Centro Federal de Educação Tecnológica de Minas Gerais

25 de Novembro de 2017

CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA – CEFET – MG

CURSO DE ENGENHARIA DE COMPUTAÇÃO

LAIS RIBEIRO MATOS LOUREIRO DE CARVALHO

LUCAS NASCIMENTO HUATI CORRÊA

MAURO FLORIANO DOS SANTOS

RELATÓRIO

ANALISADOR SEMÂNTICO

Professora: Kecia Aline Marques Ferreira

Compilador

Um compilador é formado por um ou mais programas que ao serem executados conseguem interpretar uma código em uma determinada linguagem textual em um código em linguagem de máquina. Pode ser dividido em três analisadores, (léxico, sintático e semântico) de forma a dividir as tarefas necessárias para a compilação.

A análise léxica é a primeira fase de um processo de compilação, em que é feita a leitura do programa fonte, caractere a caractere, que ao serem agrupados de forma correta de acordo com a gramática da linguagem geram um lexema, formando assim um token para cada lexema.

O analisador sintático já tem como tarefa principal determinar se o código fornecido, representado pelo fluxo de tokens, possui uma sentença válida para a gramática da linguagem sendo compilada, ou seja, se a ordem dos tokens presentes no código está de acordo com as regras estabelecidas pela gramática.

A análise semântica é a terceira parte da compilação, responsável por verificar se valores foram atribuídos corretamente, através da árvore sintática, verificando se o código apresentado faz sentido de acordo com a gramática apresentada pela linguagem.

No atual trabalho, é analisado e implementado o analisador semântico utilizando a linguagem Java através da IDE Netbeans.

Verificação de Tipos

A verificação de tipos implementada no projeto da linguagem foi a estuda em sala que está especificada abaixo:

$$S \rightarrow \text{if } E \text{ then } S_1 \quad \{ \text{ se } E.tipo = booliano \text{ então } S.tipo = S_1.tipo \text{ senão } S.tipo = tipo\_erro \}$$

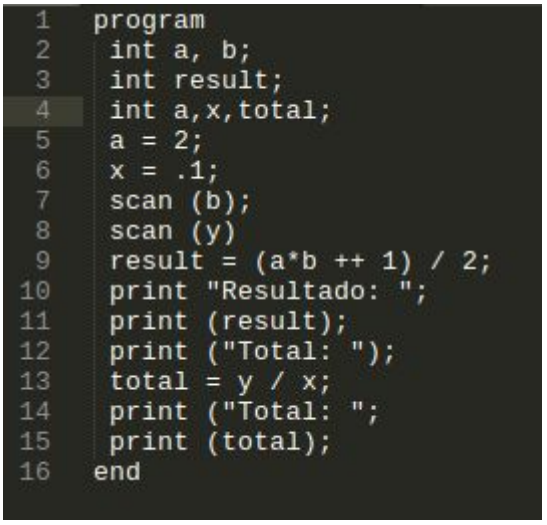
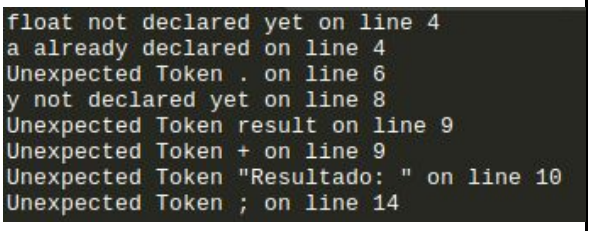
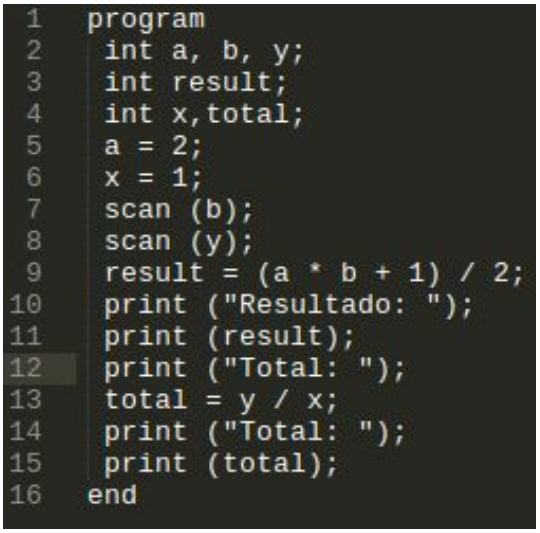
Implementação

Foi implementado a verificação de unicidade e tipo da linguagem projetada em questão.

