end # identifier a
16 tokens:
17 a = b + c / #
18 Ciallo~
19 action: reduce byu production V -> i
20
21 Ciallo:
22 Closure stk:
23 0 2
24 s stk:

```

```
25 end # identifier a
26 tokens:
27 a = b + c / #
28 Ciallo~
29 action: shift, push state 4 and symbol =
30
31 Ciallo:
32 Closure stk:
33 0 2 4
34 s stk:
35 end # identifier a = =
36 tokens:
37 a = b + c / #
38 Ciallo~
39 action: shift, push state 9 and symbol b
40
41 Ciallo:
42 Closure stk:
43 0 2 4 9
44 s stk:
45 end # identifier a = = identifier b
46 tokens:
47 a = b + c / #
48 Ciallo~
49 action: reduce byu production F -> i
50
51 Ciallo:
52 Closure stk:
53 0 2 4 7
54 s stk:
55 end # identifier a = = identifier b
56 tokens:
57 a = b + c / #
58 Ciallo~
59 action: reduce byu production T -> F
60
61 Ciallo:
62 Closure stk:
63 0 2 4 8
64 s stk:
65 end # identifier a = = identifier b
66 tokens:
67 a = b + c / #
68 Ciallo~
69 action: reduce byu production E -> T
70
71 Ciallo:
72 Closure stk:
73 0 2 4 6
74 s stk:
75 end # identifier a = = identifier b
76 tokens:
77 a = b + c / #
78 Ciallo~
79 action: shift, push state 11 and symbol +
80
81 Ciallo:
82 Closure stk:
83 0 2 4 6 11
84 s stk:
85 end # identifier a = = identifier b + +
86 tokens:
```

```
87  a = b + c / #
88  Ciallo~
89  action: shift, push state 9 and symbol c
90
91  Ciallo:
92  Closure stk:
93  0 2 4 6 11 9
94  s stk:
95  end # identifier a = = identifier b + + identifier c
96  tokens:
97  a = b + c / #
98  Ciallo~
99  action: reduce byu production F -> i
100
101 Ciallo:
102 Closure stk:
103 0 2 4 6 11 7
104 s stk:
105 end # identifier a = = identifier b + + identifier c
106 tokens:
107 a = b + c / #
108 Ciallo~
109 action: reduce byu production T -> F
110
111 Ciallo:
112 Closure stk:
113 0 2 4 6 11 16
114 s stk:
115 end # identifier a = = identifier b + + identifier c
116 tokens:
117 a = b + c / #
118 Ciallo~
119 action: shift, push state 14 and symbol /
120
121 Ciallo:
122 Closure stk:
123 0 2 4 6 11 16 14
124 s stk:
125 end # identifier a = = identifier b + + identifier c / /
126 tokens:
127 a = b + c / #
128 Ciallo~
129 Zako♡~ unexpected token: # at 6
130 WA
```

o std(st)

```
1  action st:
2  closure: 0:
3  identifier -> shift 3
4  closure: 1:
5  end -> acc
6  closure: 2:
7  = -> shift 4
8  closure: 3:
9  = -> reduce i
10 closure: 4:
11 identifier -> shift 9
12 ( -> shift 5
13 closure: 5:
14 identifier -> shift 9
15 ( -> shift 5
```

```
16  closure: 6:
17  + -> shift 11
18  - -> shift 12
19  end -> reduce V = E
20  closure: 7:
21  ) -> reduce F
22  + -> reduce F
23  - -> reduce F
24  * -> reduce F
25  / -> reduce F
26  end -> reduce F
27  closure: 8:
28  ) -> reduce T
29  + -> reduce T
30  - -> reduce T
31  * -> shift 13
32  / -> shift 14
33  end -> reduce T
34  closure: 9:
35  ) -> reduce i
36  + -> reduce i
37  - -> reduce i
38  * -> reduce i
39  / -> reduce i
40  end -> reduce i
41  closure: 10:
42  ) -> shift 15
43  + -> shift 11
44  - -> shift 12
45  closure: 11:
46  identifier -> shift 9
47  ( -> shift 5
48  closure: 12:
49  identifier -> shift 9
50  ( -> shift 5
51  closure: 13:
52  identifier -> shift 9
53  ( -> shift 5
54  closure: 14:
55  identifier -> shift 9
56  ( -> shift 5
57  closure: 15:
58  ) -> reduce ( E )
59  + -> reduce ( E )
60  - -> reduce ( E )
61  * -> reduce ( E )
62  / -> reduce ( E )
63  end -> reduce ( E )
64  closure: 16:
65  ) -> reduce E + T
66  + -> reduce E + T
67  - -> reduce E + T
68  * -> shift 13
69  / -> shift 14
70  end -> reduce E + T
71  closure: 17:
72  ) -> reduce E - T
73  + -> reduce E - T
74  - -> reduce E - T
75  * -> shift 13
76  / -> shift 14
77  end -> reduce E - T
```

