Desenvolvimento de Um Compilador - Análise Sintática

Pedro H. R. Souza
Departamento de Ciência da Computação – Universidade Federal de Lavras(UFLA)
37.200-000 – Lavras – MG – Brasil
phramos@computacao

1. Introdução

Um compilador é um programa capaz de processar uma dada entrada em texto e a transformar em algo que um computador possa compreender. O processo de compilação possui diversas etapas, entre elas: análise léxica; análise sintática; análise semântica; geração de código intermediário; otimização de código; e geração de código. A análise sintática é uma fase responsável por ler caractere por caractere da entrada e os converter em diversos tokens agrupados de maneira lógica e coerente. Este trabalho tem por objetivo implementar um analisador sintático simplificado de uma linguagem de programação.

2. Metodologia

2.1. Ferramentas e Tecnologias Utilizadas

Este trabalho utilizou diversas tecnologias para o desenvolvimento de um analisador sintático de uma linguagem de programação. Dentre as tecnologias utilizadas, as principais e mais significativas foram: *antlr4*; *Java*; *Intellij IDEA*. Informações mais detalhadas sobre essas ferramentas podem ser obtidas nas subseções deste tópico.

2.1.1. ANTLR4

O antlr 4 é um poderoso gerador de conversores para leitura, processamento, execução, e tradução de arquivos em texto ou binários. Atualmente, o antlr é amplamente utilizado se construir linguagens e ferramentas de programação.

Este trabalho utilizou o *antlr4* para se converter definições de uma gramática em notação padrão dentro de um arquivo .g4 para classes Java que puderam ser utilizadas na implementação de um analisador sintático para a linguagem de programação.

2.1.2. Linguagem Java

Java é uma linguagem de programação orientada à objetos que pode ser utilizada para os mais diversos propósitos, entre eles até mesmo o desenvolvimento de um compilador.

Este trabalho utilizou classes geradas em Java por meio do antlr4 para implementar um analisador semântico de uma linguagem de programação.

2.1.3. Intellij Idea

Intellij IDEA é uma robusto ambiente integrado de desenvolvimento para a linguagem de programação JAVA. Ele é capaz de realizar análises de código estaticamente para poder aumentar a produtividade do desenvolvimento de programas escritos em JAVA.

Este trabalho utilizou o IntelliJ Idea para escrever a gramática do analisador léxico e o analisador sintático em um arquivo .g4, e um arquivo java implementando um listener do antlr4(CustomListener.java) para verificar os erros semânticos, gerar o código intermediário, e gerar as quadruplas do código fonte processado.

2.2. Estrutura do Projeto

Este trabalho foi desenvolvido dentro em uma estrutura proposta pelo autor deste relatório. Um esquema resumido da estrutura deste projeto encontra-se abaixo:

--Pasta raiz do projeto contendo todas as subpastas, arquivos de código, e
--bibliotecas de terceiros.
/

--Arquivos em texto utilizados na execução do programa.
arquivos
--Arquivos de testes utilizados para validação da implementação.
gen
--Arquivos gerados pelo antlr4 utilizados na implementação da solucão.
gramatica
--Arquivos .g4 que são utilizados como base da geração de código pelo antlr4.
lib
--Bibliotecas de terceiros utilizados no projeto, tais como o antlr4.
src
--Arquivos.java

2.3. Soluções Propostas

A solução proposta foi baseada no arquivo base de um analisador sintátito disponibilizado pelo professor Rafael Serapilha Durelli do Departamento de Ciência da Computação da Universidade Federal de Lavras. Mais detalhes da implementação pode ser verificada no arquivo CMinus.g4 disponibilizado juntamente ao código fonte da solução

3. Testes

Foi desenvolvido dois arquivos para auxiliar nos testes e validação da gramática. mais detalhes a respeito dos testes podem ser encontrados nas subseções desse tópico.

