编译原理期末复习

该资料来源于百度文库，仅供大家复习时参考，不作为考试范围。

鉴于编译原理马上就要期末考试，我将手中集中的一些资料上的题目进行了整理归类，每种类型题目给出了所涉及到的基本知识，然后对每类题目中的第一道例题进行了做法进行了讲解，剩下的例题请给大家作为练习，答案也都给出，希望对大家复习有所帮助，最后由于时间很紧，整理的有些仓促，整理中难免有遗漏或错误，请大家见谅。

注：下面出现的字母中，若无特别说明，小写英文字母为终结符，大写英文字母为非终结符，希腊字母为终结符与非终结符的任意组合。

1、简答题（或者名词解释）

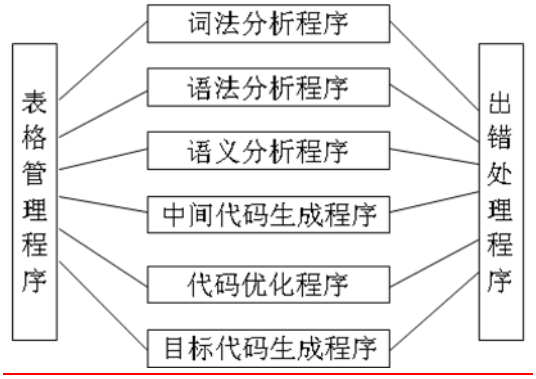
下面涉及到的概念中，加下划线的都是在以往一些试卷中出现的原题，务必掌握。

注：这类题目老师说答案不会超过一百个字，否则写的再多也不给分，有些点到即可，不要重复啰嗦。

（1）简述编译程序的概念及其构成

答：1）编译程序：它特指把某种高级程序设计语言翻译成等价的低级程序设计语言的翻译程序。

2）构成：



（2）简述词法分析阶段的主要任务（也有可能问语法分析阶段主要任务）

答：词法分析的任务是输入源程序，对源程序进行扫描，识别其中的单词符号，把字符串形式的源程序转换成单词符号形式的源程序。

语法分析的主要任务是对输入的单词符号进行语法分析（根据语法规则进行推导或者归约），识别各类语法单位，判断输入是不是语法上正确的程序

（3）简述编译程序的构造过程（这个大家看看，是对（1）和（2）的综合）

答：1）构造词法分析器：用于输入源程序进行词法分析，输出单词符号；

2）构造语法分析器：对输入的单词符号进行语法分析，识别各类语法单位，判断输入是不是语法上正确的程序

3）构造语义分析和中间代码产生器：按照语义规则对已归约出的语法单位进行语义分析并把它们翻译成中间代码。

4）构造优化器：对中间代码进行优化。

5）构造目标代码生成器：把中间的代码翻译成目标程序。

6）构造表格管理程序：登记源程序的各类信息和编译各阶段的进展情况。

7）构造错误处理程序：对出错进行处理。

(4) 说明编译和解释的区别：

1)编译要程序产生目标程序，解释程序是边解释边执行，不产生目标程序；

2)编译程序运行效率高而解释程序便于人机对话。

（5） 文法：描述语言语法结构的形式规则，一般用一个四元式表示： $G=(V_T, V_N, S, P)$，其中 V_T：终结符集合(非空) V_N：非终结符集合(非空)，且 $V_T \cap V_N=\varnothing$ S：文法的开始符号，$S \in V_N$ P：产生式集合(有限)。

（6）二义性文法：一个文法中存在某个句子，它有两个不同的最左（或者最右推导），则称该文法是二义性的。

例子如文法 G：S→if expr then S |other

S→if expr then S else S 句子 if e1 then if e2 then s1 else s2 是二义性的。

（7）文法的形式（注：文法的形式一定要牢记，特别是 2 型和 3 型文法一定要牢记，不仅在概念题中有用，在下面的根据语言写文法题中也有重要作用，如果所写的文法形式不对，一切都是无用功……）

1）0 型文法(短语文法，图灵机)：产生式形式为：$\alpha \rightarrow \beta$ ，其中：$\alpha \in (V_T \cup V_N)^*$且至少含有一个非终结符；$\beta \in (V_T \cup V_N)^*$

2）1 型(上下文有关文法，线性界限自动机)：产生式形式为:$\alpha \rightarrow \beta$ 其中：$|\alpha| \leq |\beta|$，仅 $S \rightarrow \varepsilon$ 例外，但是 S 不得出现在任何产生式右部。

3）2 型文法（上下文无关文法，非确定下推自动机)：产生式形式为：$P \rightarrow \alpha$， $P \in V_N$， $\alpha \in (V_T \cup V_N)^*$ 。

4）3 型文法(正规文法，有限自动机)：右线性文法：产生式形如：$A \rightarrow \alpha B$ 或 $A \rightarrow \alpha$ 其中：$\alpha \in V_T^*$；A，$B \in V_N$ 左线性文法：产生式形如：$A \rightarrow B\alpha$ 或 $A \rightarrow \alpha$ 其中：$\alpha \in V_T^*$；A，$B \in V_N$

例题：设 $G=(V_T, V_N, S, P)$是一个上下文无关文法，产生集合 P 中的任一个产生式应具有什么样的形式？若 G 是 1 型文法呢？若 G 是正则文法呢？

上下文无关文法， 产生式形式为：$P \rightarrow \alpha$， $P \in V_N$， $\alpha \in (V_T \cup V_N)^*$ 。

1 型文法：产生式形式为:$\alpha \rightarrow \beta$ 其中：$|\alpha| \leq |\beta|$，仅 $S \rightarrow \varepsilon$ 例外，但是 S 不得出现在任何产生式右部。

正则文法：右线性文法：产生式形如：$A \rightarrow \alpha B$ 或 $A \rightarrow \alpha$ 其中：$\alpha \in V_T^*$；A，$B \in V_N$

左线性文法：产生式形如：$A \rightarrow B\alpha$ 或 $A \rightarrow \alpha$ 其中：$\alpha \in V_T^*$；A，$B \in V_N$

(8)什么是 PDA(下推自动机)？

答：PDA 是下推自动机，PDA M 用一个七元组表示(K, Σ, f, H, h0, S, Z)

K：有穷状态集 Σ：输入字母表(有穷) H:下推栈符号的有穷字母表

h0 :H 中的初始符号 f：K × (Σ∪{ε}) × H -> K × H* S：属于 K 是初始状态。

Z:包含于 K，是终结状态集合。

(9) 什么是 DFA（有穷状态自动机）？

自动机 M 是一个五元式 $M=(S, \Sigma, f, S_0, F)$,其中：

1）S：有穷状态集， 2） Σ：输入字母表(有穷)，

3） f：状态转换函数，为 S×Σ→S 的单值部分映射，f(s，a)=s’表示：当现行状态为 s，输入字符为 a 时，将状态转换到下一状态 s’。我们把 s’称为 s 的一个后继状态。