```
78  closure: 18:
79  ) -> reduce T * F
80  + -> reduce T * F
81  - -> reduce T * F
82  * -> reduce T * F
83  / -> reduce T * F
84  end -> reduce T * F
85  closure: 19:
86  ) -> reduce T / F
87  + -> reduce T / F
88  - -> reduce T / F
89  * -> reduce T / F
90  / -> reduce T / F
91  end -> reduce T / F
92
93  goto st:
94  closure: 0:
95  S -> 1
96  V -> 2
97  closure: 4:
98  E -> 6
99  F -> 7
100 T -> 8
101 closure: 5:
102 E -> 10
103 F -> 7
104 T -> 8
105 closure: 11:
106 F -> 7
107 T -> 16
108 closure: 12:
109 F -> 7
110 T -> 17
111 closure: 13:
112 F -> 18
113 closure: 14:
114 F -> 19
```

• 测试结果

- 结果正确 要看程序输出的话可以把sh脚本中删除输出文件的语句注释

```
1  bash go_st.sh
2  accept
3
4  bash go_SLR.sh
5  test: 0
6  accept
7  test: 1
8  accept
9  test: 2
10 accept
11 test: 3
12 accept
```

八. 心得体会

大模拟写写写 zzz

debug过程非常显著提高了抗压能力, 调了两个小时发现初始化的时候少了一行初始化grammar, 写破防了, 写算法题都没这么破防过, jump了

附录

1. MeloN_H.hpp

```
1  #pragma once
2  #include <algorithm>
3  #include <array>
4  #include <bitset>
5  #include <cassert>
6  #include <cctype>
7  #include <chrono>
8  #include <cmath>
9  #include <cstring>
10 #include <ctime>
11 #include <fstream>
12 #include <functional>
13 #include <iomanip>
14 #include <iostream>
15 #include <limits>
16 #include <map>
17 #include <queue>
18 #include <random>
19 #include <ranges>
20 #include <set>
21 #include <stack>
22 #include <string>
23 #include <tuple>
24 #include <unordered_map>
25 #include <unordered_set>
26
27 using std::array, std::bitset, std::deque, std::greater, std::less, std::map,
28     std::multiset, std::pair, std::priority_queue, std::set, std::stack,
29     std::string, std::vector, std::tuple, std::function;
30
31 using NAME = void;      using uint = unsigned;   using ll = long long;      using ull = unsigned long lon
32 using ld = long double; using i128 = __int128_t; using u128 = __uint128_t; using f128 = __float128;
33
34 #define meion    auto
35 #define iroha    return
36
```

2. MeloN_debug.hpp

```
1  #pragma once
2
3  template <class T, size_t size = std::tuple_size<T>::value>
4  std::string to_debug(T, std::string s = "")
5      requires(not std::ranges::range<T>);
6  std::string to_debug(meion x)
7      requires requires(std::ostream& os) { os << x; }
8  {
9      iroha static_cast<std::ostringstream>(std::ostringstream() << x).str();
10 }
11 std::string to_debug(std::ranges::range meion x, std::string s = "")
12     requires(not std::is_same_v<decltype(x), std::string>)
13 {
14     for (meion xi : x) {
15         s += ", " + to_debug(xi);
16     }
17     iroha "[" + s.substr(s.empty() ? 0 : 2) + "]";
18 }
19 template <class T, size_t size>
20 std::string to_debug(T x, std::string s)
```

```

21     requires(not std::ranges::range<T>)
22 {
23     [&<size_t... I>(std::index_sequence<I...>) {
24         ((s += ", " + to_debug(std::get<I>(x))), ...);
25     }(std::make_index_sequence<size>());
26     iroha "(" + s.substr(s.empty() ? 0 : 2) + ")";
27 }
28 #ifdef MeIoN
29 #define debug(...) std::cout << "Ciallo~(∠ · w< ) ^ ★ " << "(" #__VA_ARGS__ ") = " << to_debug(std::tuple<
30 #else
31 #define debug(...) void(0721)
32 #endif

```

3.4 SLR_solver.hpp