3.1. Testes de Sucesso

3.1.1. Sintático

Foram realizados diversos testes de sucesso para validar as principais regras sintáticas implementadas

O arquivo com o código completo pode ser verificado a seguir:

```
struct fauno{
  int altura;
  float peso;
  char sexo = 'M';
};

int variavelglobal;

int main(char args) {
  float a = 666;
  float b;
  float c = 10.0E56;


  while (a != b) {
     if (a == b) {
         a = b - 1;
     } else {
         b = a + 1;
     }
  }

  c = a + b - 35;

  struct fauno2{
     int altura;
     float peso;
     char sexo = 'M';
  };

  return 0;
}
```

3.1.2. Semântico

3.1.2.1. teste3acexpressao1.c

```
void teste3acexpressao1(int a){
  a = 562;
}
```

Resultado:

```
a=562
----------------------QUADRUPLAS----------------------
<op, arg1, arg2, result>
<=, 562, null, a>


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, teste3acexpressao1, teste3acexpressao1>
<int, a, teste3acexpressao1>
```

3.1.2.2. teste3acexpressao2.c

```
void teste3acexpressao2(int a){
  a = 562+56*15-12;
}
```

Resultado:

```
T0=56*15
T1=562+T0
a=T1-12
----------------------QUADRUPLAS----------------------
<op, arg1, arg2, result>
<*, 56, 15, T0>
<+, 562, T0, T1>
<-, T1, 12, a>


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, teste3acexpressao2, teste3acexpressao2>
<int, a, teste3acexpressao2>
```

3.1.2.3. teste3acexpressao3.c

```
void teste3acexpressao3() {
  int a;
  int b;
```

```
  int c;
  int d;
  int e;
  a = a+b*c/d-e;
}
```

Resultado:

```
T0=b*c
T1=T0/d
T2=a+T1
a=T2-e
----------------------QUADRUPLAS----------------------
<op, arg1, arg2, result>
<*, b, c, T0>
</, T0, d, T1>
<+, a, T1, T2>
<-, T2, e, a>


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, teste3acexpressao3, teste3acexpressao3>
<int, a, teste3acexpressao3>
<int, b, teste3acexpressao3>
<int, c, teste3acexpressao3>
<int, d, teste3acexpressao3>
<int, e, teste3acexpressao3>
```

3.1.2.4. teste3acif1.c

```
void teste3acif1() {
  int a;
  int b;

  a = 1;
  a = 2;

  if (a < b) {
     b = a;
  }

  b = a * 2;

}
```

Resultado:

```
a=1
```

```
a=2
ifZ (a<b) goto L0:
b=a
L0:
b=a*2
----------------------QUADRUPLAS----------------------
<op, arg1, arg2, result>
<=, 1, null, a>
<=, 2, null, a>
<ifZ, (a<b), null, goto L0:>
<=, a, null, b>
<Label, null, null, L0:>
<*, a, 2, b>


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, teste3acif1, teste3acif1>
<int, a, teste3acif1>
<int, b, teste3acif1>
```

3.1.2.5. teste3acif2.c

```
void teste3acif2() {
  int a;
  int b;

  a = 1;
  a = 2;

  if (a < b) {
     b = a;
  } else {
     a = b;
  }
}
```

Resultado:

```
a=1
a=2
ifZ (a<b) goto L0:
b=a
L0:
a=b
----------------------QUADRUPLAS----------------------
<op, arg1, arg2, result>
<=, 1, null, a>
<=, 2, null, a>
<ifZ, (a<b), null, goto L0:>
<=, a, null, b>
```

```
<Label, null, null, L0:>
<=, b, null, a>


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, teste3acif2, teste3acif2>
<int, a, teste3acif2>
<int, b, teste3acif2>
```

3.1.2.6. teste3acif3.c

```
void teste3acif3() {
  int a;
  int b;

  a = 1;
  a = 2;

  if (a < b) {
      b = a;

      if (b > b) {
          b = b*2;
      } else {
          a = a*2;
      }
  }
  int c;

  c = a+b;
}
```