4） $S_0 \in S$ 是唯一的一个初态；　　5） $F \subset S$ ：终态集(可空)。

（10）推导：所谓推导就是对于一个含非终结符 A 的符号串，利用规则 A→α，把 A 替换成 α 得到新符号串的过程。

最左推导：在推导的每一步，选择符号串最左边的非终结符进行替换。

最右推导：在推导的每一步，选择符号串最右边的非终结符进行替换。

(11)归约：归约是推倒的逆过程，即用产生式的左部非终结符替换右部符号串。

(12)什么是句型？什么称为句子？什么是语言？

句型：从文法的起始符号出发，经过有限步的推导能够推导出来的符号称为句型。

句子：只由终结符构成的句型称为句子。

语言：所有句子的集合构成该文法描述的语言。

(13)什么是短语？什么是直接短语（也称为简单短语）？什么是句柄？（以下几个概念一定要理解，考试中肯定会考哪些是短语，直接短语，句柄等，具体方法请参见后面的）

短语：如果存在某个文法非终结符 P，满足 $S \overset{*}{\Rightarrow} \beta P \gamma$，并且 $P \overset{+}{\Rightarrow} \alpha$ 则称 α 为句型 β α γ 相对于非终结符 P 的短语。

直接短语：如果 $P \Rightarrow \alpha$，即从 P 出发一步推出 α，则 α 称为直接短语。

句柄：一个句型的最左直接短语称为该句型的句柄。

（14）自顶向下的语法分析方法中需要解决的主要问题什么？如何表示？

答：1）主要需要解决回溯和左递归问题。

2）表示方式，回溯：文法中存在形如 $A \rightarrow \alpha_1 | \alpha_2 | \cdots | \alpha_n$，即产生式右部存在多个候选，导致推导时不能确定到底应该选择哪个候选式。

左递归：文法中存在形如 P→Pα 的形式，推导时会导致推导过程无休止的进行下去。

注：解决方法，对回溯采取的是提取左公因子，对左递归使消除左递归（包括直接左递归和间接左递归）。

2、最左最右推导，画语法树，找短语、直接短语、句柄等。

这种题目很简单，送分题，一定不能丢分！

考题：1）文法 G[S]为：S→SdT | T　T→T<G | G　G→(S) | a 试给出句型(SdG)<a 的推导过程及语法树，并找出(SdG)<a 的短语，直接（简单）短语、句柄和最左素短语。

分析：（1）推导和画语法树这些都很简单，不再赘述。

（2）根据所画出的语法树，可以很快的找出短语，直接短语，句柄和最左素短语等，先讲一个简单子树的概念，所谓简单子树是指仅具有单层分支的子树。

具体方法如下：

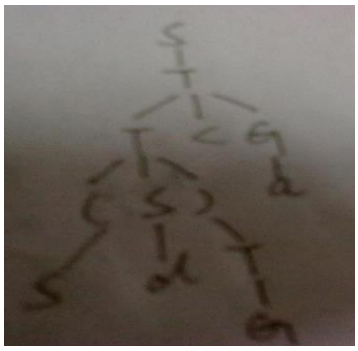
短语：任一子树的树叶全体(具有共同祖先的叶节点符号串)皆为短语；

直接短语：任一简单子树的树叶全体(具有共同父亲的叶节点符号串)皆为简单短语；

句柄：最左边的直接短语；

答：（1）S⇒T⇒ T<G ⇒G<G⇒(S)<G⇒(SdT)<G⇒(SdG)<G⇒(SdG)<a

语法树：

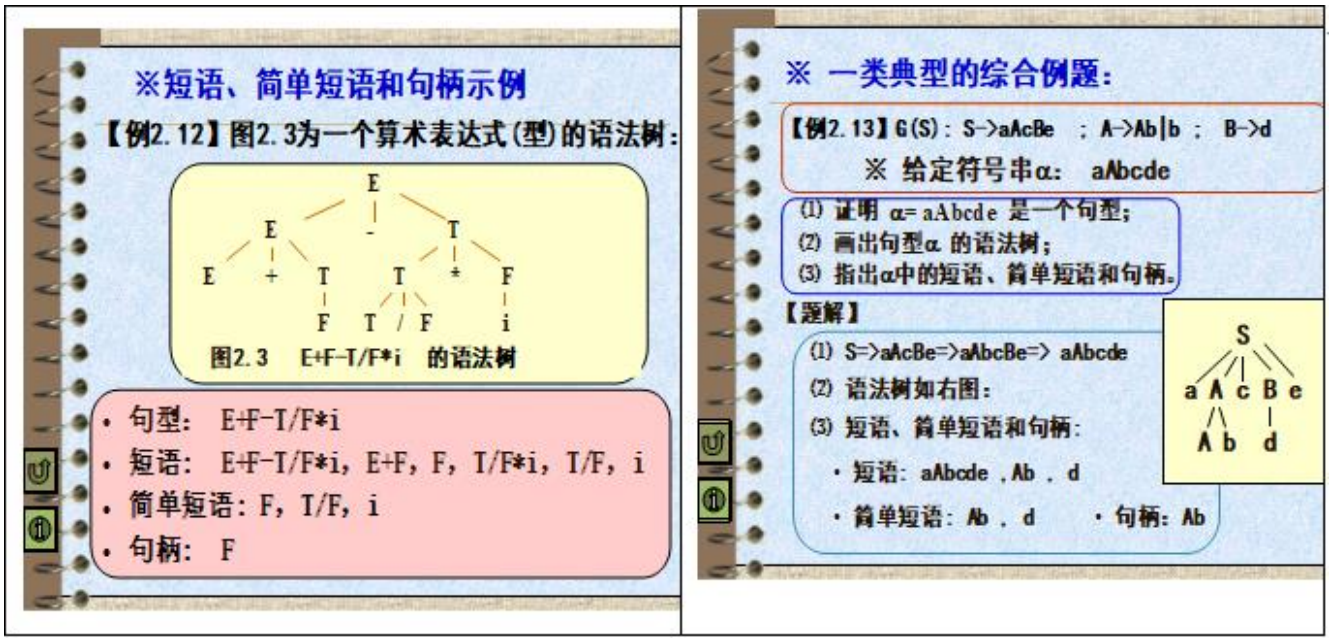


（2）短语：G，SdG，(SdG)，a，(SdG)<a　　直接短语：G，a　句柄：G 最左素短：SdG

注:还有一份试卷将句型改为 adT<(S)，与这个类似，大家自己做做，答案是

短语：a,(S),T<(S),adT<(S)　　直接短语：a,(S)　句柄：a

下面两道例题大家看看，一定要会找短语，直接短语，句柄等.