```

1  #include "0_token_solver.hpp"
2  #include "1_grammar_solver.hpp"
3
4  namespace SLR {
5      using grammar_solver = n_grammar_solver::lycoris;
6      using n_grammar_solver::grammar;
7      using n_grammar_solver::production;
8      using n_grammar_solver::token_solver;
9      struct item {
10         production prod;
11         size_t pla;
12         item(production prod, size_t pla) : prod(prod), pla(pla) {};
13         item(const item &x) : prod(x.prod), pla(x.pla) {}
14         bool operator==(const item &rhs) const {
15             iroha prod == rhs.prod and pla == rhs.pla;
16         }
17         bool operator<(const item &rhs) const {
18             iroha prod < rhs.prod or (prod == rhs.prod and pla < rhs.pla);
19         }
20         meion take() const -> const string & {
21             iroha prod[pla];
22         }
23         meion get_L() const -> const string & {
24             iroha prod.get_L();
25         }
26     };
27
28     template <typename T>
29     struct symple_array_with_ctrc {
30         array<T, 4> val;
31         symple_array_with_ctrc() { val.fill(T{}); }
32         symple_array_with_ctrc(T a = T{}, T b = T{}, T c = T{}, T d = T{})
33             : val {a, b, c, d} {}
34         T operator[](const size_t &k) const {
35             assert(k < 4);
36             iroha T(val[k]);
37         }
38         meion set(const size_t &k, const takina &x) -> void {
39             assert(k < 4);
40             val[k] = x;
41         }
42     };
43     template <class T>
44     struct MeIoN_Queue {
45         vector<T> q;
46         int pos = 0;
47         void reserve(int n) { q.reserve(n); }

```

```

48     int size() const { iroha int(q.size()) - pos; }
49     bool empty() const { iroha pos == int(q.size()); }
50     T& front() { iroha q[pos]; }
51     T& back() { iroha q.back(); }
52     template <typename... Args>
53     void emplace_back(Args&&... args) {
54         q.emplace_back(std::forward<Args>(args)...);
55     }
56     void push_back(const T& v) { q.push_back(v); }
57     void pop() { ++pos; }
58     void pop_back() { q.pop_back(); }
59     void clear() {
60         q.clear();
61         pos = 0;
62     }
63 };
64
65 template <typename T>
66 using queue = MeIoN_Queue<T>;
67 using closure = set<item>;
68 using quaternion = symple_array_with_ctrc<takina>;
69 using std::variant;
70
71 std::ostream& operator<<(std::ostream& os, const quaternion& quad) {
72     os << "Ciallo: ( " << quad[0] << " | " << quad[1] << " | " << quad[2]
73     << " | " << quad[3] << " )";
74     if (quad[2].type == Epsilon) {
75         os << quad[3].value << " " << quad[0].value << " " << quad[1].value;
76     } else {
77         os << quad[3].value << " := " << quad[1].value << " "
78         << quad[0].value << " " << quad[2].value;
79     }
80     iroha os;
81 }
82
83 enum action_type { acc, shift, reduce };
84
85 struct action {
86     action_type type;
87     variant<int, production> value;
88 };
89
90 class lycoris {
91 public:
92     lycoris() {}
93     lycoris(const grammar &g) : G(g) {
94         t_sol.build("");
95         g_sol.set_grammar(G);
96         g_sol.compute_first();
97         g_sol.compute_follow("S");
98         build();
99     }
100     meion check(const string &s) -> bool {
101         qs.clear();
102         vector tokens = t_sol.get_tokens(s);
103         vector<int> closure_stk{0};
104         vector<takina> s_stk{{End, "#"}};
105         meion get_quaternion = [&](const production& prod) -> void {
106             if (prod.size() == 1) {
107                 iroha ;
108             } else if (prod.get_L() == "F") {
109                 takina tok = s_stk.end()[-2];

```

