# 10.2 （1）

```
L0:
        if 1 >= x goto L3
        if y <= 1 goto L3
L1:
        t4 = 2 - n
        t5 = x * y
        if t4 < t5 goto L2
        a = 100
        goto L0
L2:
        b = a
        goto L0
L3:
```

图一：更贴近汇编

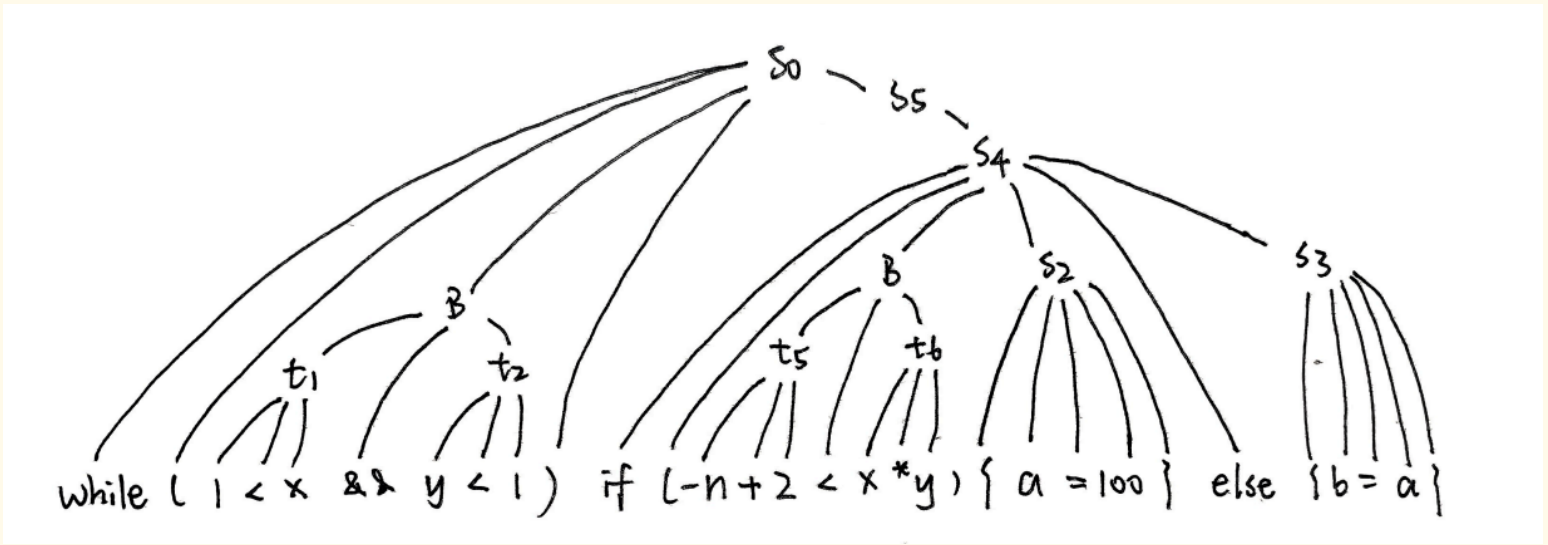
```
L0:
        t1 = 1 < x
        t2 = y > 1
        t3 = t1 && t2
        if t3 == 0 goto L3
L1:
        t4 = 2 - n
        t5 = x * y
        if t4 < t5 goto L2
        a = 100
        goto L0
L2:
        b = a
        goto L0
L3:
```

图二：更直观

整体逻辑正确即可，

（语法树上的节点记得写一些注释，B.code等，有一些就行）

最好不要使用 if ... then ... else ...