考题：2) 文法 G[E]的产生式为 E→E+T|T　　T→T*F|F　　F→i|(E)

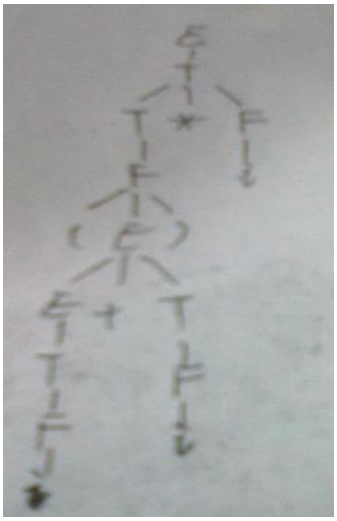
①对于句型(i+i)*i，给出最左最右推导及相应的推导树

②列出句型的所有短语、简单短语。（还有一份试卷上是出句型 F+T*F 的所有短语、简单短语和句柄，大家自己做做）

答：（1）最左推导：E⇒T⇒T*F⇒F*F⇒(E)*F⇒(E+T)*F⇒(T+T)*F⇒(F+T)*F⇒(i+T)*F⇒(i+F)*F⇒(i+i)*F⇒(i+i)*i

最右推导 E⇒T⇒T*F⇒T*i⇒F*i⇒(E)*i⇒(E+T)*i⇒(E+F)*i⇒(E+i)*i⇒(T+i)*i⇒(F+i)*i⇒(i+i)*i

推导树如下：



（2）短语：i，i+i，(i+i)，(i+i)*i　直接短语：i　句柄：i

3、根据语言推文法

这类题目首先要看清题目，指定的是什么文法，一般都是 2 型（上下无关文法）或者 3 型（正规文法），这两类文法形式一定要记住，具体请参见第 2 页的文法分类。

掌握几个基本形式 $\{a^n \mid n>0\}$ **对应文法** $S\to aS \mid a$ **如果是** $n>=0$ **则为** $S\to aS \mid \varepsilon$（ε **是空字**）

$\{a^nb^n \mid n>0\}$ **对应文法** $S\to aSb \mid ab$

下面这四道题目老是在试卷中重复出现，所以大家好好看看。

考题

1、按指定类型给出下列语言的文法,并指出语言的类型。（10 分） (1)L1={$a^nb^mc^n$| n≥0,m>0 } 二型文法：S→aSc|B　B→bB|b (2) L2={ $0^na1^nb^mc^m$| n>0，m ≥0} 二型文法：S→AB A→0A1|0a1 B→bBc|ε	2、按指定类型给出下列语言的文法。（10 分） (1)L1={ ca^ndb^m| n≥0,m>0 } 用正规文法。 S→cA　A→aA|B　B→dC C→bC|b (2)L2={ $0^na\ 1^nb^mc^m$| n>0,m ≥0}用二型文法。 S→AB　A→0A1|0a1　B→bBc|ε
3、按指定的类型给出下列语言的文法（10 分） (1)L1={ ca^nb^m| n≥0,m>0 } 用正规文法。 S→cA　A→aA|B　B→bB|b (2)L2={ $0^na\ 1^nb^m$| n>0，m ≥0} 用二型文法。 S→AB　A→0A1|0a1　B→bB|ε	4、按指定的类型给出下列语言的文法（10 分） (1) L1={a^nb^mc|n>=0,m>0}用正规文法 S→aS|A　A→bA|bB　B→c (2) L2={$a0^n1^nbd^m$|n>0,m>0}用二型文法 S→AB　A→aT　T→0T1|01 B→bD　D→dD|d

这是书 P36 第 11 题的答案如下：大家作为练习，可以发现比上述题目简单的多了。

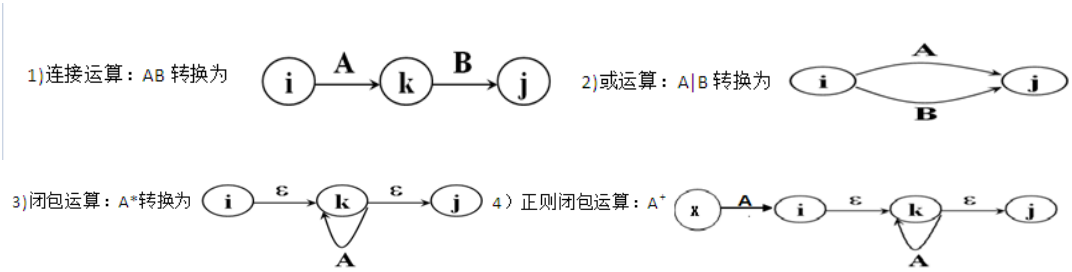
G1:	G2:	G3:	G4:
S→AC	S→AB	S→AB	S→1S0|A
A→aAb|ab	A→aA| ε	A→aAb| ε	A→0A1| ε
C→cC| ε	B→bBc|bc	B→aAb| ε	

或者 G4:

$$S \to A \mid B$$
$$A \to 0A1 \mid \varepsilon$$
$$B \to 1B0 \mid A$$

4、词法分析———正规式、NFA 和 DFA 之间的转化：

（1）这类题目一般是三者之间的转换，正规式→NFA→确定化的 DFA→最小化的 DFA，有时已经给出 NFA 了，则只需要确定化为 DFA 和最小化就行了，一般每一步都是五分。具体怎么转换请参照我期中考试时整理的提纲，主要就是下面这幅关系对照图，因时间和篇幅限制不再追溯。

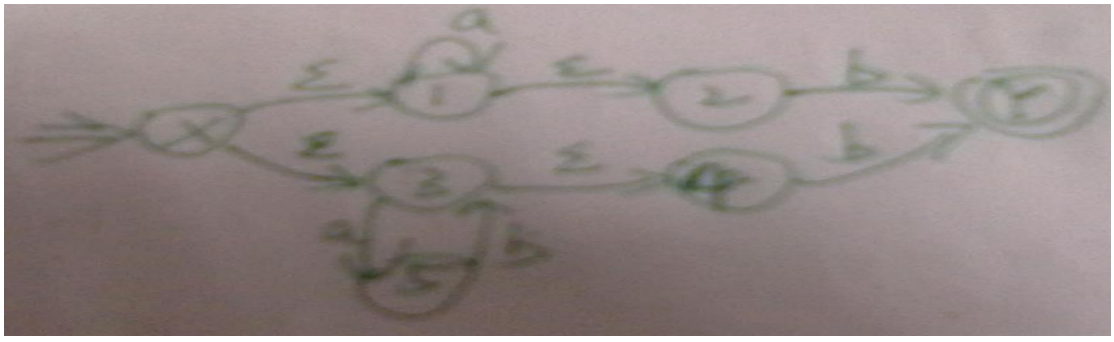


（2）注意优先级关系，闭包运算*最高，连接运算.次之，或运算|最低。

(3)考题 1：

1）构造正则式 a*b|(ab)*b 对应的 DFA。（15 分）

①画出 NFA



②确定化 DFA

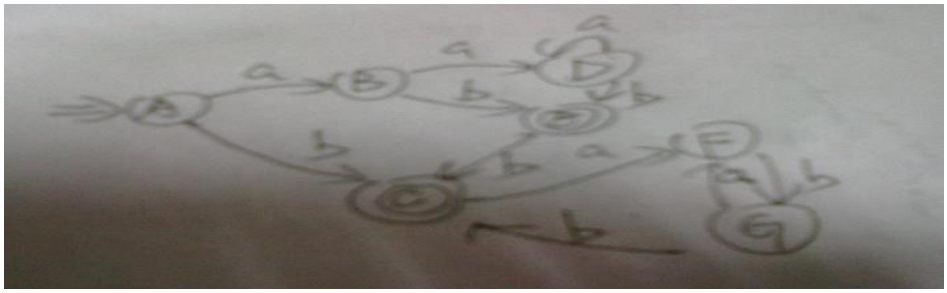
X	I_a	I_b
{X, 1, 2, 3, 4}	{1, 2, 5}	{Y}
{1, 2, 5}	{1, 2}	{3, 4, Y}
{Y}	–	–
{1, 2}	{1, 2}	{Y}

