作业5

1、假设文法G[E]为基础文法的翻译模式如下，写出相应的递归下降翻译程序（可直接使用MatchToken函数）

E->T {R.in := T.val} R {E.val := R.val}

R->+T {R_1.in := R.in + T.val} R_1 {R.val := R_1.val}

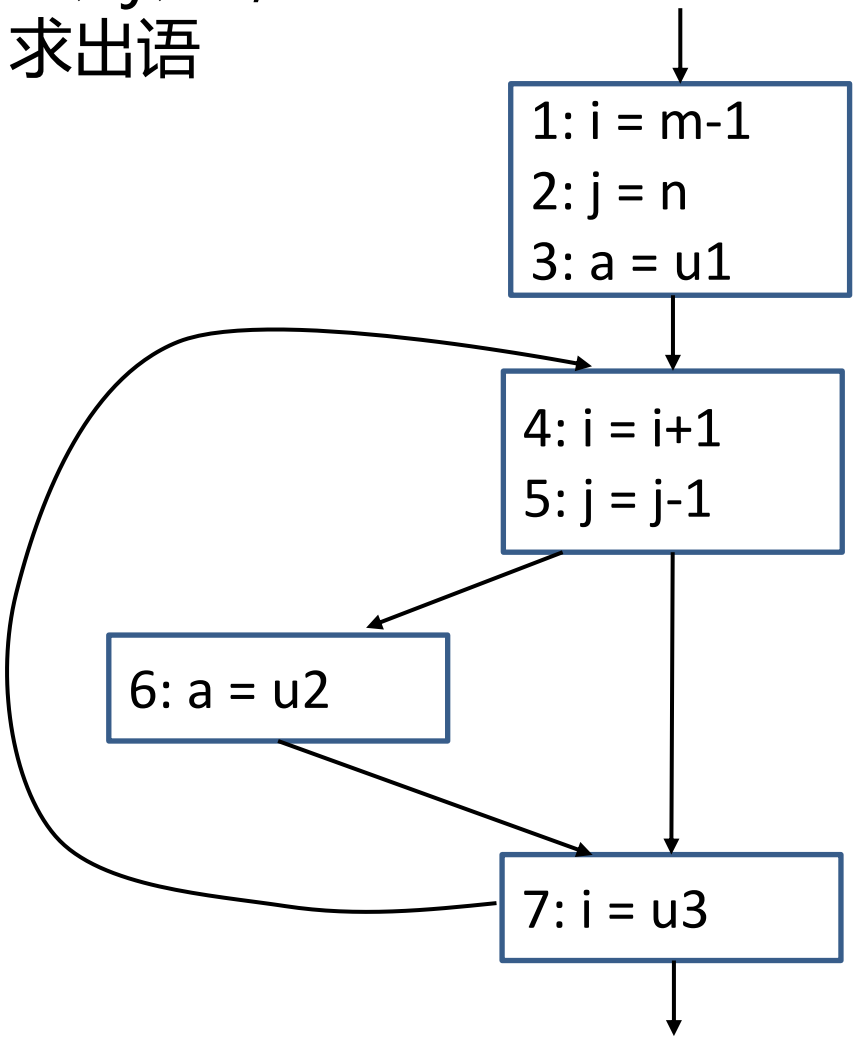
R->-T {R_1.in := R.in - T.val} R_1 {R.val := R_1.val}