Resultado:

```
a=1
a=2
ifZ (a<b) goto L0:
b=a
ifZ (b>b) goto L1:
b=b*2
L1:
a=a*2
L0:
c=a+b
----------------------QUADRUPLAS----------------------
<op, arg1, arg2, result>
<=, 1, null, a>
<=, 2, null, a>
```

```
<ifZ, (a<b), null, goto L0:>
<=, a, null, b>
<ifZ, (b>b), null, goto L1:>
<*, b, 2, b>
<Label, null, null, L1:>
<*, a, 2, a>
<Label, null, null, L0:>
<+, a, b, c>


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, teste3acif3, teste3acif3>
<int, a, teste3acif3>
<int, b, teste3acif3>
<int, c, teste3acif3>
```

3.1.2.7. teste3acifwhile.c

```
void teste3acifwhile() {
  int a;
  int b;
  a = 1;
  b = 10;

  if (a > b) {

     while (a < b) {
        b = b - 1;
     }

     a = a*2;
  }

  a = 0;
  b = 0;
}
```

Resultado:

```
a=1
b=10
ifZ (a>b) goto L0:
L1:
ifZ (a<b) goto L2:
b=b-1
goto L1:
L2:
a=a*2
L0:
```

```
a=0
b=0
----------------------QUADRUPLAS----------------------
<op, arg1, arg2, result>
<=, 1, null, a>
<=, 10, null, b>
<ifZ, (a>b), null, goto L0:>
<Label, null, null, L1:>
<ifZ, (a<b), null, goto L2:>
<-, b, 1, b>
<ifZ, (a<b), null, goto L1:>
<Label, null, null, L2:>
<*, a, 2, a>
<Label, null, null, L0:>
<=, 0, null, a>
<=, 0, null, b>


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, teste3acifwhile, teste3acifwhile>
<int, a, teste3acifwhile>
<int, b, teste3acifwhile>
```

3.1.2.8. teste3acwhile1.c

```
void teste3acwhile1(){
  int a;
  int b;
  int c;
  a = 1;
  b = 10;
  while (a>b) {
     c = c + a *2;
     a = a+1;
  }
  b = 0;
}
```

Resultado:

```
a=1
b=10
L0:
ifZ (a>b) goto L1:
T0=a*2
c=c+T0
a=a+1
goto L0:
L1:
```

```
b=0
----------------------QUADRUPLAS----------------------
<op, arg1, arg2, result>
<=, 1, null, a>
<=, 10, null, b>
<Label, null, null, L0:>
<ifZ, (a>b), null, goto L1:>
<*, a, 2, T0>
<+, c, T0, c>
<+, a, 1, a>
<ifZ, (a>b), null, goto L0:>
<Label, null, null, L1:>
<=, 0, null, b>


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, teste3acwhile1, teste3acwhile1>
<int, a, teste3acwhile1>
<int, b, teste3acwhile1>
<int, c, teste3acwhile1>
```

3.1.2.9. teste3acwhile2.c

```
void teste3acwhile2(){
  int a;
  int b;
  int c;
  int d;
  int e;
  a = 1;
  b = 10;
  c = 1;
  d = a + b * 2;;
  e = 0;
  while (a>b) {
      e = e + a *2;
      a = a+1;

      while (c >d) {
          c= c*2;
      }
      d = a + b;
  }
  b = 0;
}
```

Resultado:

```
a=1
b=10
c=1
T0=b*2
d=a+T0
e=0
L0:
ifZ (a>b) goto L1:
T0=a*2
e=e+T0
a=a+1
L2:
ifZ (c>d) goto L3:
c=c*2
goto L2:
L3:
d=a+b
goto L0:
L1:
b=0
----------------------QUADRUPLAS----------------------
<op, arg1, arg2, result>
<=, 1, null, a>
<=, 10, null, b>
<=, 1, null, c>
<*, b, 2, T0>
<+, a, T0, d>
<=, 0, null, e>
<Label, null, null, L0:>
<ifZ, (a>b), null, goto L1:>
<*, a, 2, T0>
<+, e, T0, e>
<+, a, 1, a>
<Label, null, null, L2:>
<ifZ, (c>d), null, goto L3:>
<*, c, 2, c>
<ifZ, (c>d), null, goto L2:>
<Label, null, null, L3:>
<+, a, b, d>
<ifZ, (a>b), null, goto L0:>
<Label, null, null, L1:>
<=, 0, null, b>


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, teste3acwhile2, teste3acwhile2>
<int, a, teste3acwhile2>
<int, b, teste3acwhile2>
<int, c, teste3acwhile2>
<int, d, teste3acwhile2>
<int, e, teste3acwhile2>
```