# 10.3

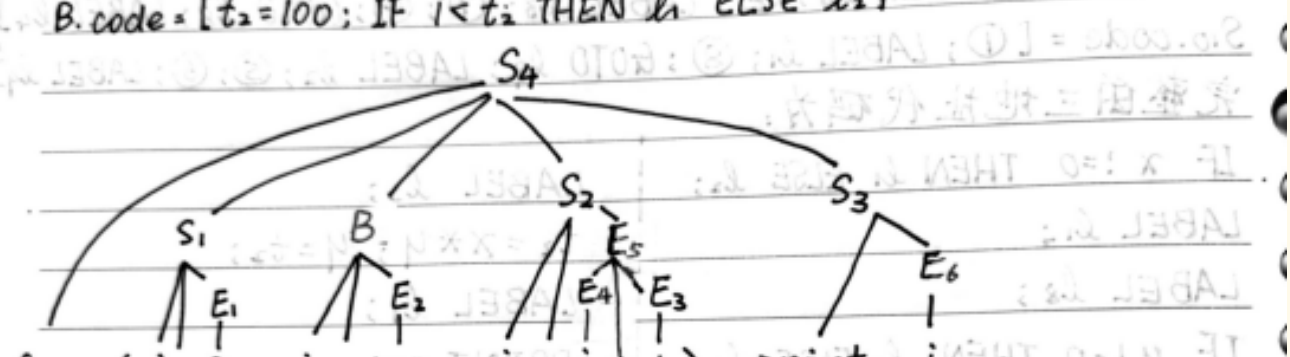
10.2 解：设计的文法为：

S → for (S ; B ; S) S { l = newlabel();
S[0].code = S[1].code ++ gen[LABEL ?l] ++
B.code ++ gen_l[LABEL ?B.tc] ++ S[3].code
++ S[2].code ++ gen[GOTO ?l] ++ gen_l[LABEL
?B.fc] }

翻译for ( i = 0; i < 100; i=i+1 ) print i

题给语句的带注释语法树为(注释在树外给出)：

$E_1.place = t_1$ $E_1.code = [t_1 = 0]$ $E_2.place = t_2$ $E_2.code = [t_2 = 100]$
$E_3.place = t_3$ $E_3.code = [t_3 = 1]$ $E_4.place = i$ $E_4.code = []$
$E_5.place = t_4$ $E_5.code = [t_4 = i + t_3]$ $S_2.code = [t_3 = 1; t_4 = i + t_3; i = t_4]$
$E_6.place = i$ $E_6.code = []$ $S_3.code = [PRINT\ i]$
$S_1.code = [t_1 = 0; i = t_1]$ $B.tc = (l_1)$ $B.fc = (l_2)$
$B.code = [t_2 = 100; IF\ i < t_2\ THEN\ l_1\ ELSE\ l_2]$




完整的三地址代码(也是$S_4$的code域的值)为，

| | |
|---|---|
| $t_1 = 0$; $i = t_1$ | $t_3 = 1$; $t_4 = i + t_3$; |
| LABEL $l_3$; | $i = t_4$; |
| $t_2 = 100$; | GOTO $l_3$; |
| IF $i < t_2$ THEN $l_1$ ELSE $l_2$; | LABEL $l_2$; |
| LABEL $l_1$; | |
| PRINT i; | |

# 10.5

### 1. `@table`（全局符号表）

```
@table:
x         : int
z         : float
a         : int[10,20]
b         : int[6]
bar       : function(float) <- (int[6])
foo       : function(float) <- (int, function(float)<-(int), int[10,10])
```

### 2. `@code`（全局代码）

只包含函数定义的跳转：

```
@code:
GOTO    main
LABEL   bar
CALL    bar@code
LABEL   foo
CALL    foo@code
```

### 3. `bar@table`（bar 函数的符号表）

```
bar@table:
brr       : int[6] (参数)
x         : float  (局部变量)
```

### 4. `bar@code`（bar 函数体代码）

```
t1 = brr[0]
t2 = brr[5]
t3 = t1 + t2
x = t3
return x
```

### 5. `foo@table`（foo 函数的符号表）

```
foo@table:
x         : int                (参数)
boo       : function(float)<-(int)  (参数)
arr       : int[10,10]         (参数)
```

### 6. `foo@code`（foo 函数体代码）

```
t1 = x == 0
if t1 == 0 goto L1
    t2 = arr[0][0]
    t3 = CALL boo(t2)
    t4 = CALL sqrt(t3)
    z = t4
    goto L2
L1:
    t5 = arr[x,x]
    t6 = CALL boo(t5)
    return t6
L2:
```