编译原理作业2

姓名：孙铎

学号：200110503

第5章

课本第5章第15题；

15. 设有如下文法 G：

$<S>\rightarrow<A>$　　$<A>\rightarrow<B><A>\mid\varepsilon$　　$<B>\rightarrow a<B>\mid b$

（1）试用识别活前缀的方式给出文法 G 的 LR(1)项目集。

（2）构造 G 的 LR(1)分析表。

（3）给出输入符号串 $w=abab$ 的自底向上语法分析过程。

（1）：

开始符号`S`只出现在产生式`S -> A`的左部，符合**只有一个接收状态**的要求，所以不使用增广文法。

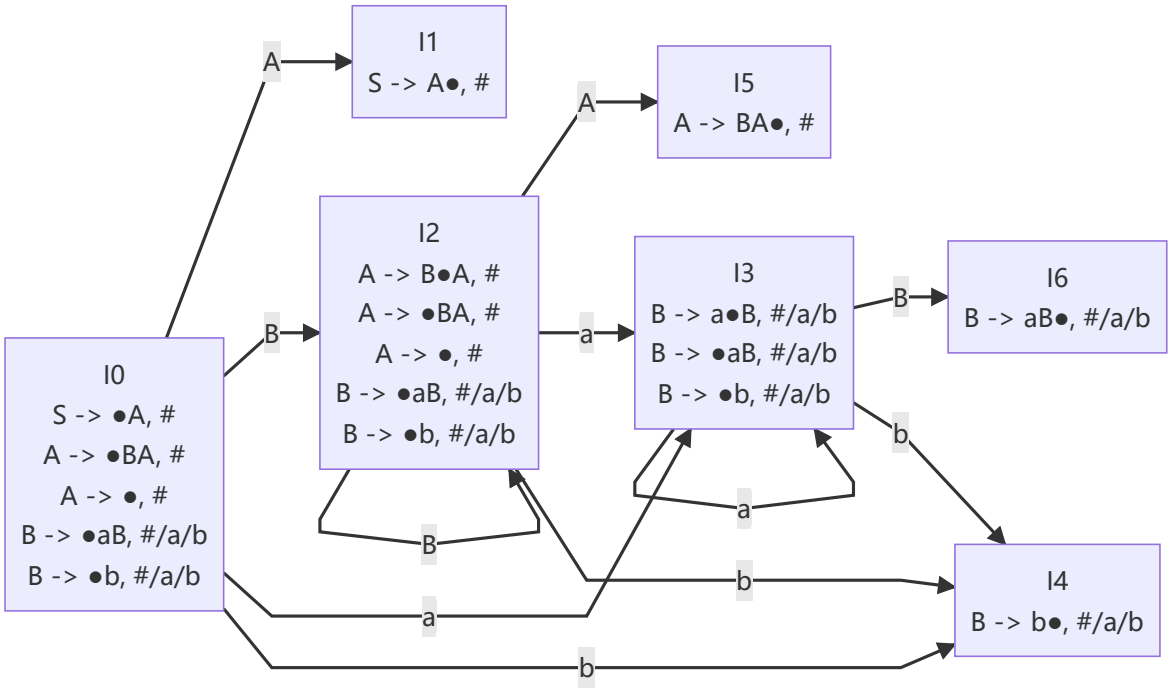
产生式编号为：

1. `S -> A`
2. `A -> BA`
3. `A -> ϵ`
4. `B -> aB`
5. `B -> b`

需要的FIRST集：

- `FIRST(A)={ϵ,a,b}`

文法G的LR(1)自动机/项目集为：



（2）：

构造G的LR(1)分析表：

STATUS	ACTION			GOTO	
	a	b	#	A	B
0	s3	s4	r3	1	2
1			acc		
2	s3	s4	r3	5	2
3	s3	s4			6
4	r5	r5	r5		
5			r2		
6	r4	r4	r4		

（3）：

输入符号串 `w=abab` 的自底向上语法分析过程为：

状态栈(栈底→栈顶)	符号栈(栈底→栈顶)	输入缓冲(自左向右读取)	ACTION	GOTO
0	#	abab#	s3	
03	#a	bab#	s4	
034	#ab	ab#	r5	6
036	#aB	ab#	r4	2
02	#B	ab#	s3	
023	#Ba	b#	s4	
0234	#Bab	#	r5	6
0236	#BaB	#	r4	2
022	#BB	#	r3	5
0225	#BBA	#	r2	5
025	#BA	#	r2	1
01	#A	#	acc	

第7章

课本第7章第22题第（1）小题；

22. 试将下面的语句翻译成四元式序列。

(1) **while** a < c ∧ b < d **do**

if a = 1 **then** c := c + 1

else while a < = d **do**

a := a + 2;

语句的三地址码为：

1. `if a < c goto 3`

2. `goto 15`
3. `if b < d goto 5`
4. `goto 15`
5. `if a = 1 goto 7`
6. `goto 10`
7. `t1 := c + 1`
8. `c := t1`
9. `goto 1`
10. `if a <= d goto 12`
11. `goto 1`
12. `t2 := a + 2`
13. `a := t2`
14. `goto 10`
15.