R->ε {R.val := R.in}

T->n {T.val := lexval(n)}

作业5

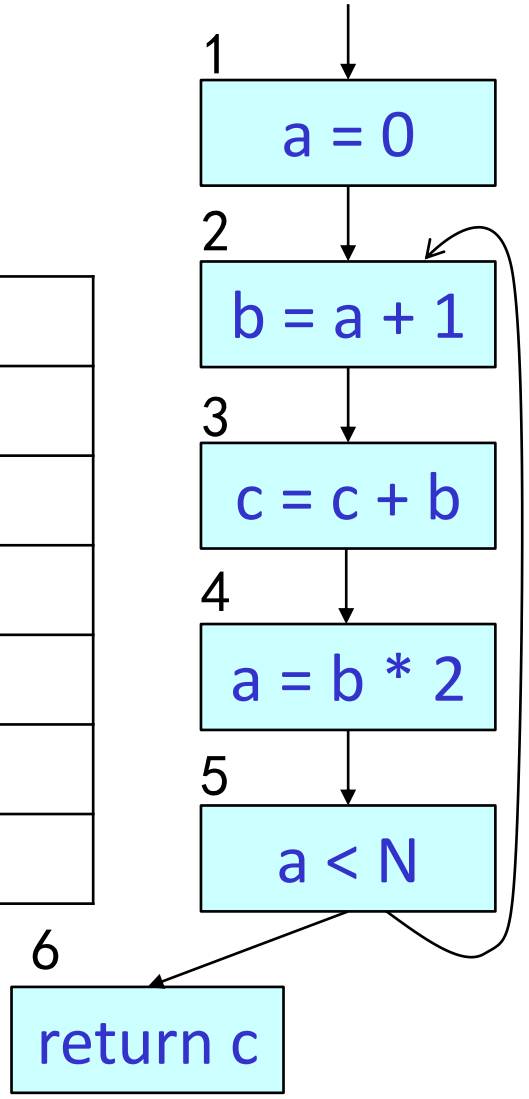
2、在下图中考虑变量i、j、a，进行到达定义分析，求出语句的in和out。



作业5

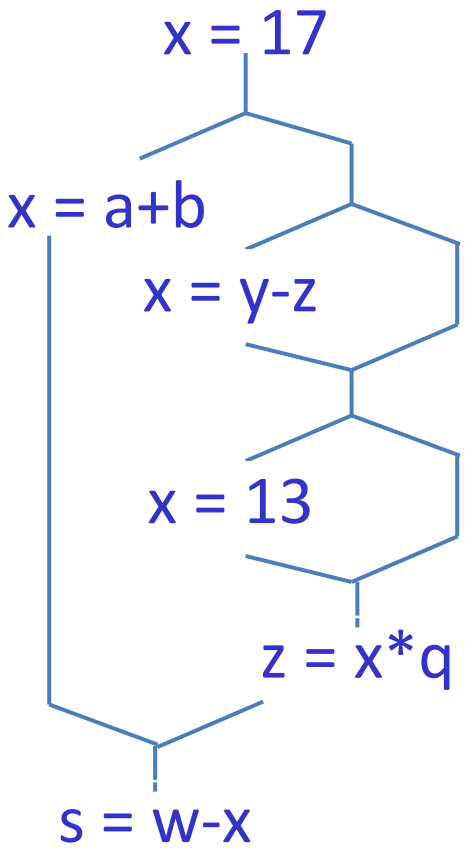
3、对下图进行活性分析，按照逆序即语句按照6、5、4、3、2、1的顺序进行计算，填写下表。

	in/out	in/out	…
1	{ } { }		
2	{ } { }		
3	{ } { }		
4	{ } { }		
5	{ } { }		
6	{ } { }		



作业5

4、写出下面原始代码片段转换出的SSA形式，下面片段主要显示了x的定义使用，省略了其他语句。



作业5

5、两个问题：

(1)请将下图的三地址代码序列划分为基本块并做出其流图。

(2)找出流图中的循环。

(1) i:=m-1
(2) j:= n
(3) t1:=4*n
(4) v:=a[t1]
(5) i:=i+1
(6) t2:=4*i
(7) t3:=a[t2]
(8) if t3 < v goto(5)
(9) j:=j-1
(10) t4:=4*j
(11) t5:=a[t4]
(12) if t5<v goto(9)
(13) if i>=j goto(23)
(14) t6:=4*i
(15) x:=a[t6]

(16) t7:=4*i
(17) t8:=4*j
(18) t9:=a[t8]
(19) a[t7]:=t9
(20) t10:=4*j
(21) a[t10]:=x
(22) goto(5)
(23) t11:=4*i
(24) x:=a[t11]
(25) t12:=4*i
(26) t13:=4*n
(27) t14:=a[t13]
(28) a[t12]:=t14
(29) t15:=4*n
(30) a[t15]:=x

1.

	E	R	T	n
继承属性	\	in	\	\
综合属性	val	val	val	lexval(n)

First_S(E) = {n}
First_S(T) = {n}
First_S(R) = {+, -, #}

翻译程序:

```
int ParseE(){
  if (lookahead == n){
    Tval = ParseT();
    Rin = Tval;
    Rval = ParseR(Rin);
    Eval = Rval;
    return Eval;
  }
  printf("syntax error\n"); exit(0);
}
```

```
int ParseT(){
  if (lookahead == n){
    MatchToken(n);
    val = lexval(n);
    return val;
  }
  printf("syntax error\n"); exit(0);
}
```

```
int ParseR(int in){
  if (lookahead == '+'){
    MatchToken('+');
    Tval = Parse(T);
    Rin = in + Tval;
    Rval = ParseR(Rin);
    val = Rval;
  } else if (lookahead == '-'){
    MatchToken('-');
    Tval = ParseT();
    Rin = in - Tval;
    Rval = ParseR(Rin);
    val = Rval;
  } else if (lookahead == '#'){
    val = in;
  } else {printf("syntax error\n"); exit(0); }
  return val;
}
```

2.