3.1.2.10. teste3acwhileif.c

```
void teste3acwhileif() {
  int a;
  int b;
  a = 1;
  b = 10;

  while (a > b) {

     if (a < b) {
        b = b - 1;
     }

     a = a*2;
  }

  a = 0;
  b = 0;
}
```

Resultado:

```
a=1
b=10
L0:
ifZ (a>b) goto L1:
ifZ (a<b) goto L2:
b=b-1
L2:
a=a*2
goto L0:
L1:
a=0
b=0
----------------------QUADRUPLAS----------------------
<op, arg1, arg2, result>
<=, 1, null, a>
<=, 10, null, b>
<Label, null, null, L0:>
<ifZ, (a>b), null, goto L1:>
<ifZ, (a<b), null, goto L2:>
<-, b, 1, b>
<Label, null, null, L2:>
<*, a, 2, a>
<ifZ, (a>b), null, goto L0:>
<Label, null, null, L1:>
<=, 0, null, a>
<=, 0, null, b>
```

```
----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, teste3acwhileif, teste3acwhileif>
<int, a, teste3acwhileif>
<int, b, teste3acwhileif>
```

3.2. Testes de Erros

3.2.1. Sintaticos

Foram definidos 10 arquivos contendo erros sintáticos para validar a implementação da sintaxe da linguagem C-.

3.2.1.1. teste_errado_colchete_main.c

Trecho de código:

```
int main(char args) {
  return 0;
/*ERRO AQUI - falta um fecha chaves*/
```

Resultado:

```
line 5:0 missing '}' at '<EOF>'

Process finished with exit code 0
```

3.2.1.2. teste_errado_colchete_main.c

Trecho de código:

```
int main(char args) {
  /*ERRO - Falta parentese aqui*/
  if args) {
      return 0;
  }
}
```

Resultado:

```
line 3:7 missing '(' at 'a'

Process finished with exit code 0
```

3.2.1.3. teste_errado_ponto_e_virgula.c

Trecho de código:

```
int main(char args) {
  /*ERRO AQUI - falta o ;*/
  return 0
}
```

Resultado:

```
line 4:0 mismatched input '}' expecting {'>', ';', '=', '<', '==', '<=',
'>=', '!=', '+', '-', '*', '/'}

Process finished with exit code 0
```

3.2.1.4. teste_errado_tipo_se_identificador1.c

Trecho de código:

```
int main(char ) {
  return 0;
}
```

Resultado:

```
line 1:14 missing Identifier at ')'

Process finished with exit code 0
```

3.2.1.5. teste_errado_tipo_se_identificador2.c

Trecho de código:

```
int main(char args ) {
  int 0;
  return 0;
}
```

Resultado:

```
line 2:8 mismatched input '0' expecting Identifier

Process finished with exit code 0
```

3.2.1.6. teste_errado_identificador_identificador.c

Trecho de código:

```
int main(char args ) {
  int int;
```

```
    return 0;
}
```

Resultado:

```
line 2:8 mismatched input 'int' expecting Identifier

Process finished with exit code 0
```

3.2.1.7. teste_errado_argumentos1.c

Trecho de código:

```
int main(char args,) {
    return 0;
}
```

Resultado:

```
line 1:19 mismatched input ')' expecting {'int', 'float', 'char'}

Process finished with exit code 0
```

3.2.1.8. teste_errado_argumentos2.c

Trecho de código:

```
int qualquercoisa(char args, beluga) {
    return 0;
}
```

Resultado:

```
line 1:29 missing {'int', 'float', 'char'} at 'beluga'

Process finished with exit code 0
```