X	I_a	I_b
A	B	C
B	D	E
C	–	–
D	D	C

{3, 4, Y}	{5}	{Y}
{5}	–	{3, 4}
{3, 4}	{5}	{Y}

E	F	C
F	–	G
G	F	C

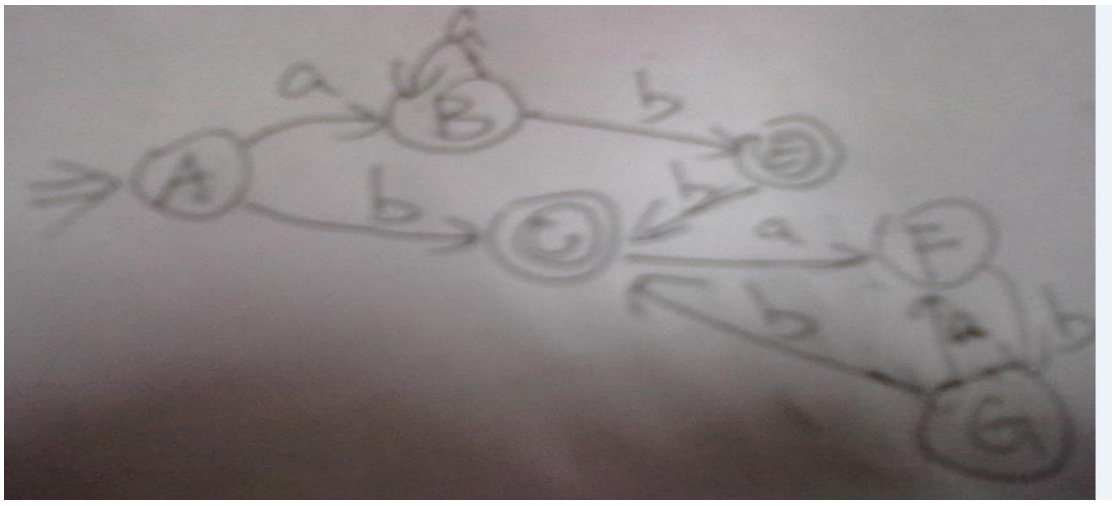
注：C 和 E 是终态



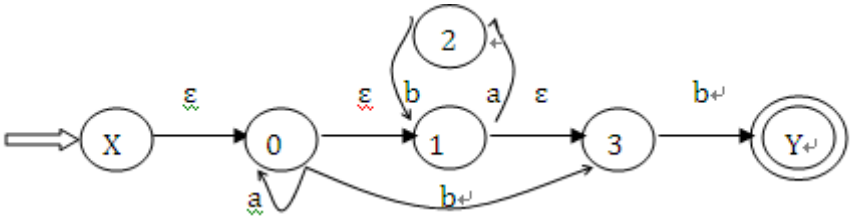
③最小化 DFA

首先将集合分为{A,B,D,F,G},{C,E}。{A,B,D,G}都有 a,b 作为条件输出，F 只有 b 输出，所以分为{A,B,D,G}和{F} 同理{C,E}分为{C},{E}。$\{A,B,D\}_a$={B,D} $\{G\}_a$={F}所以{A,B,D,G}分为{A,B,D}和{G}。$\{A\}_b$={C} $\{B,D\}_a$={D}所以分为{A} {B,D}

综上所述：划分的结果为{A},{B,D},{C},{E},{G}.



考题 2： 将 NFA 确定化为 DFA(10 分)



NFA:

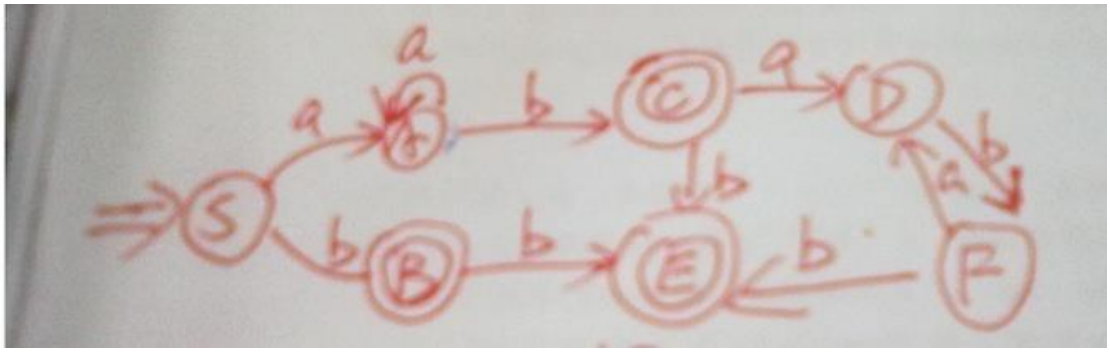
	Ia	Ib
{X,0,1,3}	{0,2,1,3}	{3,Y}
{0,2,1,3}	{0,2,1,3}	{1,3,Y}

{3,Y}	Φ	{Y}
{1,3,Y}	{2}	{Y}
{2}	Φ	{1,3}
{Y}	Φ	Φ
{1,3}	{2}	{Y}

DFA：

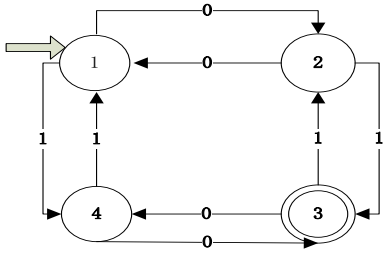
	a	b
S	A	B
A	A	C
B	Φ	E
C	D	E
D	Φ	F
E	Φ	Φ
F	D	E

含有 Y 的状态子集为 DFA 的终态,上例中的终态有 B,C,E.



题目中要求是确定化，没有要求最小化，如若最小化，划分的集合为{S,A},{B,C}{F},{D},{E}然后再画出最小化后的 DFA 这里不再赘述。

考题 3：构造奇数个 0 和奇数个 1 组成的自动机。（10 分）



状态 1：偶数个 0 偶数个 1　　状态 2：奇数个 0 偶数个 1

状态 3：奇数个 0 奇数个 1　　状态 4：偶数个 0 奇数个 1

如果题目改成偶数个 0，奇数个 1，只要将状态 4 转换成终态即可，其他类似。

5、语法分析——自顶向下分析法（LL（1）分析法和递归向下构造子程序法）

注：自顶向下分析法本质是由开始符号，按照产生式逐步推导看能否产生需要分析的句子。

（1）自顶向下分析中存在的问题

左递归和回溯（具体请参见简答题中的第（14）题）

（2）消除回溯——提取公因子法

提取公共左因子:假定关于 A 的规则 $A\to\delta\beta_1|\delta\beta_2|\ldots|\delta\beta_n|\gamma_1|\gamma_2|\ldots|\gamma_m$(其中，每个 γ 不以 δ 开头) 那么，可以把这些规则改写成 $A\to\delta A'|\gamma_1|\gamma_2|\ldots|\gamma_m$

