

考试科目名称 编译原理; A 卷

考试方式: 开卷 日期: 2023年06月24日 教师: 魏恒峰、冯洋

院系 (专业): 软件学院 (软件工程) 年级: 2021级

姓名: _____ 学号: _____ 成绩: _____

| 题号 | 一 | 二 | 三 | 四 | 五 | 六 | 总分 |
|----|---|---|---|---|---|---|----|
| 分数 | | | | | | | |

注意事项:

- 诚信考试, 不得作弊。
- 若对题意有疑问, 请及时提出。
- 题目不是按照难度排列的, 请注意合理分配时间。
- 为了避免连锁错误造成的影响, 请尽量给出关键步骤。
- 字迹要尽可能工整, 解题过程要尽可能清晰。

前五道题目均与 C 语言中的声明语法相关。
文法如下, 记为 \mathcal{G}_D , 其中 D 表示 Declaration。

$$D' \rightarrow D \quad (0)$$

$$D \rightarrow D () \quad (1)$$

$$D \rightarrow D [] \quad (2)$$

$$D \rightarrow * D \quad (3)$$

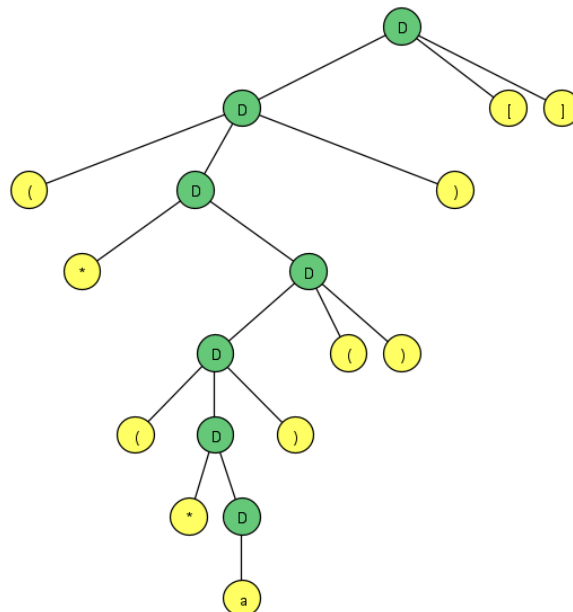
$$D \rightarrow (D) \quad (4)$$

$$D \rightarrow \text{id} \quad (5)$$

说明:

- 第 (0) 条是为第 3 题准备的增广产生式。
- 第 (1) 条中的 $()$ 表示函数调用。
- 第 (2) 条中的 $[]$ 表示数组。
- 第 (3) 条中的 $*$ 表示指针。
- 第 (4) 条中的 $()$ 表示括号, 主要用于改变优先级。
- 第 (5) 条中的 **id** 表示标识符, 定义从略, 题目中会对此作具体说明。
- $()$ 与 $[]$ 的优先级相同, 且比 $*$ 的优先级高。

例子: 对于输入 $(*(*a) ()) []$, 相应的语法分析树如下图所示。请对照输入与语法分析树, 仔细理解优先级问题。



题目 1 (正则表达式与自动机 (15 = 7 + 8 分))

考虑文法 \mathcal{G}_D 中的第 (1)、(2)、(5) 三条, 其中 **id** 限定为词法单元 a (记该文法为 \mathcal{R}_D):

$$D \rightarrow D () \quad (1)$$

$$D \rightarrow D [] \quad (2)$$

$$D \rightarrow a \quad (5)$$

\mathcal{R}_D 等价的正则表达式为 $a(() \mid [])^*$ 。

(注意, $(() \mid [])$ 中的最外层结构 $(\dots \mid \dots)$ 表示正则表达式中的“或者”含义。内层的 $()$ 与 $[]$ 是文法 \mathcal{G}_D 的一部分。为了简略, 此处没有对它们进行转义处理。)

(1) 请使用 Thompson 构造法构造等价的 NFA。

(2) 请使用子集构造法构造等价的 DFA。 (注: 请标明 NFA 与 DFA 状态之间的对应关系)

题目 2 (LL 语法分析 (15 = 6 + 6 + 3 分))

考虑文法 \mathcal{G}_D (不包括第 (0) 条):

$$D \rightarrow D () \quad (1)$$

$$D \rightarrow D [] \quad (2)$$

$$D \rightarrow * D \quad (3)$$

$$D \rightarrow (D) \quad (4)$$

$$D \rightarrow \mathbf{id} \quad (5)$$

(1) 请计算文法符号 D 的 FIRST 集合与 FOLLOW 集合。

(2) 请为文法 \mathcal{G}_D 构造预测分析表。

(3) 请问文法 \mathcal{G}_D 是否是 $LL(1)$ 文法, 并说明理由。

题目 3 (*LR* 语法分析 (20 = 6 + 3 + 6 + 5 分))