Para auxiliar a implementação do analisador semântico, a tabela de símbolo foi modificada e transferida sua inserção para o analisador sintático, uma vez que a heurística implementada é a orientada por sintaxe e o compilador em um passo, dessa forma para auxiliar na verificação da unicidade foi passada para essa fase da análise. Com isso, a partir do exemplo demonstrado acima foi realizado todas as verificações na tabela de símbolos e nos comandos da linguagem para realizar a verificação final da front-end.

Testes

Caso de Teste 1:

Programa Fonte - Não corrigido	**Saída**
	
Programa Fonte - Corrigido	**Saída**
	

```
1   program
2    int a, b;
3    int result;
4    int a,x,total;
5    a = 2;
6    x = .1;
7    scan (b);
8    scan (y)
9    result = (a*b ++ 1) / 2;
10   print "Resultado: ";
11   print (result);
12   print ("Total: ");
13   total = y / x;
14   print ("Total: ";
15   print (total);
16  end
```

```
float not declared yet on line 4
a already declared on line 4
Unexpected Token . on line 6
y not declared yet on line 8
Unexpected Token result on line 9
Unexpected Token + on line 9
Unexpected Token "Resultado: " on line 10
Unexpected Token ; on line 14
```

```
1   program
2    int a, b, y;
3    int result;
4    int x,total;
5    a = 2;
6    x = 1;
7    scan (b);
8    scan (y);
9    result = (a * b + 1) / 2;
10   print ("Resultado: ");
11   print (result);
12   print ("Total: ");
13   total = y / x;
14   print ("Total: ");
15   print (total);
16  end
```

```
Compilation sucessful
```

Casos de Teste 2:

Programa Fonte - Não corrigido	**Saída**
`program` `int: a, c;` `float d, _e;` `a = 0; d = 3.5` `c = d / 1.2;` `Scan (a);` `Scan (c);` `b = a * a;` `c = b + a * (1 + a*c);` `print ("Resultado: ");` `print c;` `a = b + c + d)/2;` `e = val + c + a;` `print ("E: ");` `print (e);`	`Unexpected Token : on line 2` `float not declared yet on line 3` `Unexpected Token _e on line 3` `Unexpected Token . on line 4` `Unexpected Token . on line 6` `Scan not declared yet on line 8` `Scan not declared yet on line 9` `b not declared yet on line 10` `Unexpected Token c on line 13` `Unexpected Token ) on line 14` `Unexpected Token EOF on line 18`
Programa Fonte - Corrigido	**Saída**
`program` `int a, b, c;` `int d, e, val;` `a = 0;` `d = 3;` `c = d / 12;` `scan (a);` `scan (c);` `b = a * a;` `c = b + a * (1 + a*c);` `print ("Resultado: ");` `print (c);` `a = (b + c + d)/2;` `e = val + c + a;` `print ("E: ");` `print (e);` `end`	`Compilation sucessful`

Casos de Teste 3:

Programa Fonte - Não corrigido	**Saída**
`program` ` int pontuacao, pontuacaoMaxima, disponibilidade;` ` string pontuacaoMinima;` ` disponibilidade = "Sim";` ` pontuacaoMinima = 50;` ` pontuacaoMaxima = 100;` ` /* Entrada de dados` ` Verifica aprovação de candidatos` ` do` ` print("Pontuacao Candidato: ");` ` scan(pontuacao);` ` print("Disponibilidade Candidato: ");` ` scan(disponibilidade);` ` if ((pontuação > pontuacaoMinima) and (disponibilidade=="Sim") then` ` print("Candidato aprovado");` ` else` ` print("Candidato reprovado")` ` end` ` while (pontuação >= 0)end` `end`	`Type error on line 4` `Type error on line 5` `Unexpected Token ; on line 8` `pontuação not declared yet on line 14` `Unexpected Token and on line 14` `Unexpected Token then on line 14` `Unexpected Token end on line 18` `pontuação not declared yet on line 19`
Programa Fonte - Corrigido	**Saída**
`program` ` int pontuacao, pontuacaoMaxima, pontuacaoMinima;` ` string disponibilidade;` ` disponibilidade = "Sim";` ` pontuacaoMinima = 50;` ` pontuacaoMaxima = 100;` ` /* Entrada de dados Verifica aprovação de candidatos */` ` do` ` print("Pontuacao Candidato: ");` ` scan(pontuacao);` ` print("Disponibilidade Candidato: ");` ` scan(disponibilidade);` ` if ((pontuacao > pontuacaoMinima) && (disponibilidade=="Sim")) then` `  print("Candidato aprovado");` ` else` `  print("Candidato reprovado");` ` end` ` while (pontuacao >= 0)end` `end`	`Compilation sucessful`

Casos de Teste 4:

Programa Fonte - Não corrigido	**Saída**
`int: a, aux$, b;` ` string nome, sobrenome, msg;` ` print(Nome: );` ` scan (nome);` ` print("Sobrenome: ");` ` scan (sobrenome);` ` msg = "Ola, " + nome + " " +` `sobrenome + "!";` ` msg = msg + 1;` ` print (msg);` ` scan (a);` ` scan(b);` ` if (a>b) then` ` aux = b;` ` b = a;` ` a = aux;` ` end;` ` print ("Apos a troca: ");` ` out(a);` ` out(b)` `end`	`Unexpected Token int on line 1` `Unexpected Token : on line 2` `Unexpected Token aux$ on line 2` `Nome not declared yet on line 4` `Type error on line 10` `Unexpected Token ; on line 17` `Unexpected Token ; on line 18` `Unexpected Token end on line 21`
Programa Fonte - Corrigido	**Saída**

`` program int a, aux, b; string nome, sobrenome, msg; print("Nome: "); scan (nome); print("Sobrenome: "); scan (sobrenome); msg = "Ola, " + nome + " " + sobrenome + "!"; msg = msg; print (msg); scan (a); scan(b); if (a>b) then aux = b; b = a; a = aux; end print ("Apos a troca: "); print (a); print (b); end ```	``` Compilation sucessful ```

Casos de Teste 5:

Programa Fonte - Não corrigido	**Saída**
`` program int a, b, c, maior, outro; do print("A"); scan(a); print("B"); scan(b); print("C"); scan(c); //Realizacao do teste if ((a>b) && (a>c)) maior = a else if (b>c) then maior = b; else maior = c; end end print("Maior valor:""); print (maior); print ("Outro? "); scan(outro); while (outro >= 0) end ```	``` Unexpected Token maior on line 13 Unexpected Token "); print (maior); print (" on line 21 Unexpected Token EOF on line 27 ```
Programa Fonte - Corrigido	**Saída**

`` program int a, b, c, maior, outro; do print("A"); scan(a); print("B"); scan(b); print("C"); scan(c); //Realizacao do teste if ((a>b) && (a>c)) then maior = a; else if (b>c) then maior = b; else maior = c; end end print ("Maior valor:"); print (maior); print ("Outro? "); scan(outro); while (outro >= 0) end end ```	``` Compilation sucessful ```

Casos de Teste 6:

Programa Fonte - Não corrigido	**Saída**
`` program int a, b, c; string oi; oi = "kole Lucao"; a = b; c = oi + b; /* c = a + b */ tb = oi; do /* while (a) */ end ```	``` Type error on line 8 tb not declared yet on line 11 Unexpected Token end on line 16 ```
Programa Fonte - Corrigido	**Saída**

`` program int a, b, c; string tb, oi; oi = "kole Lucao"; a = b; /* c = a + b */ tb = oi; end ```	``` Compilation sucessful ```