$A'\to\beta_1|\beta_2|\ldots|\beta_n$

（3）消除左递归

1)消除直接左递归：直接消除见诸于产生式中的左递归：假定关于非终结符 P 的规则为 $P\to P\alpha|\beta$ ，其中 β 不以 P 开头。 我们可以把 P 的规则等价地改写为如下的非直接左递归形式：$P\to\beta P'$　　$P'\to\alpha P'|\varepsilon$

注：一般而言，假定 P 关于的全部产生式是 $P\to P\alpha_1|P\alpha_2|\ldots|P\alpha_m|\beta_1|\beta_2|\ldots|\beta_n$

其中，每个 α 都不等于 ε，每个 β 都不以 P 开头　那么，消除 P 的直接左递归性就是把这些规则改写成：　$P\to\beta_1P'|\beta_2P'|\ldots|\beta_nP'$　　$P'\to\alpha_1P'|\alpha_2P'|\ldots|\alpha_mP'|\varepsilon$

例：文法 G(E):

E→E＋T | T　　T→T*F | F　　F→(E) | i

经消去直接左递归后变成:

E→TE′　　E′→+TE′ | ε　　T→FT ′　　T ′→*FT ′ | ε　　F→(E) | i

2)消除间接左递归

这个请参见我期中整理的提纲篇幅较多，这里不再重复赘述，请参照下面的考题 2。

考题 1：将文法 G[S]改写为等价的 G’[S],使得 G’[S]不含左递归和左公因子。

S→[A　A→B]|AS　B→aB|a

答：消除左递归和左公因子后的文法为：

S→[A　A→ B]A’　A’ → S A’ | ε　B→aB’　B’ →B| ε

考题 2：写出文法 G[A]:　A→Ba|Cb|c　B→dA|Ae|f　C→Bg|h 消除左递归后的文法。

答：(1)经分析发现文法 G[A]并无直接左递归；

(2)消除间接左递归：将 A,B,C 按照 B,C,A 排序（建议将 A 放在最后）由于出现 C→Bg 形式，将 B 代入 C 得：C→dAg|Aeg|fg|h 又由于 A 出现 A→Ba　A→Cb 将 B,C 带入 A 得到：A→dAa|Aea|fa|dAgb|Aegb|fgb|hb|c 等价于 A→Aea|Aegb |dAa|fa|dAgb |fgb|hb|c

将 A 消除直接左递归 A→dAaA’|fa A’|dAgb A’ |fgb A’|hb A’|c A’　A’→ea A’| egb A’|ε

此时，对于 B→dA|Ae|f，C→dAg|Aeg|fg|h 由于未在任何产生式的右部出现，所以是多余的。

综上所述:消除递归后的文法为：A→dAaA’|fa A’|dAgb A’ |fgb A’|hb A’|c A’

A’→ea A’| egb A’|ε

(4)LL(1)分析法

1）含义：LL(1)分析方法（也叫预测分析法）：是指从左到右扫描、最左推导(LL)和只查看一个当前符号（括号中的 1）。

2）判断一个文法是否是 LL(1)文法的充要条件：

1. 文法不含左递归，
2. 对于文法中每一个非终结符 A 的各个产生式的候选首符集两两不相交。即，若

$A\to\alpha_1|\alpha_2|\ldots|\alpha_n$　则　$FIRST(\alpha_i)\cap FIRST(\alpha_j)=\phi$　　$(i\neq j)$

3. 对文法中的每个非终结符 A，若它存在某个候选首符集包含 ε，则 $FIRST(\alpha_i)\cap FOLLOW(A)=\phi$　i=1,2,...,n 如果一个文法 G 满足以上条件，则称该文法 G 为 LL(1)文法。

(5)LL(1)文法分析表的构造与运用

这类题目，主要就是判断文法是否满足 LL(1)文法的三个充要条件，分为以下几步：

1）首先将文法分解，判断是否有左递归好，有的话肯定不是 LL(1)文法；

2）算非终结符的 First 集合和 Follow 集合，具体方法请参见期中考试的提纲，其实最本质的要抓住 first 集合是 $U\overset{*}{\Rightarrow}a\cdots$，Follow 集合右 …Ua…即可。

3）算 Select 集合，书上没有提到 Select 集合，计算 Select 集合实质就是计算 First(α)，当然考试时不一定非要写成 Select 集合形式，可以直接计算 First(α)来判断交集是否为空，从而是否为 LL(1)文法，请看考题 1。

4）至于判断一个句子的分析过程，大家注意一下，所给的句子不一定是能通过该文法分析出来的，实际上在分析之前可以自己根据文法推推看，请看考题 1.

5）分析表中没有填的空格代表出错，如果分析时遇到了代表该句子不符合该文法。

考题 1：判断下面文法是否为 LL(1)文法，若是，请构造相应的 LL(1)分析表并分析句子 aabe 的分析过程。（15 分）

S→aD　D→STe|ε　T→bM　M→bH　H→M|ε

分析：判断一个文法是否为 LL(1)文法是否为 LL(1)文法，主要就是判断是否满足 LL(1)文法的充要条件，有一个不满足就不是 LL(1)文法。对于 aabe 根据文法 S⇒aD⇒aSTe⇒aaDTe⇒aaTe⇒aabMe 由于 M 不能为空字 ε，所以最后肯定推不出来，也就是该句子不能由该文法推出，出错。

当然一般题目都是 LL(1)文法，否则题目就不好往下做，没有意义。

答：(1)经分析该文法无左递归；

(2)①非终结符的 First 和 Follow 集合如下表所示

非终结符 X	First(X)	Follow(X)
S	a	# b
D	ε a	# b
T	b	e
M	b	e
H	ε b	e

②将文法分解为：　S→aD　D→Ste　D→ε　T→bM　M→bH　H→M　H→ε

依次计算：

First(aD)={a}　First(Ste)={a}

Follow(D)={# b}　First(bM)={b}

First(bH)={b}　First(M)={b}　Follow(H)={ e }

∵First(Ste)∩Follow(D)=Φ　First(M)∩Follow(H)=Φ

∴该文法是 LL(1)文法

LL(1)分析表如下：

	a	b	e	#
S	S→aD			

D	D→STe	D→ε		D→ε
T		T→bM		
M		M→bH		
H		H→M	H→ε	

(3)aabe 的分析过程如下：

步骤	符号栈	输入串	所用产生式
0	#S	aabe#	
1	#Da	aabe #	S→aD
2	#D	abe#	
3	#eTS	abe#	D→STe
4	#eTDa	abe#	
5	#eTD	be#	
6	#eT	be#	D→ε
7	#eMb	be#	T→bM
8	#eM	e#	出错