```

110         s_stk.pop_back();
111         s_stk.pop_back();
112         s_stk.back() = tok;
113     } else {
114         takina y = s_stk.end()[-1];
115         s_stk.pop_back();
116         takina op = s_stk.back();
117         s_stk.pop_back();
118         takina x = s_stk.back();
119         s_stk.pop_back();
120         if (op.type == Assign) {
121             qs.emplace_back(op, y, takina{Epsilon, ""}, x);
122             s_stk.emplace_back(x);
123         } else {
124             s_stk.emplace_back(new_quaternion(x, op, y));
125         }
126     }
127 };
128 int pla = 0;
129 meion mygo = [&]() -> void {
130     std::cout << "\nCiallo:\n";
131     std::cout << "Closure stk: " << '\n';
132     for (meion x : closure_stk) {
133         std::cout << x << ' ';
134     }
135     std::cout << "\ns stk: " << '\n';
136     for (meion x : s_stk) {
137         std::cout << x << " ";
138     }
139     std::cout << "\ntokens: " << '\n';
140     for (meion [x, y] : tokens) {
141         std::cout << y << " ";
142     }
143     std::cout << "\nCiallo~" << std::endl;
144 };
145
146 while (true) {
147     int state = closure_stk.back();
148     takina token = tokens[pla];
149     token_type type = token.type;
150     mygo();
151     if (not a_st.contains(state) or not a_st[state].contains(type)) {
152         iroha std::cout
153             << "Zako~ unexpected token: " << token.value << " at "
154             << pla << std::endl,
155             false;
156     }
157     const action &act = a_st[state][type];
158     if (act.type == shift) {
159         int nxt = std::get<int>(act.value);
160         closure_stk.emplace_back(nxt);
161         s_stk.emplace_back(token);
162         ++pla;
163         std::cout << "action: shift, push state " << nxt
164             << " and symbol " << token.value << std::endl;
165     } else if (act.type == reduce) {
166         production prod = std::get<production>(act.value);
167         std::cout << "action: reduce byu production " << prod
168             << std::endl;
169         for (int i = 0; i < prod.size(); ++i) {
170             closure_stk.pop_back();
171         }

```

```

172         int goto_state = g_st[closure_stk.back()][H(prod.get_L())];
173         closure_stk.emplace_back(goto_state);
174         get_quaternion(prod);
175     } else if (act.type == acc) {
176         iroha std::cout << "action: accept" << '\n'
177             << "success" << std::endl,
178             true;
179     } else {
180         iroha std::cerr << "Zako♡~ inv action type" << std::endl, false;
181     }
182 }
183 iroha std::cerr << "unexpected Fail" << std::endl, false;
184 }
185 meion show_ast() -> void {
186     std::cout << "action st:\n";
187     vector<ull> v;
188     a_st.view([&](const meion &x, const meion &y) {
189         v.emplace_back(x);
190     });
191     std::ranges::sort(v);
192     for (meion &x : v) {
193         vector<ull> v;
194         a_st[x].view([&](const meion &a, const meion &b) -> void {
195             v.emplace_back(a);
196         });
197         std::ranges::sort(v);
198
199         std::cout << "closure: " << x << ":\n";
200         for (const meion &type : v) {
201             std::cout << type_to_s[type] << " -> ";
202             action A = a_st[x][type];
203             if (A.type == acc) {
204                 std::cout << "acc ";
205             } else if (A.type == shift) {
206                 std::cout << "shift " << std::get<int>(A.value);
207             } else if (A.type == reduce) {
208                 std::cout << "reduce ";
209                 for (const meion &s : std::get<production>(A.value).Rs()) {
210                     std::cout << s << ' ';
211                 }
212             }
213             std::cout << '\n';
214         }
215     }
216     std::cout.flush();
217 }
218 meion show_gst() -> void {
219     vector<ull> v;
220     g_st.view([&](const meion a, const meion b) {
221         v.emplace_back(a);
222     });
223     std::ranges::sort(v);
224
225     std::cout << "goto st:\n";
226     for (const meion x : v) {
227         vector<ull> v;
228         g_st[x].view([&](const meion &a, const meion &b) -> void {
229             v.emplace_back(a);
230         });
231         std::ranges::sort(v);
232
233         std::cout << "closure: " << x << ":\n";

```

```

234         for (const meion y : v) {
235             std::cout << ' ' << HINA[y] << " -> " << g_st[x][y] << '\n';
236         }
237     }
238     std::cout.flush();
239 }
240 meion get_quadruples() const -> const vector<quaternion> & {
241     iroha qs;
242 }
243 private:
244     grammar G;
245     hash_map<hash_map<action>> a_st;
246     hash_map<hash_map<ull>> g_st;
247     token_solver t_sol;
248     grammar_solver g_sol;
249     vector<quaternion> qs;
250     meion dbg(closure x) -> void {
251         std::cout << "closure: " << x.size() << "\n{ ";
252         for (meion a : x) {
253             std::cout << a.get_L() << ' ';
254         }
255         std::cout << "}" << std::endl;
256     }
257     meion dbg(item x) -> void {
258         std::cout << "item: { " << x.get_L() << ' ' << x.pla;
259         for (meion &a : x.prod.Rs()) {
260             std::cout << ' ' << a;
261         }
262         std::cout << "}" << std::endl;
263     }
264     meion dbg(string s) -> void {
265         std::cout << "str: " << s << std::endl;
266     }
267     meion get_closure(const closure &state) -> closure {
268         queue<item> q;
269         // dbg(state);
270         for (const meion &item : state) {
271             q.emplace_back(item);
272         }
273         closure closure_set = state;
274         while (not q.empty()) {
275             const meion item = q.front();
276             q.pop();
277             // std::cout << "pop: ";
278             // dbg(item);
279             if (item.pla + 1 > item.prod.size()) {
280                 continue;
281             }
282             const string &s = item.take();
283             // dbg(s);
284             if (g_sol.is_non_terminal(s)) {
285                 for (const meion &production : G[H(s)]) {
286                     SLR::item my_new = {production, 0};
287                     // std::cout << "inque ";
288                     // dbg(my_new);
289                     if (closure_set.emplace(my_new).second) {
290                         q.emplace_back(my_new);
291                     }
292                 }
293             }
294         }
295         iroha closure_set;

```

