第五章作业第二次作业

5.3.1：下面是涉及运算符+和整数或浮点运算分量的表达式的文法。区分浮点数的方法是看它有无小数点。

E → E + T | T

T → num.num | num

1）给出一个 SDD 来确定每个项 T 和表达式 E 的类型

2）扩展 1）中得到的 SDD，使得它可以把表达式转换为后缀表达式。使用一个单目运算符 intToFloat 把一个整数转换为相等的浮点数

Answer

1) 令 type 为表达式的类型。

	产生式	语义规则
1)	E → E1 + T	E.type = (E1.type == integer && T.type == integer) ? integer : float
2)	E → T	E.type = T.type
3)	T → num.num	T.type = float
4)	T → num	T.type = integer

2) node 为综合属性; valToChar 把一个数字转换为字符串。

	产生式	语义规则
1)	S → E	print('E.node')

2)	E → E1 + T	if(E1.type == integer && T.type == integer)
		E.type = integer
		E.node = E1.node || T.node || '+'
		else
		E.type = float
		if(E1.type == integer)
		E1.type = float
		E1.val = intToFloat(E1.val)
		E1.node = valToChar(E1.val)
		if(T.type == integer)
		T.type = float
		T.val = inToFloat(T.val)
		T.node = valToChar(T.val)
		E.node = E1.node || T.node || '+'

3)	E → T	E.type = T.type E.val = T.val E.node = valToChar(E.val)
3)	T → num.num	E.type = float T.val = num.num.lexval T.node = valToChar(T.val)
4)	T → num	T.type = integer T.val = num.lexval T.node = valToChar(T.val)

5.4.2：改写下面的 SDT：

A → A { a } B | A B { b } | 0

B → B { c } A | B A { d } | 1

使得基础文法变成非左递归的。其中 a、b、c 和 d 是语义动作，0 和 1 是终结符号

Answer

A → 0 A’

A’ →{ a } B A’ | B { b } A’ | ε

B → 1 B’

B’ → { c } A B’ | A { d } B’ | ε

5.4.6：修改图 5-25 中的 SDD，使它包含一个综合属性 B.le，即一个 Box 的长度。两个 Box 并列后得到的 Box 的长度是这两个 Box 的长度和。然后，将你的新规则加入到图 5-26 中 SDT 的合适位置上

Answer

函数 getLe 表示获取文本的长度。

修改图 5-25 的 SDD：

	产生式	语义规则
1)	S → B	B.ps = 10
2)	B → B1 B2	B1.ps = B.ps B2.ps = B.ps B.ht = max(B1.ht, B2.ht) B.dp = max(B1.dp, B2.dp) B.le = B1.le + B2.le
3)	B → B1 sub B2	B1.ps = B.ps B2.ps = 0.7 * B.ps B.ht = max(B1.ht, B2.ht – 0.25 * B.ps) B.dp = max(B1.dp, B2.dp + 0.25 * B.ps) B.le = B1.le + 0.7 * B2.le
4)	B → (B1)	B1.ps = B.ps B.ht = B1.ht B.dp = B1.dp B.le = B1.le
5)	B → text	B.ht = getHt(B.ps, text.lexval) B.dp = getDp(B.ps, text.lexval) B.le = getLe(B.ps, text.lexval)

SDT 如下：

	产生式	语义动作
1)	S → B	{B.ps = 10;}
2)	B → B1	{B1.ps = B.ps;} {B2.ps = B.ps;}

	B2	{ B.ht = max(B1.ht, B2.ht); B.dp = max(B1.dp, B2.dp); B.le = B1.le + B2.le;}
3)	B → B1 sub B2	{B1.ps = B.ps;} { B2.ps = 0.7 * B.ps;} { B.ht = max(B1.ht, B2.ht – 0.25 * B.ps); B.dp = max(B1.dp, B2.dp + 0.25 * B.dp); B.le = B1.le + 0.7 * B2.le;}
4)	B → (B1)	{B1.ps = B.ps;} { B.ht = B1.ht; B.dp = B1.dp; B.le = B1.le;}
5)	B → text	{B.ht = getHt(B.ps, text.lexval); B.dp = getDp(B.ps, text.lexval); B.le = getLe(B.ps, text.lexval);}