1. 设有SDD如下：

|  |  |
| --- | --- |
| 产 生 式 | 语 义 规 则 |
| S → B#  B → B 0  B → B 1  B → 1 | S .val=B.val  B .val =2×B1 .val  B .val =2×B1 .val+1  B .val =1 |

给出101101的注释语法分析树。

1. 含小数点的二进制数文法定义如下：

S→L.L|L

L→LB|B

B→0|1

(1)根据上述文法设计一个SDD，实现二进制数到十进制数的翻译，比如101.101翻译为5.625。

(2)给出101.101翻译为5.625的注释语法分析树。

1. （选做）在保证语义相同的情况下，改写1题中的SDD，使文法不是左递归的。
2. 设包含有数组引用的复制表达式的SDD如下：

|  |  |
| --- | --- |
| 产 生 式 | 语 义 规 则 |
| S → i=E;  S → L=E;  E → E+E  E → i  E → L  L → i[E]  L → L[E] | gen(= E.place - i)  gen([]= E.place - L.base[L.addr])  E.place=newtemp();gen(+ E1 .place E1 .place E.place)  E.place = i  E.place=newtemp();gen(=[] L.base[l.addr] - E.place)  L.base=i;L.addr=newtemp();gen(\* E.place L.elem.width L.addr)  L.base=L1.base;L.type=L1.type;t=newtemp();L.addr=newtemp();gen(\* E.place L.elem.width t);  gen(+ L1.addr t L.addr) |

给出下列赋值语句的四元式代码：

1. x=a[i][j]+b[i][j]
2. a[b[i][j]][c[k]]=a[i][j]

**说明：**

1. gen()是产生四元式代码的函数，参数依次为操作符、第一操作数、第二操作数、计算结果;
2. newtemp()是临时变量申请函数；
3. 关于数组中用到的属性width的SDD如下：

|  |  |
| --- | --- |
| 产 生 式 | 语 义 规 则 |
| D → B iC  B → int  B → float  C → ε  C → [num]C | w=B .width；i.width=C.width  B .width =4  B .width =8  C.width=w  C.width=num.value×C1.width;C.elem.width=C1.width |

1. 语句中的数组说明：int a[10][20],b[10][20];int c[10]。
2. 设有布尔表达式以及条件语句的SDD如下：

|  |  |
| --- | --- |
| 产 生 式 | 语 义 规 则 |
| B→E  B→E= =E  B→!B  B→B||MB  B→B&&MB  S→if(B)MSNelseMS  S→whileM (B) MS  M→ε  N→ε  S→{L}  S→i=E；  L→LMS  L→S | B.true=makelist(nextinstr); B.false=makelist(nextinstr+1); gen(JT E.PLACE - 0); gen(JF E.PLACE - 0)  B.PLACE=NEWTEMP； gen(= = E1.PLACE E2.PLACE B.PLACE); B.true=makelist(nextinstr)  B.false=makelist(nextinstr+1); gen(JT B.PLACE - 0); gen(JF B.PLACE - 0)  B.true=B1.false; B.false=B1.true  backpatch(B1.false,M.instr); B.true=merge(B1.true,B2.true); B.false=B2.false  backpatch(B1.true,M.instr); B.true=B2.true; B.false=merge(B1.false,B2.false)  backpatch(B.true,M1.instr); backpatch(B.false,M2.instr); S.next=merge(S1.next,N.next,S2.next)  backpatch(S1.next,M1.instr); backpatch(B.true,M2.instr); S.next=B.false;gen(J - - M1.instr)  M.instr=nextinstr  N.next=makelist(nextinstr); gen(J - -0)  S.next=L.next  S.next=null  {backpatch(L1.next, M.instr); L.next=S.next;}  L.next=S.next |

**SDD的说明：**

a)四元式编号作为跳转地址使用

b)函数makelist(x)表示出现了标号的引用时不知道地址的情况，函数参数x为标号的使用位置，单遍处理生成中间代码需要使用拉链—返填技术，假设通过标号表记录相关信息，此处为标号的引用出现，按标号引用记录入标号表。

c)函数backpatch(x,y)表示标号x的定义位置在y,需要返填标号x的使用位置。

d)函数merge(x,y,z,….)表示标号x,y,z…合并为一个标号

e)nextinstr表示下一条四元式的编号

有程序段：

if(a==b&&(c==d||e==f))

{while(!x==y)

x=x+y;

s=x;}

else

s=y;

z=s;

问题：

(1)给出该程序段的语法分析树(表达式E结构使用4题的)。

(2)给出该程序段语义处理后的四元式形式的目标代码（代码中的表达式E结构使用4题的SDD）。

(3)给出该程序段语义处理后的拉链-返填使用的标号表。

标号表按以下示例内容填写 ：

|  |  |  |
| --- | --- | --- |
| 标号名 | 定义否（1/0） | 返填顺序 |
| Li | 1 | ⑤→②→① |
| **…** |  |  |