

## Lista de Exercícios #12

### Geração de TAC, Checagem de Tipos, Expressões Booleanas, Controle de Fluxo

- Determine os endereços relativos para os identificadores seguintes considerando o esquema de tradução abaixo:
  - `float x;`
  - `record { float x; float y; } p;`
  - `record { int tag; float x; float y; } q;`

$P \rightarrow MD$   
 $M \rightarrow \epsilon \{ \text{offset} = 0; \}$   
 $D \rightarrow T \text{ id; } \{ \text{top.put(id.lexeme, T.tipo, offset); offset} += T.\text{width}; \} D_1$   
 $D \rightarrow \epsilon$   
 $T \rightarrow \text{record } \{ ' X D ' \} Y$   
 $X \rightarrow \epsilon \{ \text{Env.push(top); top} = \text{new Env()}; \text{Stack.push(offset); offset} = 0; \}$   
 $Y \rightarrow \epsilon \{ T.\text{tipo} = \text{record(top)}; T.\text{width} = \text{offset}; \text{top} = \text{Env.pop()}; \text{offset} = \text{Stack.pop()}; \}$
- Utilize o esquema de tradução abaixo para construir diferentes tipos válidos:
 

$T \rightarrow B \{ t = B.\text{type}; w = B.\text{width}; \} C$   
 $B \rightarrow \text{int } \{ B.\text{type} = \text{integer}; B.\text{width} = 4; \}$   
 $B \rightarrow \text{float } \{ B.\text{type} = \text{float}; B.\text{width} = 8; \}$   
 $C \rightarrow \epsilon \{ C.\text{type} = t; C.\text{width} = w; \}$   
 $C \rightarrow [ \text{num} ] C_1 \{ \text{array}(\text{num.value}, C_1.\text{type}); C.\text{width} = \text{num.value} * C_1.\text{width}; \}$
- O que é avaliação em curto-circuito? Quais suas vantagens e desvantagens?
- Qual a diferença entre avaliação numérica e por controle considerando expressões booleanas?
- Considerando o esquema de tradução para avaliação numérica de expressões booleanas abaixo, gere código intermediário as seguintes expressões lógicas:
  - `a < b and not c > d`
  - `a < b or c < d and e < f`
  - `not a < b or not c > d and x < q`
  - `not (a < b or not c > d) and x < q`

$E \rightarrow E_1 \text{ or } E_2 \quad \{ E.\text{nome} = \text{temp()};$   
 $\qquad\qquad\qquad \text{gera}(E.\text{nome} = E_1.\text{nome} \text{ or } E_2.\text{nome} \}$   
 $E \rightarrow E_1 \text{ and } E_2 \quad \{ E.\text{nome} = \text{temp()};$   
 $\qquad\qquad\qquad \text{gera}(E.\text{nome} = E_1.\text{nome} \text{ and } E_2.\text{nome} \}$   
 $E \rightarrow \text{not } E_1 \quad \{ E.\text{nome} = \text{temp()};$   
 $\qquad\qquad\qquad \text{gera}(E.\text{nome} = \text{not } E_1.\text{nome} \}$   
 $E \rightarrow (E_1) \quad \{ E.\text{nome} = E_1.\text{nome} \}$   
 $E \rightarrow E_1 \text{ op } E_2 \quad \{ E.\text{nome} = \text{temp()};$   
 $\qquad\qquad\qquad \text{gera}(\text{if } E_1.\text{nome} \text{ op.simb } E_2.\text{nome} \text{ goto proxq+3});$   
 $\qquad\qquad\qquad \text{gera}(E.\text{nome} = 0);$   
 $\qquad\qquad\qquad \text{gera}(\text{goto proxq+2});$   
 $\qquad\qquad\qquad \text{gera}(E.\text{nome} = 1); \}$   
 $E \rightarrow \text{true} \quad \{ E.\text{nome} = \text{temp()};$   
 $\qquad\qquad\qquad \text{gera}(E.\text{nome} = 1); \}$   
 $E \rightarrow \text{false} \quad \{ E.\text{nome} = \text{temp()};$   
 $\qquad\qquad\qquad \text{gera}(E.\text{nome} = 0); \}$
- Avalie as expressões do exercício 5. considerando o esquema para avaliação por fluxo de controle abaixo.

```

B → { B1.t=B.t; B1.f=rot(); } B1 or { B2.t=B.t; B2.f=B.f; } B2
    { B.code=B1.code || label(B1.f) || B2.code }
B → { B1.t=rot(); B1.f=B.f; } B1 and { B2.t=B.t; B2.f=B.f; } B2
    { B.code=B1.code || label(B1.t) || B2.code }
B → not { B1.t=B.f; B1.f=B.t; } B1 { B.code=B1.code; }
B → (B1) { B.code=B1.code; B.t=B1.t; B.f=B1.f; }
B → true { B.code=gera(goto B.t); }
B → false { B.code=gera(goto B.f); }
B → E1 relop E2 { B.code=E1.code || E2.code ||
    gera(if E1.local relop.lexval E2.local goto B.t) ||
    gera(goto B.f); }

