1. 判断：过程调用时传递参数，如果参数是表达式，则直接传递表达式的地址。（ 错 ）

如果是表达式，如A+B，先计算值并存放再某个临时单元T中，然后传送T的地址

1. 判断：过程调用时传递参数，如果参数是变量或数组，则直接传递变量或数组的地址。（ 对 ）
2. 在分支和循环中会用到条件表达式，而用作条件表达式的通常是布尔表达式。（ 对 ）
3. 程序设计语言中，布尔表达式的关系运算符的优先级，and的优先级高于not。（ 错 ）

Not优先级最高，其次是and，最后是or

1. 在对赋值语句进行翻译时，也要用到回填技术。（ 错 ）控制语句才需要回填技术
2. 程序设计语言中，布尔表达式有两个基本作用，一个是 逻辑演算/计算逻辑值 ，一个是 用于控制语句的条件表达式 。
3. 数组元素的地址计算与数组的 存储 方式有关，也与数组的类型及机器对存储器的编址方式有关。
4. 常用的参数传递方式有 传地址 、 传值 和 传名 。
5. 在过程调用时，参数传递方式为得结果，则每个形参对应两个形式单元，第一个形式单元存放 实参地址 ，第二个形式单元存放 实参的值 。
6. 以下说法不正确的是（ B ）

A. 为布尔表达式和控制流语句生成目标代码时，关键问题之一是确定跳转指令的目标标号

B. 在生成跳转指令时，就可以确定目标标号 不能确定，需要回填

C. 在生成跳转指令时，目标标号还不能确定

D. 可以将标号的地址作为继承属性传递到生成相关跳转指令的地方，但是这样的做法需要再进行一趟处理，将标号和具体地址绑定起来

1. 写出下列语句的中间代码

A[I, J] := B[I, J] + C[A[K, L]] +D[I+J]

假设A，B，C数组每维下界为1，每维的长度n1, n2，数组每个元素宽度为w则

一维数组的C1 = low \* w = w；

二维数组的C2 = (low1 \* n2 + low2) \* w = (n2 + 1) \* w;

从左到右依次写出对应的三地址代码：

A[I, J]: T1:= I \* n2

T2:= T1 + J

T3:= A – C2

T4:=w\*T2

T5:=T3[T4]

B[I, J]: T6:= I \* n2

T7:= T7 + J

T8:= B – C2

T9:=w\*T7

T10:=T8[T9]

A[K, L]: T11:= K \* n2

T12:= T11 + L

T13:= A – C2

T14:=w\*T13

T15:=T13[T14]

C[A[K, L]]: T16:= w\*T15

T17: = C – w

T18:=T17[T16]

D[I+J]: T19:= I+J

T20:=w\*T19

T21: = D – w

T22:=T21[T20]

T23:= T10+T18

T24:= T23+T22

T3[T4] := T24

1. 翻译如下语句成四元式序列

if a>b then x:=a+b\*c else x:=b-a;

第一步：只有一个条件控制语句1. if…then…else

第二步：if E1 then E2 else E3， E1(a>b), E2 (x:=a+b\*c), E3 (x:=b-a)

第三步：if E1 then M1 E2 N else M2 E3

第四步：

100: (j>, a, b, 0)

101: (j, \_, \_, 0)

102: (\*, b, c T1)

103: (+, a, T1, T2)

104: (:=, T2, \_, x)

105: (j, \_, \_, 0)

106: (-, b, a, T3)

107: (:=, T3, \_, x)

108: (j, \_, \_, 0)

109

第五步：E1 = 100, 101 E2 = 102-105 E3 = 106-108

M1 = E2 = 102 M2 = E3 = 106 N = 109

E1(a>b)条件判断语句：

E1.true 100 → M1= 102

E1.false 101→ M2 = 106

E2 (x:=a+b\*c)计算赋值语句：

105跳转→N = 109

E3 (x:=b-a) 计算赋值语句：

108跳转→109

第六步：

100: (j>, a, b, **102**)

101: (j, \_, \_, **106**)

102: (\*, b, c T1)

103: (+, a, T1, T2)

104: (:=, T2, \_, x)

105: (j, \_, \_, **109**)

106: (-, b, a, T3)

107: (:=, T3, \_, x)

108: (j, \_, \_, **109**)

109

1. 翻译如下语句成四元式序列

While a＞0 and b＜0 do Begin

 X：＝X＋1；

     if a＞0 then a：＝a－1 else b：＝b＋1

End；

第一步：两个条件控制语句1. while do;

