UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
CIÊNCIA DA COMPUTAÇÃO
ELC 408 – Compiladores
Prof. Giovani Rubert Librelotto – Peso 8,0

Prova do Segundo Bimestre de Compiladores

Nome: *Data:*

1. (5,0 pontos) Dado o esquema de tradução a seguir, mostre a árvore de derivação decorada e as tabelas de símbolos que serão construídas para o reconhecimento do programa abaixo.

P	M D	{ defTam (top (tabPtr), top (desloc)); pop (tabPtr); pop (desloc) }
M	ε	{ t = geraTab (nil); push (t, tabPtr); push (0, desloc) }
D	D ; D	
D	id : T	{ adSimb (top (tabPtr), id.nome, T.tipo, top (desloc)); top (desloc) = top (desloc) + T.tam }
D	proc id ; N D ;	{ t = top (tabPtr); defTam (t, top (desloc)); pop (tabPtr); pop (desloc) ; adProc (top (tabPtr), id.nome, t) }
N	ε	{ t = geraTab (top (tabPtr)); push (t, tabPtr); push (0, desloc) }
T	int	{ T.tipo = int; T.tam = 4 }
T	real	{ T.tipo = real; T.tam = 8 }
T	array [num] of T1	{ T.tipo = matriz (num.val, T1.tipo); T.tam = num.val * T1.tam }
T	^ T1	{ T.tipo = ponteiro (T1.tipo); T.tam = 4 }

```
a : array [3] of int;
c : real;
proc P2;
        b : array [4] of real;
        e : int;
proc P3;
        f : real;
        d : array [2] of real;
        proc P1;
                g : array [5] of real;
                h : real;
```

2. (5,0 pontos) A partir do esquema de tradução abaixo, gerar a árvore de derivação decorada e o código intermediário gerado para a palavra: A < D and (B > E or C > F)

E	E_1 or E_2	{ E.nome = geratemp; geracod(E.nome ":=" E_1.nome "or" E_2.nome) }
E	E1 and E_2	{ E.nome = geratemp; geracod(E.nome ":=" E_1.nome "and" E_2.nome) }
E	not E_1	{ E.nome = geratemp; geracod(E.nome ":=" "not" E_1.nome) }
E	(E_1)	{ E.nome = E_1.nome }
E	id_1 oprel id_2	{ E.nome = geratemp; geracod("if" id_1.nome oprel.simb id_2.nome "goto" proxq + 3) geracod(E.nome ":= 0"); geracod("goto" proxq + 2); geracod(E.nome ":= 1") }
E	true	{ E.nome = geratemp; geracod(E.nome ":= 1") }
E	false	{ E.nome = geratemp; geracod(E.nome ":= 0") }