第五章

P155: 3 如下非分程序结构语言的程序段，画出编译该程序段时将生成的有序符号表。

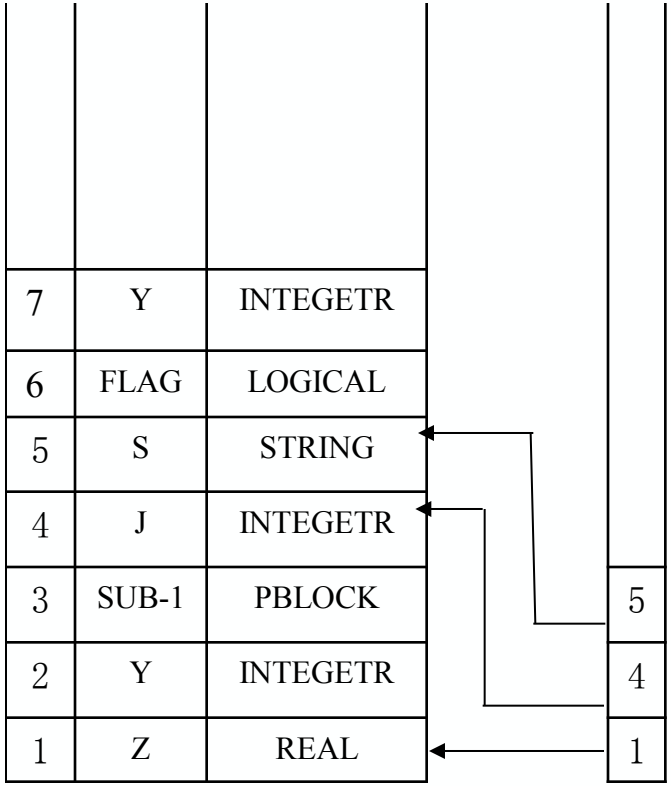
```
BLOCK
REAL X,Y,Z1,Z2,Z3;
INTEGER I,J,K,LASTI;
STRING LIST-OF-NAMES;
LOGICAL ENTRY-ON EXIT-OFF;
ARRAY REAL VAL(20);
ARRAY INTEGER MIN-VAL-IND(20);
  …
END OF BLOCK;
```

变量名	类型	维数
ENTRY-ON	LOGICAL	0
EXIT-OFF	LOGICAL	0
I	INTEGER	0
J	INTEGER	0
K	INTEGER	0
LASTI	INTEGER	0
LIST-OF-NAMES	STRING	0
MIN-VAL-IND	INTEGER	1
VAL	REAL	1
X	REAL	0
Y	REAL	0
Z1	REAL	0
Z2	REAL	0
Z3	REAL	0

写错了MIN-VAL-IND和VAL的类型或是维数

P185: 5 画出下面的分程序结构的程序段当程序段3和4编译即将完成以前，栈式符号表的图形(包括有效部分和失效部分）。

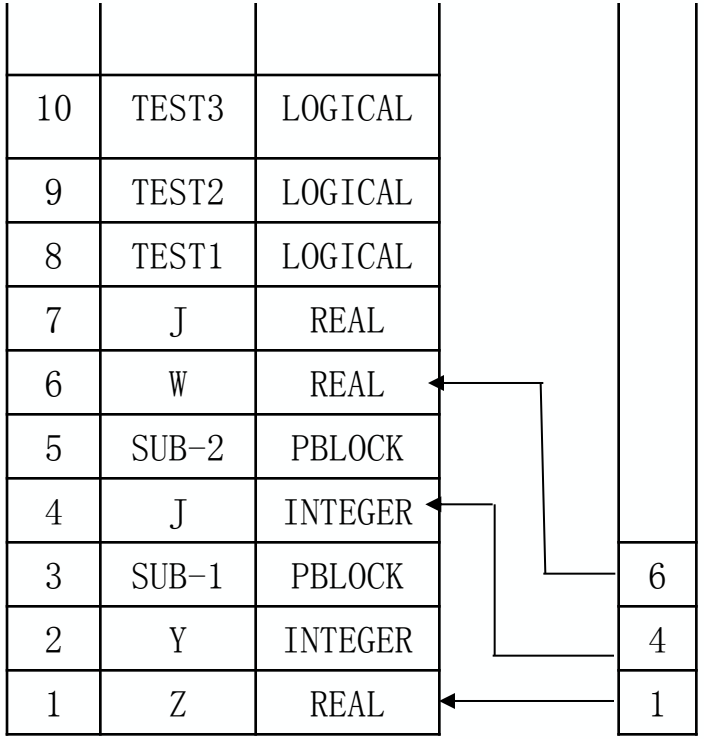
```
BBLOCK;
    REAL  Z; INTEGER  Y;
    SUB-1  PBLOCK(INTEGER  J);
        …
      BBLOCK;
         ARRAY  STRING  S(J+2);
          LOGICAL  FLAG;  INTEGER  Y;
            …
       END;
       SUB-2  PBLOCK(REAL  W);
            REAL  J; LOGICAL  TEST1,TEST2,TEST3;
            …
       END SUB-2;
    END SUB-1;
END;
```



```
BBLOCK;
            REAL Z; INTEGER Y;
            SUB-1 PBLOCK（INTEGER J）;
        3       BBLOCK;
        3         ARRAY STRING S（J+2）;
        3         LOGICAL FLAG; INTEGER Y;
        3       END;
        4       SUB-2 PBLOCK（REAL W）;
        4          REAL J; LOGICAL TEST1，TEST2
，TEST3;
        4       END SUB-2;
            END SUB-1;
          END;
```