考虑文法 \mathcal{G}_D (已作增广处理, 即包括第 (0) 条):

$$D' \rightarrow D \quad (0)$$

$$D \rightarrow D () \quad (1)$$

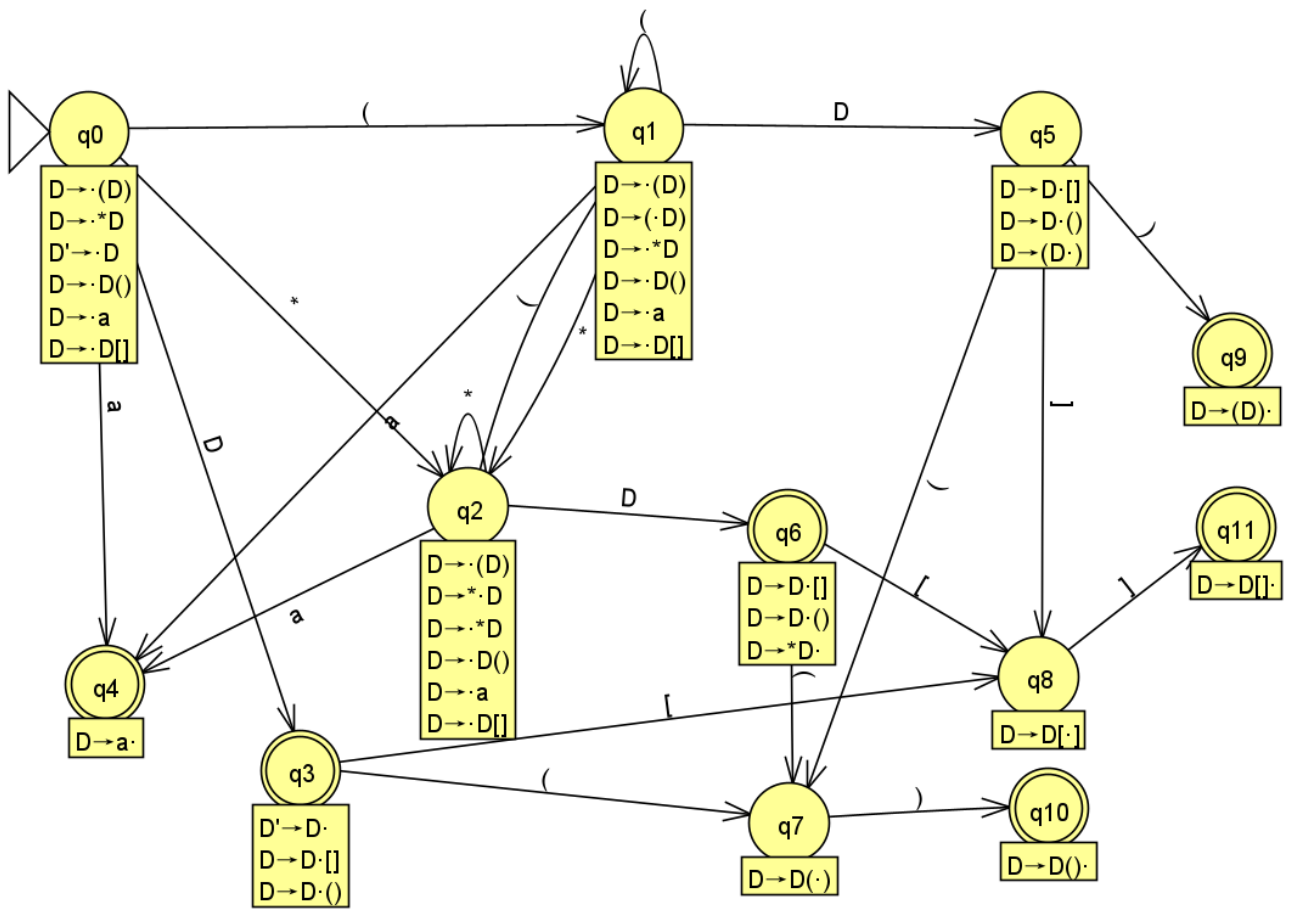
$$D \rightarrow D [] \quad (2)$$

$$D \rightarrow * D \quad (3)$$

$$D \rightarrow (D) \quad (4)$$

$$D \rightarrow \mathbf{id} \quad (5)$$

- (1) 请根据第 8 页所示的 *LR*(0) 自动机为 \mathcal{G}_D 构造 *SLR*(1) 表。(待填的 *SLR*(1) 表格见第 9 页)
- (2) 已知 \mathcal{G}_D 不是 *SLR*(1) 文法, 请论述理由。
- (3) 请说明, 如何消解 *SLR*(1) 中的冲突, 使得消解后的 *SLR*(1) 表可以正确解析 \mathcal{G}_D 文法。请简要说明理由。
- (4) 请在消解后的 *SLR*(1) 的基础上给出识别输入串 $(*(\mathbf{a}))() []$ 时自动机所经历的状态 (编号), 其中 \mathbf{a} 为标识符。