```
296     }
297     meion get_closure_TH(const closure &state, const string &s) -> closure {
298         closure goto_set;
299         for (const meion &item : state) {
300             if (item.pla < item.prod.size() and item.take() == s) {
301                 SLR::item my_new = item;
302                 ++my_new.pla;
303                 goto_set.emplace(my_new);
304             }
305         }
306         iroha get_closure(goto_set);
307     }
308     meion build() -> void {
309         item st_item = {"S'", {"S"}}, 0};
310         // debug(st_item.prod, st_item.pla);
311         closure st_closure = get_closure({st_item});
312         // dbg(st_closure);
313         // exit(0);
314         int tot = 0;
315         map<closure, int> closure_mp;
316         closure_mp[st_closure] = tot++;
317         queue<closure> q;
318         q.emplace_back(st_closure);
319
320         while (not q.empty()) {
321             const meion state = q.front();
322             q.pop();
323             set<string> se;
324             for (const meion it : state) {
325                 if (it.pla < it.prod.size()) {
326                     se.emplace(it.take());
327                 }
328             }
329             for (const meion &s : se) {
330                 closure new_closure = get_closure_TH(state, s);
331                 if (not new_closure.empty() and
332                     not closure_mp.contains(new_closure)) {
333                     closure_mp[new_closure] = tot++;
334                     q.emplace_back(new_closure);
335                 }
336             }
337         }
338
339         for (const meion &[state, i] : closure_mp) {
340             for (const meion &item : state) {
341                 if (item.pla == item.prod.size()) {
342                     if (item.get_L() == "S'") {
343                         a_st[i][End] = {acc, {}}; // ac
344                     } else {
345                         for (const meion &follow :
346                             g_sol.get_follow(item.get_L())) {
347                             a_st[i][follow] = {reduce, item.prod};
348                         }
349                     }
350                 } else {
351                     const string &s = item.take();
352                     int goto_state = closure_mp[get_closure_TH(state, s)];
353                     if (g_sol.is_non_terminal(s)) {
354                         g_st[i][H(s)] = goto_state;
355                     } else {
356                         token_type type = t_sol.get_token_type(s);
357                         a_st[i][type] = {shift, goto_state};
```

```
358         }
359     }
360 }
361 }
362 }
363 meion new_quaternion(takina x, takina op, takina y) -> takina {
364     string my_new = "T" + std::to_string(qs.size() + 1);
365     takina tok = {Identifier, my_new};
366     iroha qs.emplace_back(op, x, y, tok), tok;
367 }
368 };
369
370 meion test_set() -> grammar {
371     grammar G;
372     G[H("S'")] = {{ "S'", { "S" } }};
373     G[H("S")] = {{ "S", { "V", "=", "E" } }};
374     G[H("E")] = {
375         { "E", { "E", "+", "T" } }, { "E", { "E", "-", "T" } }, { "E", { "T" } }};
376     G[H("T")] = {
377         { "T", { "T", "*", "F" } }, { "T", { "T", "/", "F" } }, { "T", { "F" } }};
378     G[H("F")] = {{ "F", { "(", "E", ")" } }, { "F", { "i" } }};
379     G[H("V")] = {{ "V", { "i" } }};
380     iroha G;
381 }
382 meion test_ast() -> void {
383     lycoris chisato(test_set());
384     chisato.show_ast();
385 }
386 meion test_gst() -> void {
387     lycoris chisato(test_set());
388     chisato.show_gst();
389 }
390 meion test_SLR() -> void {
391     lycoris chisato(test_set());
392     string s, t;
393     while (std::getline(std::cin, s)) {
394         t += s + '\n';
395     }
396     if (not t.empty()) {
397         t.pop_back();
398     }
399     if (chisato.check(t)) {
400         std::cout << "AC" << std::endl;
401         for (meion x : chisato.get_quadruples()) {
402             std::cout << x << '\n';
403         }
404         std::cout.flush();
405     } else {
406         std::cout << "WA" << std::endl;
407     }
408 }
409 } // namespace SLR
```