2. if…then…else

一个计算赋值语句 X：＝X＋1

第二步：1. While E1 do Begin…End, E1(a＞0 and b＜0), Begin(E2, E3, E4, E5)End

E2(X：＝X＋1)

2. if E3 then E4 else E5， E3(a>b), E4 (a：＝a－1), E5 (b：＝b＋1)

第三步：1. While M1 E1 do M2 Begin(E2, E3, E4, E5)End

E2

2. if E3 then M3 E4 N else M4 E5

第四步：

100: (j>, a, 0, 0)

101: (j, \_, \_, 0)

102: (j<, b, 0, 0)

103: (j, \_, \_, 0)

104: (+, X, 1, T1)

105: (:=, T1, \_, X)

106: (j, \_, \_, 0)

107: (j>, a, 0, 0)

108: (j, \_, \_, 0)

109: (-, a, 1, T2)

110: (:=, T2, \_, a)

111: (j, \_, \_, 0)

112: (+, b, 1, T3)

113: (:=, T3, \_, b)

114: (j, \_, \_, 0)

115

第五步：E1 = 100-103 E2 = 104-106 E3 = 107-108 E4 = 109-111 E5 = 112-114

M1 = E1 = 100 M2 = E2 = 104 M3 = E4 = 109 M4 = E5 = 112 N = E1 = 100

E1(a＞0 and b＜0)布尔条件判断语句：

E1’(a>0), E1’’(b<0)

E1’.true 100 → E1’’.true = 102

E1’.false 101→ 115 E1.false条件不满足退出循环

E1’’.true 102→M2 = 104 E1.true条件满足进入循环

E1’‘.false 103→ E1’.false = 101 E1.false条件不满足退出循环

E2(X：＝X＋1)计算赋值语句：

106跳转→E3= 107

第三步：

1. While M1 E1 do M2 Begin(E2, E3, E4, E5)End

E2

2. if E3 then M3 E4 N else M4 E5

E3(a>b) 条件判断语句：

E3.true 107 → M3= 109

E3.false 108→ M4 = 112

E4 (a：＝a－1) 计算赋值语句：

111跳转→N= 100

E5 (b：＝b＋1) 计算赋值语句：

114跳转→M1= 100

第六步：

100: (j>, a, 0, **102**)

101: (j, \_, \_, **115**)

102: (j<, b, 0, **104**)

103: (j, \_, \_, **101**)

104: (+, X, 1, T1)

105: (:=, T1, \_, X)

106: (j, \_, \_, **107**)

107: (j>, a, 0, **109**)

108: (j, \_, \_, **112**)

109: (-, a, 1, T2)

110: (:=, T2, \_, a)

111: (j, \_, \_, **100**)

112: (+, b, 1, T3)

113: (:=, T3, \_, b)

114: (j, \_, \_, **100**)

115

1. 试为文法G[S]

S→（L）| a

L→L, S | S

* 1. 写出一个翻译方案，它输出配对括号的个数。如对于句子(a, (a,a))，输出结果为2；
  2. 写出一个翻译方案，它输出配对括号的最大嵌套深度。如对于句子(a, (a,a))，输出结果为2。

1. 分析：

产生式右部带括弧的只有S→（L），则规约时，S的括弧数等于L的括弧数+1；

产生式L→S，则规约时，L的括弧数等于S的括弧数；

产生式L→L, S，则规约时，L的括弧数等于L的括弧数+S的括弧数；

产生式S→a，则规约时，S的括弧数等于0；

为了输出S的括弧数，需要对原文法进行拓广，增加一条S’→S，对应语义为输出S的括弧数。

文法G[S]的语义规则：

S’→S {print (S.num)}

S→（L） {S.num = L.num+1}

S→a {S.num = 0}

L→L1, S {L.num = L1.num+S.num}

L→S {L.num = S.num}

1. 分析：

产生式右部带括弧的只有S→（L），则规约时，S的嵌套深度等于L的嵌套深度+1；

产生式L→S，则规约时，L的嵌套深度等于S的嵌套深度；

产生式L→L, S，则规约时，L的嵌套深度等于**max{L的嵌套深度, S的嵌套深度}**；

产生式S→a，则规约时，S的嵌套深度等于0；

文法G[S]的语义规则：

S’→S {print (S.num)}

S→（L） {S.num = L.num+1}

S→a {S.num = 0}

L→L1, S {L.num = Max(L1.num, S.num)}

L→S {L.num = S.num}