- 在4编译即将完成以前的符号表中把J算作了失效
- 对有些符号表项失效后的符号表没有重新编号
- 没有把SUB-1、SUB-2加入符号表
- 没有画分程序索引表

```
BBLOCK;
    REAL  Z; INTEGER  Y;
    SUB-1  PBLOCK(INTEGER  J);
        …
      BBLOCK;
         ARRAY  STRING  S(J+2);
          LOGICAL  FLAG;  INTEGER  Y;
            …
       END;
       SUB-2  PBLOCK(REAL  W);
            REAL  J; LOGICAL  TEST1,TEST2,TEST3;
            …
       END SUB-2;
    END SUB-1;
END;
```



P166:2 考虑下面的类ALGOL程序，画出当程序执行到①和②时，运行栈内容的图像。

常见的错误

I 在画程序运行②时，没有画出局部数据区的R;

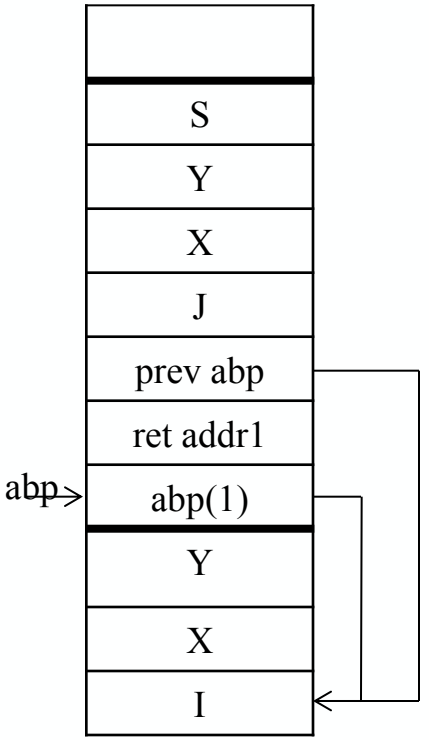
II 活动记录间应该用双线条间隔；

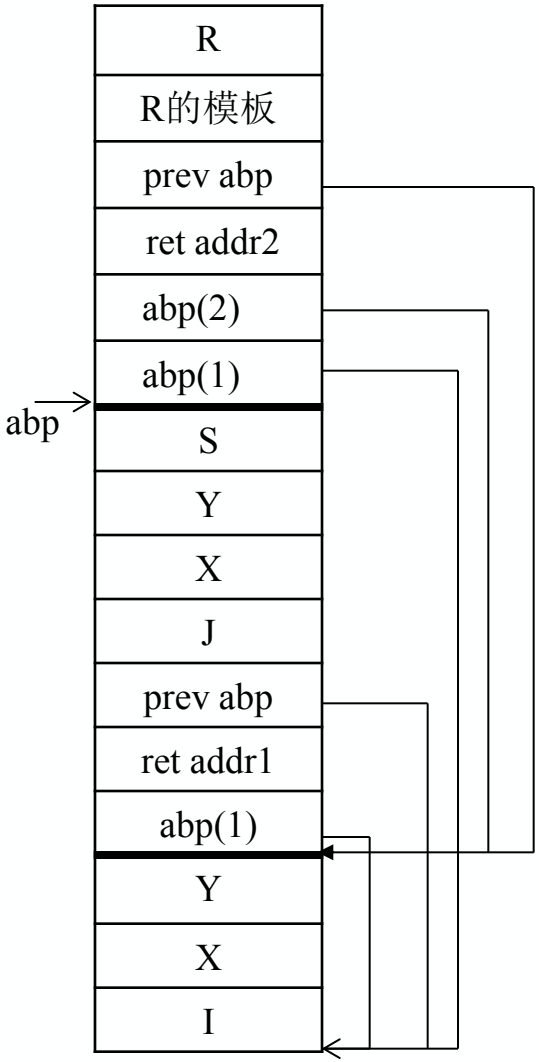
III 很多同学在画指针的指向时，把ret addr作为指针的引出地点

仅将栈画出，但没有用箭头指示层与层之间的关系，例如prev abp指向哪个AR,

有同学将ret addr指向前一层的AR区

有的同学对栈中各字段含义不是很清楚





P166: 6.

①做题情况

该题做的不好，大多数同学计算R(1,1)的地址没有考虑模板的情况因此得出的答案是30。

另外，Y的地址没有说清楚，有两种答案

②参考答案

Addr(R(1,1))=abp+(BL-1)+nip+ON

=26+2+3+0=31

Addr(I)=display[BL]+(BL-1)+nip+ON

=5+0+0+0=5

Addr(Y)=display[BL]+(BL-1)+nip+ON

=12+1+2+2=17

Addr(Y)=display[BL]+(BL-1)+nip+ON

=5+0+0+2=7

P175:2 将下面的语句 A:= (B+C) ↑E+(B+C)*F 转换成三元式，间接三元式和四元式序列。

三元式

(1) + B, C
(2) ↑ (1),E
(3) + B, C
(4) * (3),F
(5) + (2),(4)
(6) := A, (5)

四元式

(1) + B, C, T1
(2) ↑ T1, E, T2
(3) + B, C T3
(4) * T3, F, T4
(5) + T2, T4, T5
(6) := A, T5

间接三元式

操作
1、 (1)
2、 (2)
3、 (1)
4、 (3)
5、 (4)
6、 (5)

(1) + B, C
(2) ↑ (1), E
(3) * (1), F
(4) + (2), (3)
(5) := A, (4)