考题 2：判断下面文法是不是 LL(1)文法，若是请构造相应的 LL(1)分析表，写出 aaabd 的分析过程。

S→aH　H→aMd|d　M→Ab|ε　A→aM|e

答:（1）可以判断该文法无左递归。

（2）将文法分解为分解：S→aH　H→aMd　H→d　M→Ab　M→ε　A→aM　A→e

求 First 和 Follow 集合

非终结符 X	First(X)	Follow(X)
S	a	#
H	a,d	#
M	ε, a,e	d,b
A	a,e	b

求 Select 集合

Select(S→aH)=First(aH)={a}　Select(H→aMd)=First(aMd)={a}

Select(H→d)={d}　Select(M→Ab)=First(Ab)={a,e}

Select(M→ε)=First(ε)UFollow(M)= Follow(M)={d,b}

Select(A→aM)=First{aM}={a}　Select(A→e)=First(e)={e}

∵Select(H→aMd) ∩ Select(H→d)= Φ

Select(M→Ab) ∩Select(M→e)=Φ

Select(A→aM) ∩Select(A→e)= Φ

∴该文法是 LL(1)文法。

（3）LL(1)分析表如下：

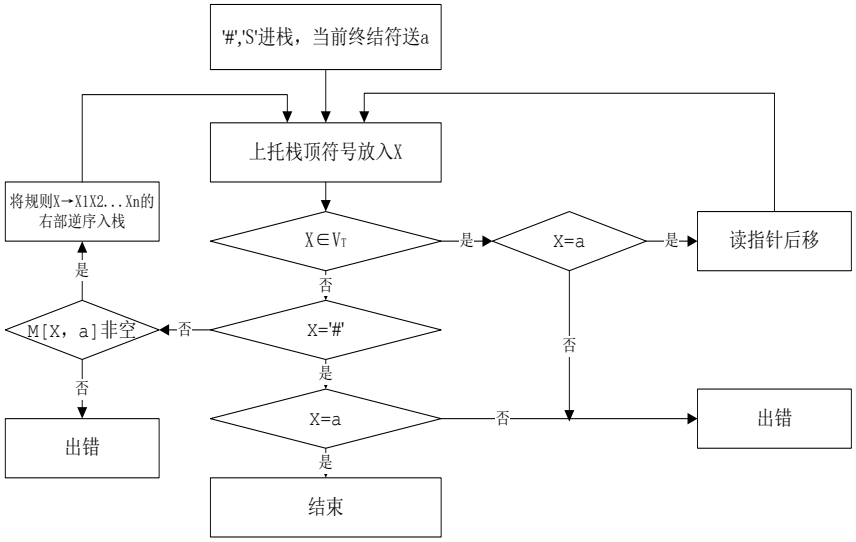
	a	d	b	e
S	S→aH			
H	H→aMd	H→d		

M	M→Ab	M→ε	M→ε	M→Ab
A	A→aM			A→e

aaabd#的分析过程如下：

步骤	符号栈	输入串	所用产生式
0	#S	aaabd#	
1	#Ha	aaabd #	S→aH
2	#H	aabd#	
3	#dMa	aabd#	H→aMd
4	#dM	abd#	
5	#dbA	abd#	M→Ab
6	#dbMa	abd#	A→aM
7	#dbM	bd#	
8	#db	bd#	M→ε
9	#d	d#	
10	#	#	

考题 3：构造 LL(1)分析方法的总控流程图



6、构造递归下降识别程序

这类题目首先看文法有无左递归和左公因子，有的话一定要消除，我期中给大家的答案错了，

考题 1 为更正后的答案。

考题 1：为文法 G[E]： E→V | V+E V→N | N[E] N→i 构造递归下降识别程序（10 分）

解：（1）提取左公因子：E→VE’ E’→+E|ε V→NV’ V’→[E]| ε N→i

(3) 构造递归下降识别程序

```
PROCEDURE  E;
BEGIN
    V; E'
END;

PROCEDURE  E';
IF SYM='+' THEN
    BEGIN
        ADVANCE;

PROCEDURE  V;
BEGIN
    N;V'
END;
```

```
                    E
                END
PROCEDURE   F;          PROCEDURE   N;
IF SYM= '['  THEN       IF  SYM='i'  THEN
   BEGIN                      ADVANCE
     ADVANCE;           ELSE  ERROR;
     E;
     IF SYM= ']'  THEN
        ADVANCE
     ELSE  ERROR
   END
 ELSE ERROR;
```

考题 2：为文法 G[E]:E→E+T|T　T→T*F|F　F→(E)|i 构造递归下降识别程序

解：（1）消除左递归：E→TE′　E′→+TE′ | ε　T→FT′　T′→*FT′ | ε　F→(E) | i

(2)构造递归下降识别程序

```
PROCEDURE   E;          PROCEDURE   T;          PROCEDURE   F;
BEGIN                   BEGIN                     IF SYM='i' THEN
        T; E′                   F; T′               ADVANCE
END;                    END                       ELSE
                                                    IF SYM='(' THEN
PROCEDURE   E′;         PROCEDURE   T′;             BEGIN
   IF SYM='+' THEN         IF SYM='*' THEN            ADVANCE;
   BEGIN                   BEGIN                      E;
     ADVANCE;                ADVANCE;                 IF SYM=')'
      T; E′                  F; T′                THEN ADVANCE
   END                     END                        ELSE ERROR
                                                    END
                                                  ELSE ERROR;
```

7、自底向上分析法——LR（0）分析法

注：自底向上分析法本质是从输入串开始，逐步进行“归约”，直到文法的开始符号,其关键问题是寻找句柄。

自底向上分析法还有 SLR(1),LR（1），LR(0)等等，这里以 LR(0)分析法为例，也是一份试卷上的考题，首先介绍一些相关知识。

（1）　LR(0)项目：通俗点讲，文法 G 中的任何一个产生式的右部的任何位置添加一个圆点就成了 LR(0)项目，比如 A→Ab 是产生式，则 A→•Ab A→A•b　A→Ab•都是该产生式对应的全部项目。项目动态的表示归约的一个阶段：对于项目 A→A•b，可以这样理解它：“•”之前的 A 表示的是在识别 Ab 过程中已输入到栈终的部分；“•”之后的表示在识别 Ab 过程中期望出现的部分；“•”表示则是在识别 Ab 过程中当前的识别进度的定位。项目 A→Ab•已经具备了识别 Ab 的一切条件，因此被称为归约项目。

项目可以分为以下四类：P→α•aβ 称为移进项目，P→α•Xβ 称为待约项目，其中 X 为非终结符，P→α• 称为归约项目，S’ →S 称为接收项目

（2）LR(0)的分析栈可以看成两部分状态栈

LR 分析器的核心是一张分析表：

ACTION[s，a]：当状态 s 面临输入符号 a 时，应采取什么动作.

GOTO[s，X]：状态 s 面对文法符号 X 时，下一状态是什么.

