



**SCC0217 - Linguagens de Programação e
Compiladores**

Professor: Diego Raphael Amancio

**Trabalho 1 - Analisador Léxico para a linguagem
LALG**

**Gabriela Pinto Cesar Duque - 7694220
Pedro Henrique Chagas Anchieta - 7696264
Ricardo Cardoso Cunha - 7696330**

abril de 2015

Parte 1: Decisões de projeto

Foram tomadas as seguintes decisões:

1. Todos os tokens estão identificados pelo analisador léxico. Caso seja uma palavra, é verificada na tabela hash se essa palavra é reservada ou não. Se não for palavra reservada, ela é interpretada como identificador.
2. Os nomes dos símbolos serão utilizados como nome dos tokens. Nos foram dadas duas opções de nomes dos tokens: simbolo_simbolox ou apenas simbolox. O grupo optou pela segunda alternativa, pois achamos que é mais simplificada.
3. No .l são identificados comentários, símbolos, caracteres não reconhecidos, número inteiro e real, variáveis, são removidas quebras de linha, são reconhecidos erros de formatação de número real e inteiro.

Parte 2: Especificação do analisador léxico

O analisador léxico é responsável pelo tratamento de erros por expressões regulares, e encontra-se em trabalho1.l.

Já o trabalho1.c é responsável por criar duas hashes, uma preenchida com todos os valores das palavras reservadas e a outra com os símbolos. É retornado da hash um nó com as informações referente àquela palavra reservada. O próximo passo é preencher as tabelas de palavras reservadas e símbolos.

O loop é responsável pelos seguintes passos:

- busca por erros tratados no trabalho1.l e, caso tenha algum, imprime mensagem de erro
- busca por outra expressão regular que não seja identificador e que foi reconhecida em trabalho1.l. Verifica na hash se esse identificador é uma palavra reservada ou algum símbolo específico da linguagem, ou se é apenas um identificador qualquer
- se o token foi identificado com sucesso, uma mensagem é impressa na tela, do tipo "token_lido: nome_token"
- Se for identificado um erro, será impressa uma mensagem na tela do tipo "token_lido: ERRO: tipo_erro"

Em trabalho1.h estão as definições de valor de cada tipo de token pelo seu nome (exemplo: nome_token, com nome_token em português).

Funções da hash:

- `Lookup_string`: é responsável por verificar se um token é identificado por algum valor que já está presente na tabela hash atual. Se tiver, retorna o nó com as informações do tipo da variável.
- `add_string`: preenche a hash com os valores pré-definidos
- `populate_list`: contém as informações das palavras reservadas. É utilizada para preencher a hash.

Parte 3: Passo a passo para compilação

1. Compilar o trabalho1.l: `$ flex trabalho1.l`
2. Compilar os arquivos .c: `$ gcc -o analisador lex.yy.c trabalho1.c hash.c`
3. Rodar: `$./analisador < input`

Parte 4: Exemplos

- Exemplo 1:

```
program lalg;  
{exemplo2}  
var a: real;  
begin  
  a = 1.2@3;  
end
```

- Exemplo 2:

```
program teste ;  
var minha_enorme_variavel: integer ;  
{comentario}  
begin  
  read(a) ;  
  a := @ + a2 + 11.5 ;  
  a:=1a.5+a2.4*1.0a/1.t4##1fg5hjk.8fj5hk  
  1.  
  write(a) ;  
end .
```

Parte 5: Teste

```
pedro@ubuntu: ~/Documents/Compiladores/Ver3
File Edit View Search Terminal Help
pedro@ubuntu:~/Documents/Compiladores/Ver3$ ./analizador <input
program - program - palavra reservada
teste - identificador
; - ponto e virgula
var - var - palavra reservada
minha_enorme_variavel - ERRO :Identificador excedeu tamanho limite
integer - integer - palavra reservada
; - ponto e virgula
begin - begin - palavra reservada
read - read - palavra reservada
( - parenteses esquerdo
a - identificador
) - parenteses direito
; - ponto e virgula
a - identificador
:= - atribuicao
@ - caracter invalido na linguagem
+ - mais
a2 - ERRO: numero inteiro mal formado
+ - mais
11.5 - numero real
; - ponto e virgula
a - identificador
:= - atribuicao
1a.5 - ERRO: numero real mal formado
+ - mais
a2.4 - ERRO: numero real mal formado
* - vezes
1.0a - ERRO: numero real mal formado
/ - dividido
1.t4 - ERRO: numero real mal formado
## - caracter invalido na linguagem
1fg5hjk.8fj5hk - ERRO: numero real mal formado
1. - ERRO: numero real mal formado
write - write - palavra reservada
( - parenteses esquerdo
a - identificador
) - parenteses direito
; - ponto e virgula
end - end - palavra reservada
. - ponto
pedro@ubuntu:~/Documents/Compiladores/Ver3$
```