Teoria da computação e compiladores - SCC0605
Trabalho 1

Guilherme Prearo - 8910409
Guilherme Villela Pelizer - 8910243
Bruno de Andrade Stefano - 8910156

18 de Abril de 2017

1 Decisões de projeto e implementação

O grupo implementou o analisador léxico da linguagem *LALG* utilizando o programa *flex*.

As palavras reservadas que o nosso analisador identifica são:

- *program*
- *procedure*
- *var*
- *const*
- *begin*
- *end*
- *if*
- *then*
- *else*
- *integer*
- *real*
- *char*

O analisador também identifica os operadores aritméticos (+, -, * e /), dois pontos, ponto e vírgula, operador de atribuição, comparadores numéricos e linhas de comentário.

2 Tratamento de erros

Números inteiros e reais são identificados e possíveis erros são acusados. No caso do erro em números inteiros, foi utilizada a seguinte expressão regular

$\{digit\}+\{alpha\}+\{alpha\_num\}$, ela identifica que se houver alguma letra entre números, o inteiro é inválido. Também tratamos o erro em números reais, com a seguinte expressão $(\{num\_int\_erro\}\backslash.\{alpha\_num\}+)|(\{num\_int\}\backslash.\{num\_int\}*\{alpha\}+\{alpha\_num\}*)$, se houver um inteiro inválido antes do ponto ou depois o analisador acusa erro.

Os caracteres também são identificados e os possíveis erros tratatos. Para identificar um carater, utilizamos a expressão \'.\' que engloba um único caractere entre apóstrofes. Para o tratamento de erro foi utilizada a expressão \'.[^\']+\'*, ela identifica o aparecimento de mais de um caractere entre apóstrofes ou o não fechamento do mesmo.

O analisador consegue também tratar o erro no caso em que há o início do comentário e o fim do arquivo antes do fechamento dele.

3 Utilização

Utilizamos o Ubuntu 16.04 para a implementação do trabalho.

É necessário instalar o *flex* e o *gcc* utilizando a seguinte linha de comando:

```
sudo apt-get install flex gcc
```

Então, no diretório que o arquivo estiver, executar a seguinte linha de comando:

```
flex nomedoarquivo.l
```

Esta linha de comando irá gerar um arquivo .c chamado lex.yy.c, então devemos compilá-lo:

```
gcc lex.yy.c -o out
```

Finalmente podemos utilizar o analisador léxico com algum arquivo de entrada:

```
./out < entrada
```

3.1 Exemplo de entrada e saída

Entrada:

```
program lalg;
{entrada}
var a: integer;
begin
readd(a, @, 1);
4a.44
44.4a
5
56.4
end.
'a
```

Saída:

```
program - program
lalg - id
; - simb_ponto_virgula
var - var
a - id
: - simb_dois_pontos
integer - integer
; - simb_ponto_virgula
begin - begin
readd - id
( - simb_abre_parentese
a - id
, - simb_virgula
Erro: "Entrada não identificada" na linha 4. Token = @
, - simb_virgula
1 - num_int
) - simb_fecha_parentese
; - simb_ponto_virgula
Erro: "Número real inválido" na linha 5. Token = 4a.44
Erro: "Número real inválido" na linha 6. Token = 44.4a
5 - num_int
56.4 - num_real
end - end
. - simb_ponto
Erro: "Caracter inválido!" na linha 10. Token = 'a
```