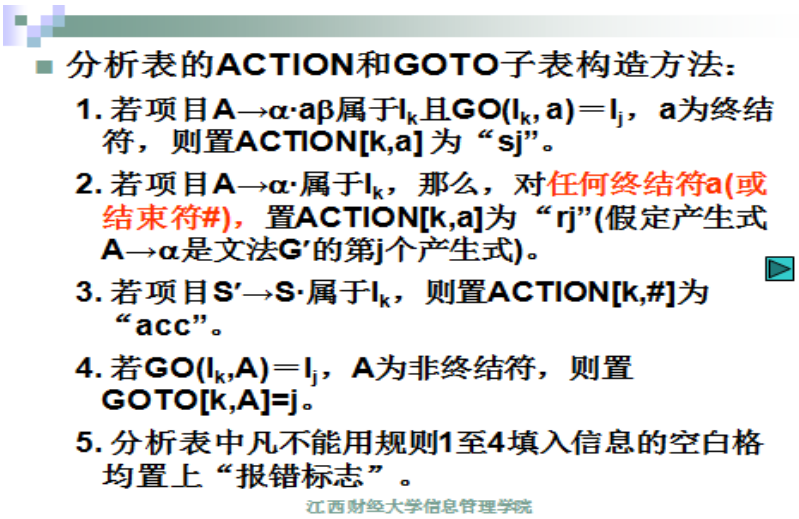
这一章的概念较多较抽象，我一时半会也讲不完讲不清楚，这里不再赘述，直接看题目，知道怎么做就行，下面以一道考题这也是老师讲的原题为例讲解如何做这类题目，首先一般这种题目分三步走：

(1)增广文法：假定文法 G 是一个以 S 为开始符号的文法，我们构造一个 G′，它包含了整个 G，但它引进了一个不出现在 G 中的非终结符 S′，并加进一个新产生式 S′→S，而这个 S′是 G′的开始符号。那么，我们称 G′是 G 的拓广文法。这样，便会有一个仅含项目 S′→S. 的状态，这就是唯一的“接受”态。

（2）构造识别文法的 DFA:对于任意的项目即 I,其闭包 CLOSURE(I)计算方法如下，I 中的所有项目都属于 CLOSURE(I);如果 P→ α •X β ,并且 X 为非终结符，则所有形如 X→•γ 的项目也属于 ClOSURE(I)定义函数 GO(I,X)，其中 I 是项目集，X 是任意的文法符号（终结符，非终结符都可以），GO(I，X)=CLOSURE(J).J 是从 I 中项目出发后读取 X 后到达的后继项目，即 J={P→ α X• β | P→ α •X β ∈I}

有了上述 CLOSURE 和 GO 的定以后，从 CLOSURE({S′→•S})出发，利用 GO 函数，产生出它在每个可能的文法符号下，要转移的项目集，再对新产生的项目集采取同样的方法直到没有新产生的项目集未被处理为止。

(4) 根据计算出的项目集之间的关系画出 DFA 和 LR(0)分析表,其中 LR(0)构造方法如下:



(5) 对具体的句子运用 LR(0)分析的方法如下：

每一项 ACTION[s，a]所规定的四种动作:

1. 移进 把(s，a)的下一状态 s’和输入符号 a 推进栈，下一输入符号变成现行输入符号.

2. 归约 指用某产生式 A→ β 进行归约. 假若 β 的长度为 r， 归约动作是， 去除栈顶 r 个项，使状态 s_{m-r} 变成栈顶状态，然后把(s_{m-r}, A)的下一状态 s’=GOTO[s_{m-r},

A]和文法符号 A 推进栈.

3. 接受（即 acc） 宣布分析成功，停止分析器工作.

4. 报错

考题：构造文法 G[E]的 LR(0)分析表，并给出 accd 的分析过程。

（0）S′→E（1）E→aA （2）E→bB （3）A→cA （4） A→d （5）B→cB （6）B→d

分析：题中已经进行了推广文法，所以第一步就不需要了，下面就是开始构造文法的 DFA，然后构造出 LR（0）分析表，最后在进行分析，实际上对于 accd 我们自己可以先推一下，E⇒aA⇒acA⇒accA⇒accd 所以该句子符合文法，那么最终构造出的 LR（0）分析表对该句子进行分析后必定得到“acc”（接受的意思），否则构造的 LR(0)分析表出错。

答：（1）构造文法的 DFA:

I_0=Closure({S′→•E })={ S′→•E, E→•aA, E→•bB }

I1=GO(I0,E)=Closure({S′→E•})={ S′→E• }

I_2=GO(I_0,a)=Closure({ E→a•A })={ E→a•A，A→•cA ， A→•d }

I_3=GO(I_0,b)=Closure({ E→b•B })={E→b•B, B→•cB ， B→•d }

（为了方便，下面计算中的 Closure 不再写了，直接给出答案,考试时可以不写，当然计算 I_0 是必须要的）

I_4=GO(I_2,A)={ E→aA•} I_5=GO(I_2,c)={ A→c•A，A→•cA ， A→•d }

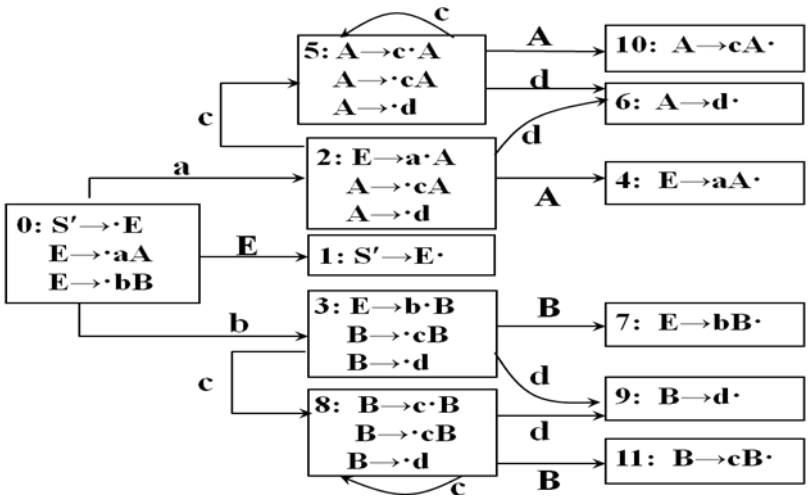
I_6= GO(I_2,d)={ A→d•} I_7=GO(I_3,B)={ E→bB•}

I_8= GO(I_3,c)= { B→c•B，B→•cB ， B→•d } I_9= GO(I_3,d)={ B→d•}

I_{10}= GO(I_5,A) ={ A→cA•} GO(I_5,c)=I_5 GO(I_5.d)=I_6

I_{11}= GO(I_8,B) { B→cB•} GO(I_8,c)=I_8 GO(I_8.d)=I_9

画出 DFA:



注：实际上构造 LR(0)分析表时这个图没有必要画，根据上述计算结果即可填写下表。

（2）画出 LR(0)分析表

	ACTION					GOTO		
状态	a	b	c	d	#	E	A	B
0	s2	s3				1		
1					acc			
2			s5	s6			4	
3			s8	s9				7
4	r1	r1	r1	r1	r1			
5			s5	s6			10	
6	r4	r4	r4	r4	r4			
7	r2	r2	r2	r2	r2			
8			s8	s9				11
9	r6	r6	r6	r6	r6			
10	r3	r3	r3	r3	r3			
11	r5	r5	r5	r5	r5			

注：至于怎么填这个表请参见上一页中的 PPT,这里不再赘述,Action 表中 si 代表跳转到第 i 个状态，ri 代表选择文法中第 i 条规则进行归约,acc 代表接受，即分析成功。Goto 表中数字 i 代表跳转到第 i 个状态。

（3）对 accd#进行分析

步骤	分析栈	输入串	操作
1	#0	accd#	s2
2	#0a2	ccd#	s5
3	#0a2c5	cd#	s5
4	#0a2c5c5	d#	s6
5	#0a2c5c5d6	#	r4
6	#0a2c5c5A10	#	r3
7	#0a2c5A10	#	r3
8	#0a2A4	#	r1
9	#0E1	#	acc