1 大部分同学忽略 :=的表示
有同学对间接三元式的概念不是很清楚，混淆了三元式和间接三元式

教材第九章

P193: 1.试分别构造一个符号串翻译文法，它将由一般中缀表达式文法所定义的中缀表达式翻译成波兰前缀表达式和波兰后缀表达式。

解：翻译为波兰前缀表达式的文法为：

E->@+E+T
E->T
T->@*T*F
T->F
F->(E)
F->@ii

翻译为波兰后缀表达式的文法为：

E->E+T@+
E->T
T->T*F@*
T->F
F->(E)
F->i@i

P193: 2. 构造一符号串翻译文法，它将接受由0和1组成的任意输入符号串，并产生下面的输出符号串：
(a) 输入符号串倒置 (c) 输入符号串本身。

（a）S->0S@0　　或　S->@0S0
　　S->1S@1　　　　S->@1S1
　　S->ε　　　　　S->ε

（c）E->0@0E
　　E->1@1E
　　E->ε

P193: 3. 以下的符号串翻译文法能做什么？

<s>->@C EN @HI GL @N I @E S @SE H

解：可以识别终结符号串ENGLISH，并输出符号串CHINESE。

4. 有特殊的翻译文法产生的两个活动序列是：

@x@yb@z和@qa@x@yb@z@x@x@yb@z@y

由这个翻译文法删掉诸动作符号得到的输入文法是：

<S>->a<S><S>

<S>->b

这是个什么翻译文法？

解：其文法为：

<S>->@x@yb@z

<S>->@qa<S>@x<S>@y

5.下面给出带有开始符号<S>的翻译文法，试列出属于这个文法所定义的语法制导翻译的所有对偶。

<S>-><A>xc<B>@y
<S>->@yd@xc@zb
<A>-><B>a@y
<A>->d
<B>->b@x

①做题情况
该题做的不错，不过有些同学的题目抄错了，
<s>-><A>@xc<B>@y，所以答案就写错了；还有相当的同学在写对偶时，没有把符号“@”写上

解：
(1)dcb　　@y@x@z
(2)dxcb　　@x@y
(3)baxcb　　@x@y@x@y

构造将算术表达式翻译成四元式的属性翻译文法，并写出递归下降分析程序。

1. $E_{\uparrow e} \rightarrow T_{\uparrow t1}\ \{+T_{\uparrow t2}\ @ADD\downarrow_{t1,t2,e}\}$
2. $T_{\uparrow t} \rightarrow F_{\uparrow f1}\{*F_{\uparrow f2}@MULT_{\downarrow f1,f2,t}\}$
3. $F_{\uparrow f} \rightarrow (E_{\uparrow f})$
4. $F_{\uparrow f} \rightarrow Id_{\uparrow f}$

```
PROCEDURE ADD
t1,t2,e; /* 形式参数 */
  emit（+，t1,t2,e）;
end;
```

```
PROCEDURE MULT
f1,f2,t; /* 形式参数 */
  emit（*，f1,f2,t）;
end;
```

```
PROCEDURE F;
  if CLASS = 'i' then
      begin
          NEXTSYM;
          return (Entry(i));
      end;
else if CLASS = '('  then
          begin
              NEXTSYM;
              f =E;  /*调用过程E*/
              if CLASS = ')'then
                begin
                      NEXTSYM;
                      return f;
                end
              else
                 Error;
              end
 else
       Error;
end F;
```

1. $E_{\uparrow e} \rightarrow T_{\uparrow t1}\ \{+T_{\uparrow t2}\ @ADD\downarrow_{t1,t2,e}\}$
2. $T_{\uparrow t} \rightarrow F_{\uparrow f1}\{*F_{\uparrow f2}@MULT_{\downarrow f1,f2,t}\}$
