**Compiladores – 2016/2**
**Roteiro de Laboratório 8**

# 1 Objetivos

O objetivo deste laboratório é implementar a visualização de uma *parse tree* para TINY usando o `bison` e a ferramenta GraphViz.

# 2 Entrada e Saída do *Parser*

O seu programa de entrada é lido da entrada padrão (*stdin*), como abaixo:

```
$ ./parser < program.tny
```

Exemplo de um arquivo de entrada (`program.tny`):

```
{ Sample program
  in TINY language -
  adds one to the input
}
read x;
x := x + 1;
write x;
```

Executando `./parser < program.tny`, temos como resposta no terminal:

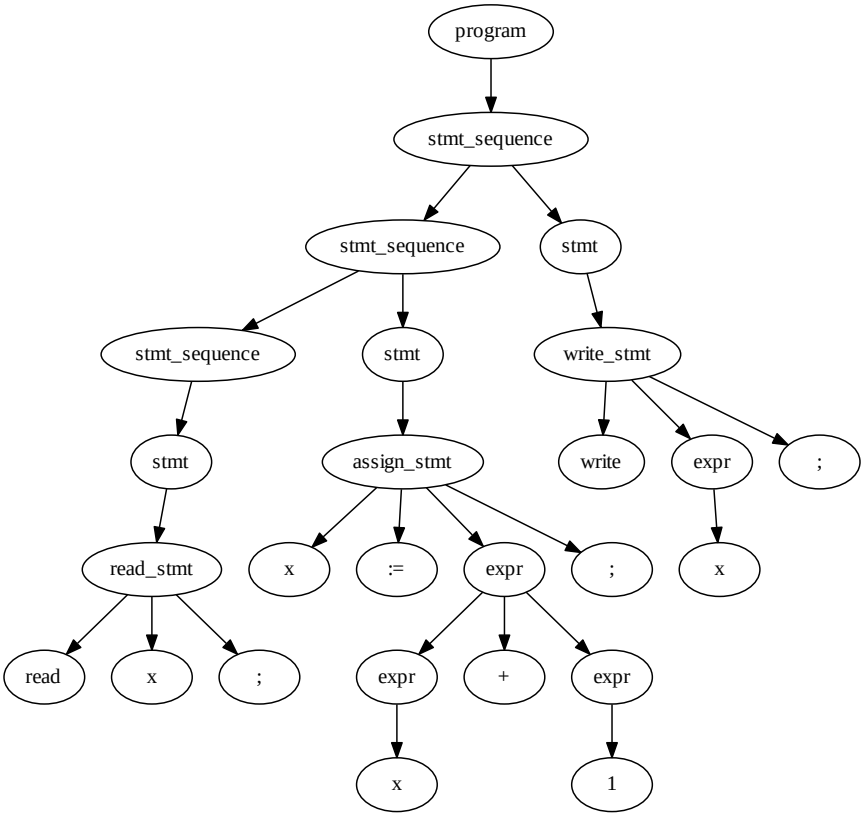
```
digraph {
graph [ordering="out"];
node0[label="program"];
node1[label="stmt_sequence"];
node2[label="stmt_sequence"];
node3[label="stmt_sequence"];
node4[label="stmt"];
node5[label="read_stmt"];
node6[label="read"];
node5 -> node6;
node7[label="x"];
node5 -> node7;
node8[label=";"];
node5 -> node8;
node4 -> node5;
node3 -> node4;
node2 -> node3;
node9[label="stmt"];
node10[label="assign_stmt"];
node11[label="x"];
node10 -> node11;
```

```
node12[label=":="];
node10 -> node12;
node13[label="expr"];
node14[label="expr"];
node15[label="x"];
node14 -> node15;
node13 -> node14;
node16[label="+"];
node13 -> node16;
node17[label="expr"];
node18[label="1"];
node17 -> node18;
node13 -> node17;
node10 -> node13;
node19[label=";"];
node10 -> node19;
node9 -> node10;
node2 -> node9;
node1 -> node2;
node20[label="stmt"];
node21[label="write_stmt"];
node22[label="write"];
node21 -> node22;
node23[label="expr"];
node24[label="x"];
node23 -> node24;
node21 -> node23;
node25[label=";"];
node21 -> node25;
node20 -> node21;
node1 -> node20;
node0 -> node1;
}
```

A saída exibida acima corresponde uma representação textual da árvore de derivação escrita na linguagem `.dot` da ferramenta GraphViz. Salvando a saída para um arquivo `tree.dot` e executando:

```
dot -Tpdf tree.dot -o tree.pdf
```

O resultado será um arquivo PDF como a figura a seguir:




## 3 Passos para realização do laboratório

1. Faça o *download* dos arquivos de exemplo no AVA. Compile-os e execute-os como explicado pelo professor no início da aula.
2. Estenda a implementação de árvore binária fornecida nos exemplos para uma árvore cujos os nós aceitam um número arbitrário de filhos. Você pode estabelecer um limite de filhos ou usar uma implementação encadeada, como preferir.
3. Utilize a implementação dessa nova árvore para construir a árvore de derivação (*parse tree*) do programa de entrada usando ações semânticas do `bison`.
4. Implemente um caminhamento na árvore que gera a saída do arquivo `.dot`.
5. Teste a sua implementação comparando com os resultados fornecidos pelo professor.

Observações importantes:

- O seu *parser* deve continuar exibindo as mesmas mensagens de erro do laboratório 07.
- Veja mais exemplos de entrada e saída no AVA.
- Uma implementação de referência para esse laboratório será disponibilizado pelo professor em um futuro próximo. No entanto, você é *fortemente* encorajado a realizar a sua implementação completa antes de ver uma solução em outro lugar.