3.2.1.9. teste_errado_struct1.c

Trecho de código:

```
struct fauno {
}
```

Resultado:

```
line 3:0 mismatched input '}' expecting {'int', 'float', 'char'}

Process finished with exit code 0
```

3.2.1.10. teste_errado_struct2.c

Trecho de código:

```
struct pessoa {
   int idade;
   list associados;
};
```

Resultado:

```
line 3:4 extraneous input 'list' expecting {'int', 'float', 'char', '}'}

Process finished with exit code 0
```

3.2.2. Semânticos

3.2.2.1. erroatribuicao1.c

```
void erroatribuicao1(){
  int a;
  a = 6.0;
}
```

Resultado:

```
ERRO: Tipo encontrado em 6.0 não compativel com a. Esperado int mas
foi encontrado float em L3C8


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, erroatribuicao1, erroatribuicao1>
<int, a, erroatribuicao1>
```

3.2.2.2. erroatribuicao2.c

```
void erroatribuicao2(float a){
  int b;
  b = a;
}
```

Resultado:

```
ERRO: Tipo encontrado em a não compativel com b. Esperado int mas foi
encontrado float em L3C8


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, erroatribuicao2, erroatribuicao2>
<float, a, erroatribuicao2>
<int, b, erroatribuicao2>
```

3.2.2.3. erroatribuicao3.c

```
void erroatribuicao3(){
  int a;
  float b;
  a = b;
```

```
}
```

Resultado:

```
ERRO: Tipo encontrado em b não compativel com a. Esperado int mas foi encontrado float em L4C8


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, erroatribuicao3, erroatribuicao3>
<int, a, erroatribuicao3>
<float, b, erroatribuicao3
```

3.2.2.4. errochar1.c

```
void errochar1() {
  int a;
  char b;

  b = a;
}
```

Resultado:

```
ERRO: Conversão entre tipos de b e a. Esperado char mas foi encontrado int em L5C8


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, errochar1, errochar1>
<int, a, errochar1>
<char, b, errochar1>
```

3.2.2.5. errodeclaracaoatribuicao1.c

```
void errodeclaracaoatribuicao1() {
  char a = 'c' + 2;
  int b;
}
```

Resultado:

```
ERRO: Operação inválida sobre o tipo char.Tipo char permite apenas a opeção de atribuição(=), mas foi econtrada uma operação (+ | -) em L2C19
ERRO: Conversão entre tipos de 'c' e 2. Esperado char mas foi encontrado int em L2C19
```

```
----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, errodeclaracaoatribuicao1, errodeclaracaoatribuicao1>
<char, a, errodeclaracaoatribuicao1>
<int, b, errodeclaracaoatribuicao1>
```

3.2.2.6. errodeclaracaoatribuicao2.c

```
void errodeclaracaoatribuicao2(int b) {
  char a = b;
}
```

Resultado:

```
ERRO: Conversão entre tipos de a e b. Esperado char mas foi
encontrado int em L2C13
```

```
----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, errodeclaracaoatribuicao2, errodeclaracaoatribuicao2>
<int, b, errodeclaracaoatribuicao2>
<char, a, errodeclaracaoatribuicao2>
```

3.2.2.7. errodeclaracaoduplicada1.c

```
void errodeclaracaoduplicada1(int b, int b){

}
```

Resultado:

```
ERRO: b já declarado no escopo errodeclaracaoduplicada1 em L1C41
```

```
----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, errodeclaracaoduplicada1, errodeclaracaoduplicada1>
<int, b, errodeclaracaoduplicada1>
```

3.2.2.8. errodeclaracaoduplicada2.c

```
void errodeclaracaoduplicada2(){
  int a;
  int a;
```

```
}
```

Resultado:

```
ERRO: a já declarado no escopo errodeclaracaoduplicada2 em L3C8


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, errodeclaracaoduplicada2, errodeclaracaoduplicada2>
<int, a, errodeclaracaoduplicada2>
```

3.2.2.9. errodeclaracaoduplicada3.c

```
void errodeclaracaoduplicada3(int a){
  int a;
}
```