3. $F_{\uparrow f} \rightarrow (E_{\uparrow f})$
4. $F_{\uparrow f} \rightarrow Id_{\uparrow f}$

```
PROCEDURE T;
 f1,f2;  /*局部变量*/
  t;
 f1 = F;  /*调用过程F*/
 while  CLASS = '*' do
     begin
              NEXTSYM;
              f2 = F;
              t = NewAddr;
            /*用一个新单元存放计算结果*/
              MULT(f1,f2,t);
              f1 = t;
    end
    return (f1);
end T;
```

1. $E_{\uparrow e} \rightarrow T_{\uparrow t1}\ \{+T_{\uparrow t2}\ @ADD_{\downarrow t1,t2,e}\}$
2. $T_{\uparrow t} \rightarrow F_{\uparrow f1}\{*F_{\uparrow f2} @MULT_{\downarrow f1,f2,t}\}$
3. $F_{\uparrow f} \rightarrow (E_{\uparrow f})$
4. $F_{\uparrow f} \rightarrow Id_{\uparrow f}$

```
PROCEDURE E;
  t1,t2;  /*局部变量*/
    e;
  t1 = T;  /* 调用过程T */
  while CLASS = '+' do
    begin
        NEXTSYM;
        t2 = T; /*调用过程T*/
        e = NewAddr;
          /*用一个新单元存放计算结果*/
        ADD(t1,t2,e);
        t1 = e;
   end
  return (t1);
end E;
```

Ch10: 1. 试设计Pascal记录变量声明（无变体）的属性翻译文法，并构造相应的语义动作程序。

PASCAL记录类型的例子：

```
TYPE person = RECORD
                name: ARRAY[1..8] OF char;
                  sex: boolean;
                age: integer
                  END;
```

输入文法：

<record del>→<entity> = RECORD <comp_list> END

<comp_list>→<comp_del><comp_dels>

<comp_dels>→;<comp_del><comp_dels> | ε

<comp_del>→<name> : <type>

属性翻译文法：

<record del>→<entity>↑n = RECORD↑k@init↑m,s,a <comp_list>↓a,m,s↑m',s' END
@insertsym↓n,k,m',s'
<comp_list>↓a,m,s↑m',s'→<comp_del>↓a↑L,a' @compinfo↓m,L↑m',s'
<comp_dels>↓a',m',s'↑m'',s''
<comp_dels>↓a,m,s↑m',s'→;<comp_del>↓a↑L,a'@compinfo↓m,L↑m',s'<comp_dels>
| ε
<comp_del>↓a↑L,a'→<name>↑c_n : <type>↑t,L@insertcomp↓a,c_n, t, L↑a'

n: 记录名　k：记录标识　m：分量个数　a:记录分量的符号表入口地址
s：各分量大小之和　l：分量大小　t：分量类型

```
@init:        /*初始化*/
procedure init;
  m: = 0;  /* 分量计数器清0 */
  s:= 0;   /* 分量大小计数器清0 */
  a:= NewEntry;  /*可用表项的入口地址*/
end;
```

```
@insertsym↓n,k,m,s
 /*把记录类型名、记录标识、分量个数和
          记录所需空间大小填入符号表*/
procedure insertsym(n,k,m,s)
```