| | ACTION | | | | | | | GOTO |
|----|--------|---|---|---|---|----|----|----------|
| | (|) | * | [|] | id | \$ | <i>D</i> |
| 0 | | | | | | | | |
| 1 | | | | | | | | |
| 2 | | | | | | | | |
| 3 | | | | | | | | |
| 4 | | | | | | | | |
| 5 | | | | | | | | |
| 6 | | | | | | | | |
| 7 | | | | | | | | |
| 8 | | | | | | | | |
| 9 | | | | | | | | |
| 10 | | | | | | | | |
| 11 | | | | | | | | |

题目 4 (ANTLR 4 与“优先级上升算法” (20 = 10 + 10 分))

文法 \mathcal{G}_D 的 ANTLR 4 格式如下¹。

$$D' \rightarrow D$$

$$D \rightarrow D ((\mid [)) \mid * D \mid (D) \mid \mathbf{id}$$

注意, $((\mid [))$ 中的外层结构 $(\dots \mid \dots)$ 表示 ANTLR 中的“或者”子规则。也就是说, 外层的 $(\mid [)$ 是 .g4 语法的一部分。另外, 内层的 $(\mid [)$ 是文法 \mathcal{G}_D 的一部分。为了简略, 此处没有把它们放在引号内。

- (1) 请给出 ANTLR 4 使用优先级上升算法改造后的文法。
- (2) 请给出 $((*(\mathbf{a}))\square)$ 在改造后的文法下对应的语法分析树。请给出关键的解释, 点到即可, 不必面面俱到。 注意: 与试卷第 2 页的语法分析树并不相同。

¹仍沿用 \mathcal{G}_D 中的文法符号, 包括大小写。

题目 5 (语法制导定义与翻译 (10 分))

考虑文法 \mathcal{G}_D (不包括第 (0) 条):

$$D \rightarrow D () \quad (1)$$

$$D \rightarrow D [] \quad (2)$$

$$D \rightarrow * D \quad (3)$$

$$D \rightarrow (D) \quad (4)$$

$$D \rightarrow \text{id} \quad (5)$$

请设计语法制导的**翻译方案**, 为 **id** 的类型提供英文/中文说明文档 (说明: 由于中英文语法结构的不同, 输出英文说明文档可能更容易)。

你需要自行定义合适的属性, 请**指明**哪些是综合属性, 哪些是继承属性。

举例:

- $(**a()) []$: declare **a** as pointer to function returning pointer to array
- $(**a()) []$: declare **a** as function returning pointer to pointer to array
- $(**a[])(()) []$: declare **a** as array of pointer to function returning pointer to array

