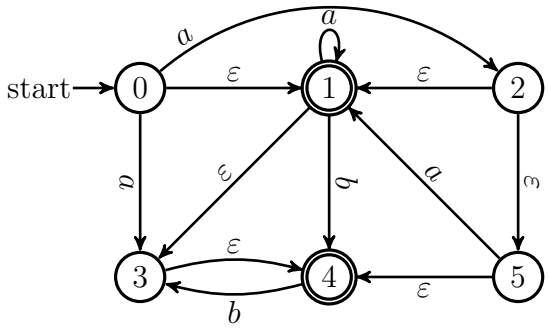
武汉大学计算机学院
2013-2014学年第一学期2011级
《编译原理》期末考试试卷（A）

学号：_____ 姓名：_____ 专业：_____ 成绩：_____

(注：①考试时间为120分钟；②所有的解答必须写在答题纸上，并注明题号。)

一、 设NFA N的状态转换图如下所示： (25分, 每小题5分)



(1) 试写出NFA N接受字符串"$aaabb$"的过程；

(2) 设用子集构造法求出的与NFA N等价的DFA M有4个状态A，B，C和D，其中$A = \varepsilon\text{-closure}(\{0\})$，$\text{Dtrans}(A, a) = B$ ，$\text{Dtrans}(A, b) = C$ 试求与状态A，B，C和D所对应的NFA N的状态集，并画出DFA M的状态转换图；

(3) 求DFA M的最小状态自动机；

(4) 试用自然语言描述NFA N所接受的语言；

(5) 求正规表达式r，使得$L(r) = L(N)$.

二、 设C语言数组初始化文法$G(I)$定义如下： (25分, 每小题5分)

$$\begin{aligned} I &\rightarrow \texttt{\{}L\texttt{\}} \mid n \\ L &\rightarrow L\texttt{,}\ L \mid I \end{aligned}$$

其中: '`{`', '`}`', '`,`'和'n'为终结符，'I'和'L'是非终结符，I是文法开始符号.

(1) 试写出语句"`{{`n`},`n`}`"的一个最左推导；

(2) 试消除文法$G(I)$中的左递归；

(3) 试对消除左递归后的文法所有非终结符求First集和Follow集；

(4) 试对消除左递归后的文法构造LL(1)分析表，从而说明$G(I)$不是LL(1)文法；

(5) 试利用你的分析表写出语句"`{{`n`},`n`}`"的一个正确的分析过程.

三、 设文法$G(I)$如题二所示： (10分, 5+5)

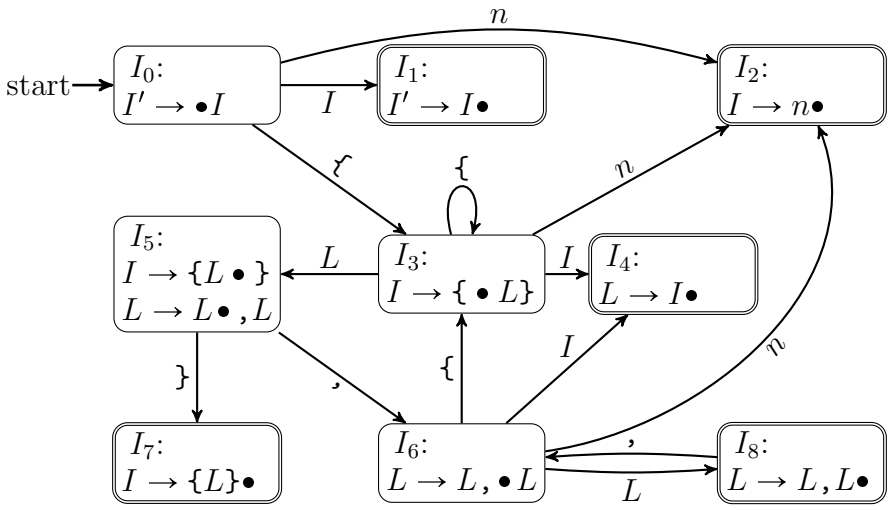
(1) 试对语句“{n,n,n}””画出两颗不同的语法树从而说明该文法为二义文法；

(2) 试设计一个与文法$G(I)$等价的无二义的文法，使得‘,’运算为**左**结合运算.

四、 设题二文法$G(I)$的拓广文法$G(I')$如下所示： (20分, 5+5+5+5)

$$
\begin{aligned}
I' &\rightarrow I && (0)\\
I &\rightarrow \{L\} && (1)\\
&\mid n && (2)\\
L &\rightarrow L,\ L && (3)\\
&\mid I && (4)
\end{aligned}
$$

文法$G(I')$的识别活前缀LR(0)项目自动机如下图所示(**注意每个状态仅列出了核心项目**)：



(1) 试求状态I_3所对应的LR(0)项目集；

(2) 试求所有的仅由**终结符**组成的活前缀所对应的正则表达式；

(3) 试构造该文法的SLR分析表，并使得运算的结合次序与题三所规定的一致；

(4) 试利用你的分析表写出语句“{n,n,n}”的分析过程.