属性翻译文法：

<record del>→<entity>↑n = RECORD↑k@init↑m,s,a <comp_list>↓a,m,s↑m',s' END
@insertsym↓n,k,m',s'

<comp_list>↓a,m,s↑m',s'→<comp_del>↓a↑L,a' @compinfo↓m,L↑m',s'
<comp_dels>↓a',m',s'↑m'',s''

<comp_dels>↓a,m,s↑m',s'→;<comp_del>↓a↑L,a'@compinfo↓m,L↑m',s'<comp_dels>
| ε

<comp_del>↓a↑L,a'→<name>↑c_n : <type>↑t,L@insertcomp↓a,c_n, t, L↑a'

n: 记录名　k：记录标识　m：分量个数　a:记录分量的符号表入口地址
s：各分量大小之和　l：分量大小　t：分量类型

```
@compinfo↓m,L↑m',s'
procedure compinfo(m,L)
  m:= m+1;  /*分量个数＋1*/
  s := s + L;  /* 分量大小被统计 */
  return (m, s );
end
```

```
@insertcomp↓a,c_n, t, L↑a'
procedure insertcomp(a,c_n, t, L )
  在a所指示的符号表项填入分量的名字、类型和大小
  a = a ＋ 1;
  return a;   //将下一个可用表项的入口返回
end;
```

2. 写出for语句在执行循环体之前先做循环条件测试的属性翻译文法及其处理动作程序。

1.<for loop>→<for head>$_{\uparrow a,\ f,\ r}$ < rest of loop>$_{\downarrow a,\ f,\ r}$

2.<for head>$_{\uparrow a,\ f,\ r}$ →for <id>$_{\uparrow a}$:= <expr>@initload
to <expr> @store$_{\uparrow e2}$@loadid$_{\downarrow a}$ @genjmp$_{\uparrow s}$ by @labgen$_{\uparrow r}$
@loadid$_{\downarrow e2}$ @loadid$_{\downarrow a}$<expr>@compare$_{\downarrow a,s\uparrow f}$

3.<rest of loop>$_{\downarrow a,\ f,\ r}$ →do <stat list> end for @retbranch$_{\downarrow r}$@labemit$_{\downarrow f}$

```
LDA, (<id>)
LOD, <expr1>
STN    ---@ initload
LOD, <expr2>
STO, E2---@store
LOD, <id>--@loadid
JMP test---@genjmp↑s   s:test
loop:  ----labgen↑r   r:loop
LOD, E2
LOD, <id>
LOD, <expr3>
ADD
STO, <id>
test:      //compare略有改动，先设置s,再生成f-end_loop,并生成BGT
BGT, end_loop
<statement>
…
JMP, loop
end_loop
```

作业： 作常数合并优化的表达式属性翻译文法及语义动作程序。

例： A := 2+3+C

A := 5+C

1) <expression>→ <expr> @opend

2) <expr>→ <term> <terms>

3) <terms>→ε

4) | + <term> @add <terms>

5) | — <term> @sub <terms>

6) <term>→<factor><factors>

7) <factors>→ ε

8) | * <factor> @mul <factors>

9) | / <factor> @div <factors>

10) <factor>→ <variable> $\uparrow_n$ @lookup $\downarrow_n \uparrow_j$
@enters1 $\downarrow_j$
| <integer> $\uparrow_i$ @enters2$\downarrow_i$
| (<expr>)