为了便于理解, 后面给出了后两个句子对应的语法分析树 (第一个句子对应的语法分析树见试卷第 2 页)。

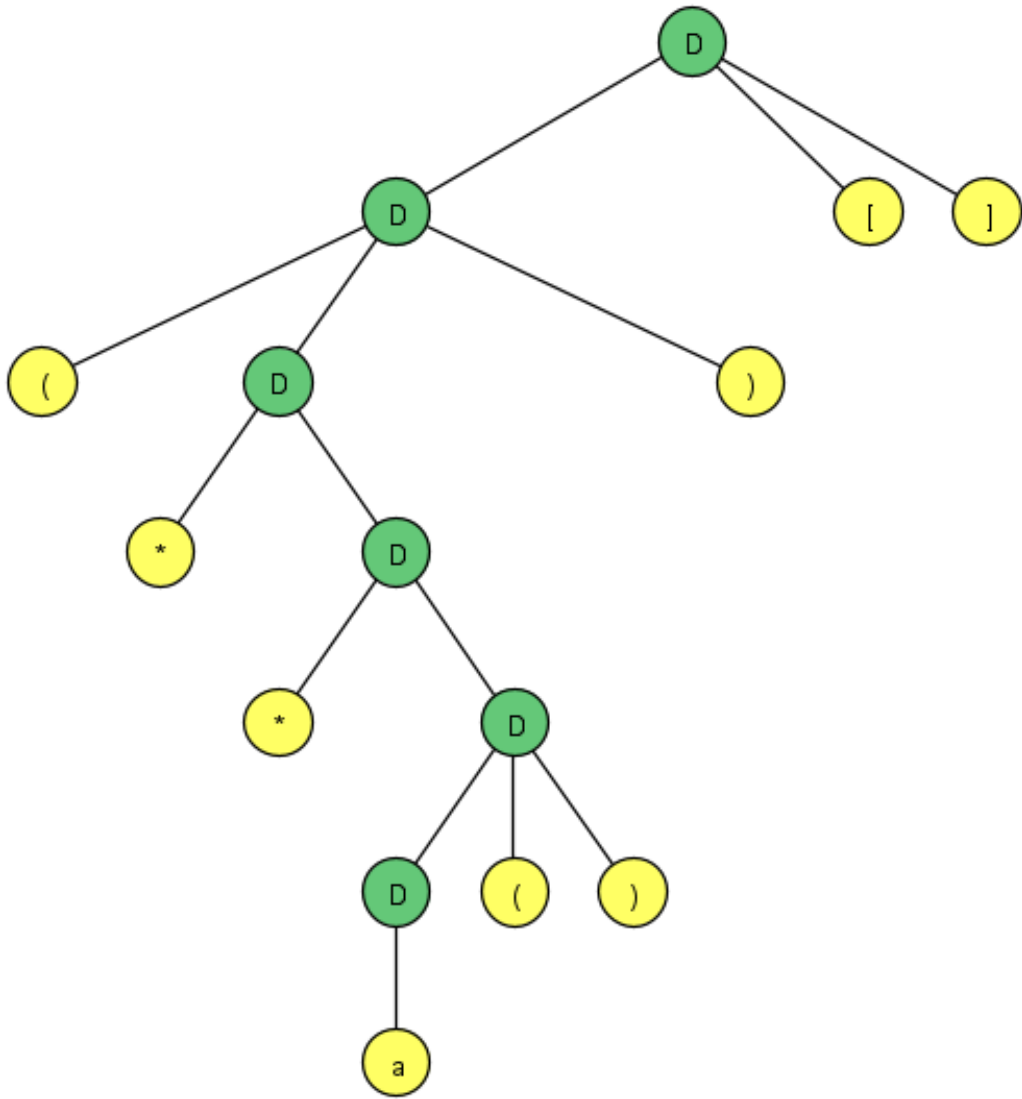


图 1: `(**a())[]`: declare a as function returning pointer to pointer to array

题目 6 (目标代码生成 ($20 = 2 + 1 + 1 + 2 + 2 + 0 + 2 + 1 + 2 + 2 + 1 + 2 + 2 + 0$ 分))

考虑如下计算第 n 项 Fibonacci 数的 C 语言程序 fib-rec.c

```

5      #include <stdio.h>
6
7      int fib(int n) {
8          // fib(0) = 0, fib(1) = 1
9          if (n <= 1) {
10             return n;
11         }
12
13         return fib(n: n - 1) + fib(n: n - 2);
14     }
15
16     int n = 20;
17
18     int main(void) {
19         int result = fib(n);
20
21         printf(Format: "fib(%d) = %d\n", n, result);
22
23         return 0;
24     }








```

下面两页给出了 fib-rec.c 对应的 RISC-V 代码片段, 请填充缺失的代码行。

说明:

- 第 17、23、49 行代码是相同的, 仅计一次分数。
- 第 42 行处, 假设 fib-rec.c 中的 n 占 4 字节, 也就是一个 word 大小。
- 所有填充处仅允许使用如下指令 (包括伪指令; 汇编伪指令不限): li、add、addi、sub、ble、bge、lw、sw、jal、jalr、ecall、j、la。

```

1  .text
2  .global main
3
4  ##### fib #####
5  fib:
6  # base case:  $n \leq 1$ 
7  li t0, 1
8  
9
10 addi sp, sp, -16    # allocate stack
11     # store a0 on stack
12     # store ra on stack
13
14 #  $n > 1$ :  $\text{fib}(n - 1) + \text{fib}(n - 2)$ 
15 lw a0, 12(sp)      # a0: n
16 addi a0, a0, -1    # a0: n - 1
17     # call fib on (n - 1)
18 mv t1, a0          # t1: fib(n - 1)
19 
20
21 lw a0, 12(sp)      # a0: n
22 addi a0, a0, -2    # a0: n - 2
23     # call fib on (n)
24 mv t2, a0          # t2: fib(n - 2)
25 

```



```

26
27 lw t1, 4(sp)
28 lw t2, 0(sp)
29 add a0, t1, t2      # a0: fib(n - 1) + fib(n - 2)
30
31       # restore ra
32       # clear stack
33
34 j end
35
36 base_case:
37
38 end:
39  # ret
40 ##### main #####
41 .data
42 n: 
43
44 # n = 10: 0 1 1 2 3 5 8 13 21 34 55
45 .text
46 main:
47  # a0: address of n
48  # a0: value of n
49  # call fib

```