对应的四元式序列为：

1. `(j<, a, c, 3)`
2. `(j, -, -, 15)`
3. `(j<, b, d, 5)`
4. `(j, -, -, 15)`
5. `(j=, a, 1, 7)`
6. `(j, -, -, 10)`
7. `(=, c, 1, t1)`
8. `(=, t1, -, c)`
9. `(j, -, -, 1)`
10. `(j<=, a, d, 12)`
11. `(j, -, -, 1)`
12. `(=, a, 2, t2)`
13. `(=, t2, -, a)`
14. `(j, -, -, 10)`
15.

第9章

课本第9章第8题；

8. 设有如下的 C 语言程序：

```
typedef struct_a{
    short  i;   short  j;   short  k;
}a;
typedef struct_b{
    long  i;   short  k;
}b;
main()
{printf("Size of short, long, a and b = %d,%d,%d,%d\n",
sizeof(short),sizeof(long),sizeof(a),sizeof(b));}
```

该程序在 x86/Linux 机器上的运行结果如下：

```
Size of short,long,a and b = 2,4,6,8
```

已知 short 类型和 long 类型分别对齐到 2 的倍数和 4 的倍数。试问，为什么类型 b 的长度会等于 8？

由地址空间分配策略可知：

因为`short`类型要对齐到2的倍数，且`sizeof(short)`为2，所以：

- `a.i`占用的相对地址为`0~1`
- `a.j`占用的相对地址为`2~3`
- `a.k`占用的相对地址为`4~5`

又因为`long`类型要对齐到4的倍数，且`sizeof(long)`为4，所以：

- `b.i`占用的相对地址为`8~11`
- `b.k`占用的相对地址为`12~13`

所以`b`整体占用的相对地址应当是`6~13`，所以类型b的长度等于8

第11章

课本第11章第2题；

2. 试确定下列指令序列的开销。

(1) MOV y,R_0
MOV z,R_1
ADD R_1,R_0
MOV R_0,x

(2) MOV i,R_0
MUL 8,R_0
MOV a(R_0),R_1
MOV R_1,b

(3) MOV p,R_0
MOV 0(R_0),R_1
MOV R_1,x

(4) MOV x,R_0
MOV y,R_1
SUB R_1,R_0
MOV R_0,*R_3

(1) **总开销为7**

- `MOV y, R0`：开销为2

  `MOV`指令本身开销为1，再加上存储变量`y`地址的开销1，寄存器不需要额外开销
- `MOV z, R1`：开销为2

  `MOV`指令本身开销为1，再加上存储变量`z`地址的开销1，寄存器不需要额外开销
- `ADD R1, R0`：开销为1

  `ADD`指令本身开销为1，寄存器不需要额外开销
- `MOV R0, x`：开销为2

  `MOV`指令本身开销为1，再加上存储变量`x`地址的开销1，寄存器不需要额外开销

(2) **总开销为8**

- `MOV i, R0`：开销为2

`MOV`指令本身开销为1，再加上存储变量`i`地址的开销1，寄存器不需要额外开销

- `MUL 8, R0`：开销为2

  `MOV`指令本身开销为1，再加上存储立即数8的开销1，寄存器不需要额外开销

- `MOV a(R0), R1`：开销为2

  `MOV`指令本身开销为1，再加上存储变量`a`地址的开销1，寄存器不需要额外开销

- `MOV R1, b`：开销为2

  `MOV`指令本身开销为1，再加上存储变量`b`地址的开销1，寄存器不需要额外开销

(3) **总开销为6**

- `MOV p, R0`：开销为2

  `MOV`指令本身开销为1，再加上存储变量`p`地址的开销1，寄存器不需要额外开销

- `MOV 0(R0), R1`：开销为2

  `MOV`指令本身开销为1，再加上存储立即数0的开销1，寄存器不需要额外开销

- `MOV R1, x`：开销为2

  `MOV`指令本身开销为1，再加上存储变量`x`地址的开销1，寄存器不需要额外开销

(4) **总开销为6**

- `MOV x, R0`：开销为2

  `MOV`指令本身开销为1，再加上存储变量`x`地址的开销1，寄存器不需要额外开销

- `MOV y, R1`：开销为2

  `MOV`指令本身开销为1，再加上存储变量`y`地址的开销1，寄存器不需要额外开销

- `SUB R1, R0`：开销为1

  `SUB`指令本身开销为1，寄存器不需要额外开销

- `MOV R0, *R3`：开销为1

  `MOV`指令本身开销为1，寄存器不需要额外开销