	1	2	3	4	5	6	7
gen	{1}	{2}	{3}	{4}	{5}	{6}	{7}
kill	{4,7}	{5}	{6}	{1,7}	{2}	{3}	{1,4}

语句 \ 迭代	in/out	in/out
1	{} {1}	{} {1}
2	{1} {1,2}	{1} {1,2}
3	{1,2} {1,2,3}	{1,2} {1,2,3}
4	{1,2,3} {2,3,4}	{1,2,3,5,6,7} {2,3,4,5,6}
5	{2,3,4} {3,4,5}	{2,3,4,5,6} {3,4,5,6}
6	{3,4,5} {4,5,6}	{3,4,5,6} {4,5,6}
7	{3,4,5,6} {3,5,6,7}	{3,4,5,6} {3,5,6,7} .

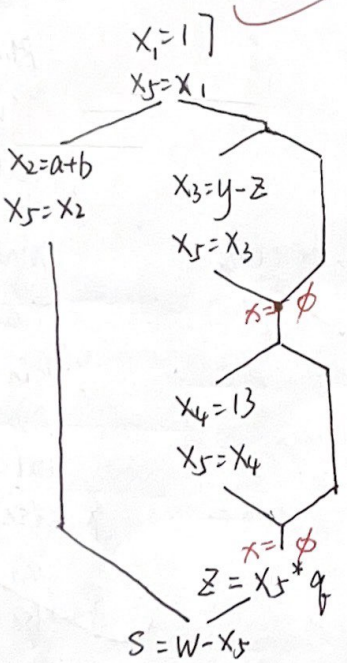
3.

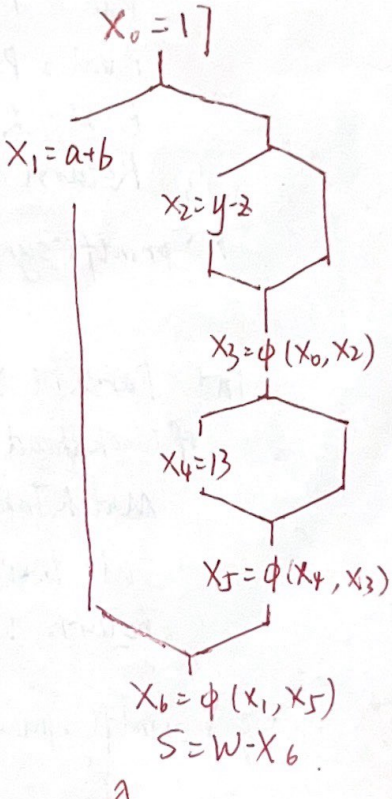
	1	2	3	4	5	6
use	{}	{a}	{b,c}	{b}	{a,N}	{c}
def	{a}	{b}	{c}	{a}	{}	{}

逆序计算:

	in/out	in/out	in/out
1	{}{}	{c}{a,c}	{c}{a,c}
2	{}{}	{a,c}{b,c}	{a,c}{b,c}
3	{}{}	{b,c}{b,c}	{b,c}{b,c}
4	{}{}	{b,c}{a,c}	{b,c}{a,c}
5	{}{}	{a,c}{c}	{a,c}{a,c}
6	{}{}	{c}{}	{c}{}

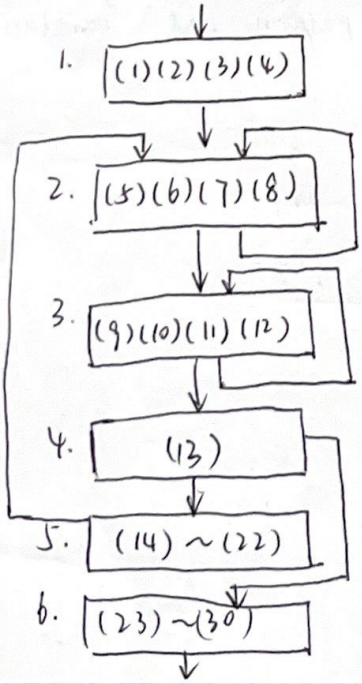
4.





A

5.



支配结点集:

D(1) = {1}
D(2) = {1,2}
D(3) = {1,2,3}
D(4) = {1,2,3,4}
D(5) = {1,2,3,4,5}
D(6) = {1,2,3,4,6}

回边: 循环:

回边	循环
2→2	{2}
3→3	{3}
5→2	{2,3,4,5}