8、参数传递

这种题目很简单，是送分题，一定要做对！

参数传递方式分为三种，值传递，地址传递和传名。

值传递过程中形参值的改变不会影响实参值的改变，地址传递形参值的改变导致对应是实参值的改变，传名传递类似于 C 语言中的宏展开，将实参原封不动的替换相应的形参（文字替换）。

请看下题：

(1) 高级语言程序中常用的参数传递方式有哪些？请根据这些传递方式写出程序的运行结果。

```
static int a=1;
    int p(int x,int y,int z)
    {
    y=y+1;
    z=z+x;
```

```
    int main()
    {
    int a=2;
    int b=3;
```

```
    cout<<"In P "<<a<<endl;                    p(a,a,a);
    }                                          cout<<"In main "<<a<<endl;
                                               return 0;
                                               }
```

分析：函数 p 中输出地 a 的值是全局变量 a 的值,与主函数中的局部变量 a 无关，不要被迷惑住，所以 p 中输出的 a 就是 1.传值时，由于实参值不受影响，所以 a 仍为 2；传地址时，在函数 P 中，将实参代入后 a=a+1 得 a=3;再由 a=a+a 得到 a=6;传名时，直接将 实参带入形参，原来程序被替换为:a=a+1;a=a+a；将 a=2 代入得到结果为 6.

答：①C 语言中函数传递的方式有 1)传值　2）传地址　3）传名

②运行结果：

1)传值方式：In P 1　　2)传地址：In P 1　　3）传名：In P 1

In main 2　　In main 6　　In main 6

（2）

```
Procedure  p (x,y,z) ;                    begin
        begin                                   a:=2;
          y:=y+1;                               b:=3;
          z:=z+x;                               p(a+b , a , a)
      end; {p}                                  print  a
                                          end.
```

传值传递时输出结果为：　2　(实参值不受影响,a=2)

传地址传递时输出结果为：8　（a=2+1=3;a=3+5=8）

传名传递时输出结果为：9　　(a=a+1；a=a+a+b;得到 a=2+1=3;a=3+3+3=9)

9、程序分块、画 DAG 图

这个也比较简单，具体方法如下：

（1）分块就是将程序分成一块块的，主要就是确定每一块的入口语句：

1）寻找入口语句的方法：满足以下条件之一的都是入口语句

a）程序的第一条语句；b）条件转移语句或无条件转移语句可以转移到的语句

c)紧跟在条件转移语句后面的语句

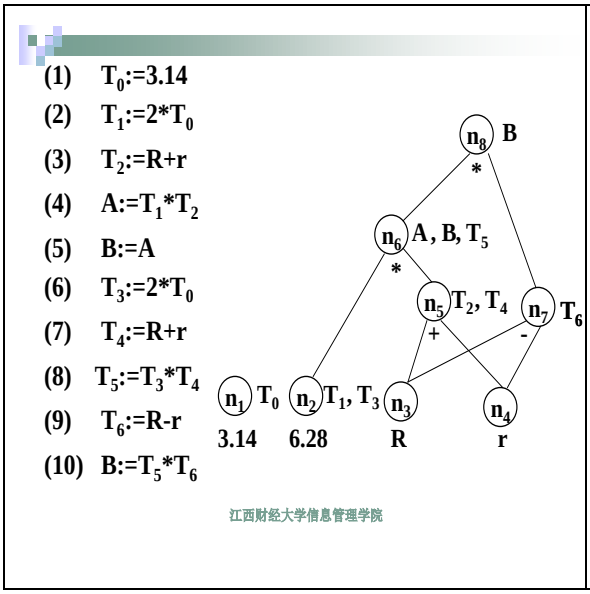
2）对每一入口语句，构造其所属的基本块。它是由该语句到下一入口语句（不包括下一入口语句），或到一转移语句（包括该转移语句），或到一停语句（包括该停语句）之间的语句序列组成的。

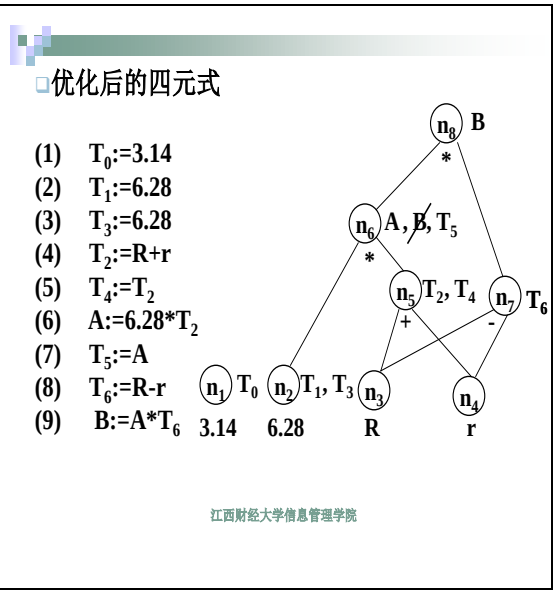
3）凡未被纳入某一基本块的语句，都是程序中控制流程无法到达的语句，因而也是不会被执行到的语句，我们可以把它们删除。

例如划分基本块:	划分结果
(1)　read X (2)　read Y (3)　R:=X mod Y (4)　if　R=0 goto (8) (5)　X:=Y (6)　Y:=R (7)　goto (3)	(1)　read X　（入口语句） (2)　read Y (3)　R:=X mod Y　（入口语句） (4)　if　R=0 goto (8) (5)　X:=Y　（入口语句） (6)　Y:=R (7)　goto (3)

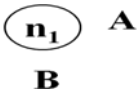
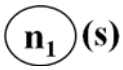
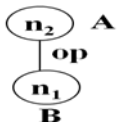
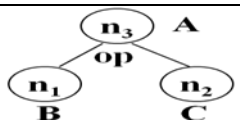
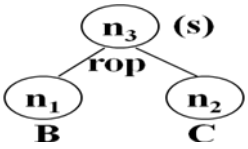
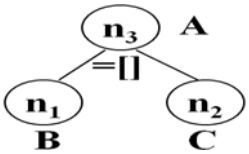
(8) write Y (9) halt	(8) write Y （入口语句） (9) halt

根据基本块画 DAG 图





上述图中的 DAG 只给出了最后完整的结果，老师要求在考试中需要把每一步都体现出来，具体请参见书上 284 页，因为篇幅限制所以不再赘述。另外，根据老师的意思可能会将 DAG 和四元式结合在一起考，给一段程序，先写出其四元式然后在画出其 DAG 图，四元式与 DAG 对应关系如下：根据下面的对应关系，DAG 图很容易画出来。

0 型： A:=B (:=, B, -, A)	
0 型: goto (s) (j, -, -, (s))	
1 型: A:=op B (op, B, -, A)	
2 型：A:=B op C (op, B, C, A)	
2 型:if B rop C goto (s) (jrop, B, C, (s))	
2 型: A:=B[C] (=[], B[C], -, A)	

3 型: D[C]:=B ([]=, B, -, D[C])