五、 C语言数组初始化是由常数与嵌套的花括号组成的列表，其文法如题二所示. 例如：

```
int a[2][3] = {0,1,2,3,4,5};
int b[2][3] = {0,1};           /* 等价于 {0,1,0,0,0,0} */
int c[2][3] = {{0,1,2},{3,4,5}}; /* 等价于{0,1,2,3,4,5} */
int d[2][3] = {{0,1},{2}};     /* 等价于 {0,1,0,2,0,0} */
int e[2][3] = {0,1,2,{3}};     /* 等价于 {0,1,2,3,0,0} */
int f[2][3] = {{1},2,3};       /* 等价于 {1,0,0,2,3,0} */
int g[2][3] = {{1}};           /* 等价于 {1,0,0,0,0,0} */
```

但是以下初始化语句有错误：

```
int i[2][3] = {0,1,2,3,4,5,6};          /* 超越数组上限 */
int j[2][3] = {0,{1}};                  /* 嵌套括号位置不正确 */
int k[2][3] = {{{1}}};    /* 嵌套括号的层数超过了数组的维数 */
```

编译器将把数组初始化翻译为连续的三地址码赋值语句，如上例中的数组a的初始化翻译为：

```
a[0] = 0; a[4] = 1; a[8] = 2; a[12] = 3; a[16] = 4; a[20] = 5;
```

数组初始化必须遵循以下原则:
(a) 对数组元素赋初值的次序以最右边的下标变化最快为原则.
(b) 初始化列表的元素的个数一定要小于或等于数组元素的个数. 若小于，则数组余下的元素将赋初值为0. 如上例中的数组b.
(c) 多维数组的初始化可用单一的括号对，也可用嵌套的括号对. 后者的含义是对元素为数组的数组赋初值. 如上例中的d的初始化, {0,1}和{2}是对d[0]和d[1]分别赋初值.
(d) 因此嵌套括号在初始化列表中所在的位置必须正好是对某一元素为数组的第一个元素开始赋初值的位置. 如上例中的c，d，e，f和g. 但数组j的初始化列表中嵌套括号{1}所在的位置是为j[0][1]赋初值，并不是准备为j[0][0]或j[1][0]赋初值所在的位置.

为完成对数组元素赋初值的翻译，特设计以下属性：
(e) 全局变量count记录当前已赋初值的数组元素的个数. 其初值为0, 每对数组元素赋一个初值，其值累加1.
(f) 函数getsize(int dim)可获得当前所赋值数组在第dim维的长度. 如当前数组为int a[4][3][2]，则getsize(0)=24，getsize(1)=6，getsize(2)=2，getsize(3)=1, 和getsize(4)=error.
(g) 对非终结符I和L, 定义两个继承属性level和limit. level对应在句型中嵌套I或L的花括号层数, 如句型"{L,{{I}}}"中，L.level = 1，I.level = 3. 属性level即是当前初始化数组元素所对应的维数. 属性limit是当前I或L能初始化元素个数的绝对上限. L.limit = count + getsize(L.level). 例如，若上例句型是对数组int a[2][3][4]的初始化, 则在L处，count = 0，L.limit = 24; 在I处，count必须是a的第二维长度的倍数，即2的倍数，否则不满足(d). 若此处count = 6，则I.limit = 6 + 2 = 8.
(h) 综合属性n.val是整数常量n所对应的数值. 设它在词法分析中已计算出来.
(i) 设在语法制导语义定义中还可使用：函数getname(void)返回当前初始化数组的数组名; emit(string)输出三地址指令; error(char *message)报错并退出. (10pt, 5+5)

(1) 完成下述整数数组初始化语句到三地址码转换的语法制导语义定义（见下页）;

(2) 试写出下述初始化语句对应的三地址码(**仅列出非零的赋初值语句即可**).

```
int a[3][2][4]={{{1},{2,3},{{4},{5,6}}};
```

产生式	语义规则
$I' \to MI$	I.level $= 0$, I.limit $= 1$
$M \to \varepsilon$	count $= 0$
$I \to \{LN\}$	/* N为余下的数组元素赋初值0 */
$I \to n$	
$N \to \varepsilon$	
$L \to L_1, L_2$	
$L \to I$	

六、 设有如下Pascal程序片段： (5分)

```
while (not (a > b) and (c > d)) do
  begin
   if not (e > f) or not (i > j) then
     x := x + 1;
   else
     continue;
   if not (m > n) then break;
   y := y + 2;
end;
```

其对应的三地址码如下所示

```
L1:  [   ] (a > b) goto L__ |      [   ] (m > n) goto L__
     [   ] (c > d) goto L__ |      t1 := y + 2
     [   ] (e > f) goto L__ |      y := t1
     [   ] (i > j) goto L__ |      goto L__
L0:  t0 := x + 1            | L2:
     x := t0                |
```

试为其中空白“__”填上正确的标号编号，并为空白“[ ]” 填上if或ifnot.

七、 设有如下C语言程序： (5分)

```
int count = 0;
int fac(int n)
{
  count++;
  if (n == 0) return 1;
  return n * fac(n - 1);
}
int main()
{
  printf("%d, %d\n", fac(5), count);
  return 0;
}
```

编译后运行输出的结果为120，0. 问输出的count为什么是0，而不是6.