Resultado:

```
ERRO: a já declarado no escopo errodeclaracaoduplicada3 em L2C8


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, errodeclaracaoduplicada3, errodeclaracaoduplicada3>
<int, a, errodeclaracaoduplicada3>
```

3.2.2.10. erroescopo1a1b.c

```
void erroescopo1a(){
  int a;
}


void erroescopo1b(){
  int b;
  b = a;
}
```

Resultado:

```
ERRO: Identificador a não declarado em L8C8


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, erroescopo1a, erroescopo1a>
<int, a, erroescopo1a>
```

```
<function, erroescopo1b, erroescopo1b>
<int, b, erroescopo1b>
```

3.2.2.11. erroexpressao1.c

```
void erroexpressao1() {
  int a;
  float b;
  a = a + b;
}
```

Resultado:

```
ERRO: Tipo encontrado em b não compativel com a. Esperado int mas foi encontrado float em L4C12


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, erroexpressao1, erroexpressao1>
<int, a, erroexpressao1>
<float, b, erroexpressao1>
```

3.2.2.12. erroexpressao2.c

```
void erroexpressao2() {
  int a;
  int b;
  a = a + 1.0 + b;
}
```

Resutado

```
ERRO: Tipo encontrado em b não compativel com 1.0. Esperado float mas foi encontrado int em L4C18
ERRO: Tipo encontrado em 1.0 não compativel com a. Esperado int mas foi encontrado float em L4C12


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, erroexpressao2, erroexpressao2>
<int, a, erroexpressao2>
<int, b, erroexpressao2>
```

3.2.2.13. errooperadorchar1.c

```
void errooperadorchar1() {
```

```
  int a;
  char b;

  b = 6.0 + b;
}
```

Resultado:

```
ERRO: Conversão entre tipos de b e 6.0. Esperado char mas foi
encontrado float em L5C8
ERRO: Operação inválida sobre o tipo char.Tipo char permite apenas a
opeção de atribuição(=), mas foi econtrada uma operação (+ | -) em
L5C14
ERRO: Conversão entre tipos de 6.0 e b. Esperado float mas foi
encontrado char em L5C14


----------------------TABELA DE SIMBOLOS----------------------
<tipo, identificador, escopo>
<function, errooperadorchar1, errooperadorchar1>
<int, a, errooperadorchar1>
<char, b, errooperadorchar1>
```

4. Conclusão

Através da implementação do analisador sintático da linguagem proposta por este trabalho pôde-se concluir que é possível realizar a implementação de um analisador sintático, um analisador semântico, um gerador de código de 3 endereços para expressões aritmétias, um gerador de código de 3 endereços para expressões if-else, um gerador de código de 3 endereços para expressões while, e um gerador de quadruplas para a linguagem C- utilizando o antlr4 juntamente aos conhecimentos obtidos na disciplina de compiladores associado ao livro texto da disciplina.

5. Referencial Teórico

Alfred V. Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman(2006) “Compilers: Principles, Techniques, and Tools”.
http://www.antlr.org/, Fevereiro de 2016.
https://www.oracle.com/java/index.html, Fevereiro de 2016.
http://compilerdesigndetails.blogspot.com.br/2012/02/v-behaviorurldefaultvmlo.html, Fevereiro de 2016.
http://www.antlr.org/api/Java/org/antlr/v4/runtime/Parser.htm,l Fevereiro de 2016.
http://www.facweb.iitkgp.ernet.in/~niloy/COURSE/Autumn2006/Compiler/notes/TAC3005.doc Março de 2017.
http://www.cs.columbia.edu/~aho/cs4115/lectures/14-03-31.htm Março de 2017.
http://digital.cs.usu.edu/~allan/Compilers/Slides/IC.pdf Março de 2017.
https://www.tutorialspoint.com/compiler_design/compiler_design_intermediate_code_generations.htm Março de 2017.
http://cse.iitkgp.ac.in/~bivasm/notes/scribe/11CS30026.pdf Março de 2017.
http://www.cs.nyu.edu/courses/spring10/G22.2130-001/lecture9.pdf Março de 2017.