Especificação Fase 1

Bacharelado em Ciência da Computação
Universidade Federal de São Carlos
Campus Sorocaba
Compiladores

18 de Março de 2015

1 Primeira fase

A primeira fase do trabalho consiste na implementação de um compilador que realiza a análise léxica e sintática da gramática da linguagem, além de gerar código em linguagem C.

Devido ao pequeno escopo, a gramática apresentada na próxima seção é apenas um subconjunto da gramática real, sendo que ela irá aumentar gradativamente a cada fase do trabalho.

2 Entrega

Data: **31/03/15**

Você deve entregar todos os arquivos `.java`. No arquivo `Main.java`, você deve manter todos os testes realizados.

3 Gramática

Esta seção define a gramática da linguagem. Ao longo de todas as fases deste trabalho, a gramática utilizada será uma versão simplificada da linguagem de programação Pascal, baseada em [1].

Em cada linha há a descrição de uma regra de produção da gramática. As palavras-chave da linguagem estão em letras maiúsculas, utilizando uma única letra distinta entre elas. Os símbolos da linguagem são mostrados entre apóstrofos. Os símbolos não-terminais da gramática são aqueles descritos por palavras em letras minúsculas.

Uma sequência de símbolos entre `[` e `]` é opcional, enquanto que uma sequência de símbolos entre `{` e `}` pode ser repetida zero ou mais vezes. Qualquer sequência de caracteres no arquivo fonte encontrado entre `{` e `}` deve ser tratado como um comentário.

Considere ainda que a primeira letra de `id` sempre é minúscula e diferente das letras utilizadas como palavras chave.

```
prog          ::= P pid ';' body '.'

body          ::= [dclpart] compstmt

dclpart       ::= V dcls

dcls          ::= dcl {dcl}

dcl           ::= idlist ':'  type ';'

idlist        ::= id {',' id}

type          ::= stdtype

stdtype       ::= I

compstmt      ::= B stmts E

stmts         ::= stmt {';' stmt} ';'

stmt          ::= L '(' vblist ')'
```

```
vblist          ::= vbl {',' vbl}

vbl             ::= id

id              ::= letter {letter | digit}

pid             ::= letter {letter | digit}
```

4 Instruções

O seu compilador deve se basear no Compilador 6 da apostila. Isto é, a partir de um programa descrito na gramática acima (implementado no arquivo `Main.java`), o seu compilador deve fazer a análise léxica, sintática e gerar código em linguagem C. Observe que seu programa deve:

- gerar uma mensagem de erro no caso de uma situação que o compilador não esperava (símbolo não reconhecido pela linguagem, símbolo faltante ou no lugar errado, falta de palavra reservada, etc.); ou
- caso contrário, gerar um arquivo de saída de extensão `.c` com o código em C equivalente ao código do arquivo de entrada.

Um compilador correto que receba um arquivo de entrada com o código abaixo deve apontar erro na terceira linha, pois ao final dessa linha o símbolo `;` era esperado.

```
P exemplo01ErroLexico;
    B
        L(a)   { # }
    E.
```

Enquanto que utilizando como entrada o exemplo de código abaixo

```
P exemplo02Correto;
    V a : I;

    B
        L(a);
    E.
```

um compilador correto deve gerar o seguinte código em C:

```
#include <stdio.h>

int main() {
    int a;

    scanf("%d", &a);

    return 0;
}
```

Referências

[1] Alfred Aho, Monica Lam, Ravi Sethi e Jeffrey Ullman. *Compilers: Principles, Techniques and Tools.* Addison-Wesley, Technical, 1986.