```

Considerando o esquema de tradução abaixo:

```

S → attr { S.code=gera(attr.lexval) || gera(goto S.next) }
S → if { B.t=rot(); B.f=S.next; }
    (B) { S1.next=S.next; }
    S1 { S.code=B.code || gera(B.t:) || S1.code }
S → if { B.t=rot(); B.f=rot(); }
    (B) { S1.next=S.next; }
    S1 else { S2.next=S.next; }
    S2 { S.code=B.code || gera(B.t:) || S1.code ||
    gera(B.f:); || S2.code }
S → while { B.f=S.next; B.t=rot(); }
    (B) { S.begin=rot(); S1.next=S.begin; }
    S1 { S.code=gera(S.begin:) || B.code ||
    gera(B.t:) || S1.code || gera(goto S.begin) }
B → { B1.t=B.t; B1.f=rot(); } B1 or { B2.t=B.t; B2.f=B.f; } B2
    { B.code=B1.code || label(B1.f) || B2.code }
B → { B1.t=rot(); B1.f=B.f; } B1 and { B2.t=B.t; B2.f=B.f; } B2
    { B.code=B1.code || label(B1.t) || B2.code }
B → not { B1.t=B.f; B1.f=B.t; } B1 { B.code=B1.code; }
B → (B1) { B.code=B1.code; B.t=B1.t; B.f=B1.f; }
B → true { B.code=gera(goto B.t); }
B → false { B.code=gera(goto B.f); }
B → E1 relop E2 { B.code=E1.code || E2.code ||
    gera(if E1.local relop.lexval E2.local goto B.t) ||
    gera(goto B.f); }

```

Gere o código TAC para os trechos de código seguintes:

1.     if (not (a < b or not c > d) and x < q) {  
        x = z;  
    }
2.     while (a < b && e != f) {  
        if (c < d){  
            x = y + z;  
        } else {  
            x = x - z;  
        }

```
    }  
}
```

```
3.    if (x > a){  
        x = a;  
    }else{  
        x = q;  
    }
```

Altere o esquema de tradução para fluxo de controle acima e adicione regras de tradução para construções do tipo:

```
1.    if (not (a < b or not c > d) and x < q) {  
        x = z;  
    }else if (x > b){  
        x = k;  
    }else if (a > e){  
        x = q;  
    }
```

```
2.    if (x > a){  
        x = a;  
    }else if (x > b){  
        x = k;  
    }else{  
        x = q;  
    }
```

```
3.    for (i = x; x < a; s = a){  
        if (x > a){  
            x = a;  
        }  
    }
```

```
4.    switch (a){  
        case d: x = c;  
        case b: x = b;  
        default: x = c;  
    }
```

Traduza para TAC o código acima utilizando o esquema de tradução modificado.