设栈S

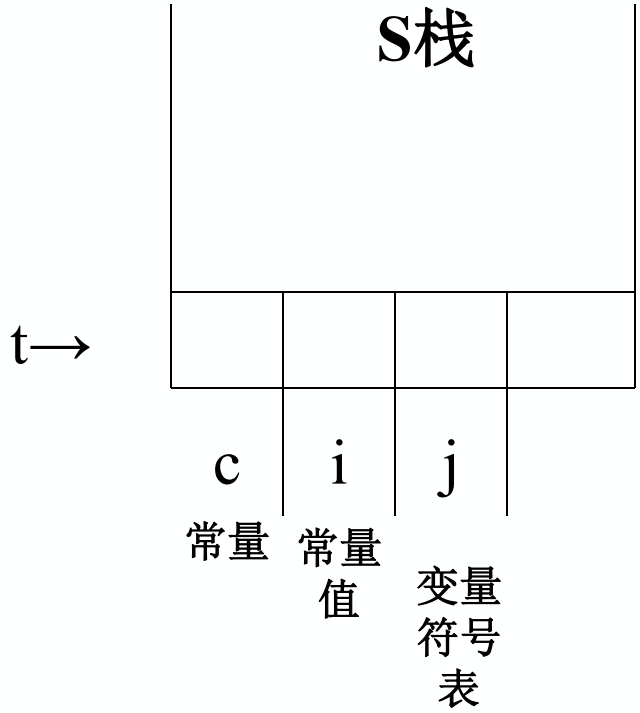
```
S: ARRAY[1…L]  OF   RECORD
                        c：  logical          /*常量标志*/
                        i:   integer          /*常数值*/
                        j:    integer         /*变量符号表地址*/
t：  integer                  /*s域指针*/
```

@enter1 ↓j　　/*变量入栈，j为符号表地址*/

```
procedure  enters1(j)
          j : integer;
      begin
          t := t+1;
         s(t).c := false;
         s(t).j = j;
       end;
```

@enter2 ↓i /*常量入栈，i为常量值 */

```
procedure   enters2(i);
        i : integer;
begin
        t := t+1;
       s(t).c := true;
        s(t).i := i;
          …
end;
```



@add

```
procedure   add
  if   s(t-1).c  and  s(t).c          then
       t := t - 1;
       s(t).i  := s(t).i + s(t+1).i;
       s(t).c := true;
   else  if   not(s(t-1).c)  and  not(s(t).c)  then
              emitl ( “LOD”, symbtbl( s(t-1).j ).addr );
              emitl ( “LOD”, symbtbl( s(t).j ).addr );
              emit ( “ADD” );
               t := t - 2;
```

```
else if  not(s(t-1).c) and s(t).c then
            emitl ( “LOD”, symbtbl(s(t-1).j).addr );
            emitl ( “LDC”, s(t).i );
            emit ( “ADD” );
             t := t-2;
else if  not(s(t).c) and s(t-1).c  then
            emitl ( “LDC”, (s(t-1).i );
            emitl ( “LOD” ,symbtbl (s(t).j).addr );
            emit ( “ADD” );
             t := t-2;
 end   add;
```

同理可编的下列动作符号的语义程序：

@sub

@mul

@div

@opend

```
procedure  opend;
    if    t > 0    then
        if    s(t).c    then
            emitl  ( “LDC”, s(t).i )
        else
            emitl ( “LOD”, symbtbl(s(t).j).addr );
end    opend;
```

- P87习题
  a) 对LL(1), SLR(1),算符优先文法的定义理解不清。很多同学做题的时候根本不考虑文法是否满足这几种文法（LL(1), SLR(1),算符优先文法）的要求，以及是否需要改写文法，便开始做题了。
  b) 几种文法（如LL(1), SLR(1),算符优先） 的判断应该归纳一下，有些人在判断时条件不充分就得出结果。
  c) 对于构造LL(1)分析表的步骤很模糊，很多同学写出来的分析表不够规范。

- P100、P108、P115习题
  a) 短语、素短语概念不清。
  b) 部分同学不会设计SLR(1)分析器。
  c) 活前缀概念不清，很多同学不能正确求出活前缀的有效项目集。

- P138习题
  a) 第2题很多同学没做对。

  p185习题
  第六题：很多同学不清楚栈式符号表的由那些部分组成。

代码生成

- 微处理器体系结构基础知识
- 基本块和流图的划分与建立方法
- 全局寄存器分配算法
  - 引用计数
  - 图着色
- 临时寄存器池管理方法与指令选择

代码优化

- DAG图
  - 消除局部公共子表达式
  - 从DAG图导出中间代码的启发式算法
- 数据流分析
  - 到达定义分析
  - 活跃变量分析
- 构建冲突图
  - 变量冲突的基本概念
  - 通过活跃变量分析构建（精度不太高的）冲突图

选做题

- 面向对象语言的编译技术
- 采用面向对象技术设计编译系统

- 提交ppt, 有2分奖励,最高5分
- 推荐交流的同学,5分,最高可到10分.

- 平时成绩最高 40分.