

	Curso	Ciência da Computação
	Atividade Acadêmica	Tradutores
	Ano/Semestre	2017/02
	Professor	Leandro Teodoro
	Data	21/09/2017
	TRABALHO 1 – Tradutores	

Construa um Analisador Léxico que reconheça:

- **Variáveis ou identificadores:** este analisador léxico deve ser capaz de reconhecer variáveis declaradas e/ou utilizadas em um código fonte:

Exemplo:

- Trecho de código:

```
int x = 7;
```

```
int y;
```

- Tokens gerados:

```
[reserved_word, int] [id, 1] [Equal_Op, =] [num, 7]
```

```
[reserved_word, int] [id, 2]
```

- **Constantes numéricas (números inteiros):** este analisador léxico deve ser capaz de reconhecer um número inteiro qualquer e convertê-lo para os respectivos tokens:

Exemplo:

- Trecho de código:

```
int x = 7 + 25 * 52;
```

- Tokens gerados:

```
[reserved_word, int] [id, 1] [Equal_Op, =] [num, 7] [Arith_Op, +] [num, 25]
```

```
[Arith_Op, *] [num, 52]
```

- **Palavras reservadas:** este analisador léxico deve ser capaz de reconhecer as palavras reservadas: *do, while, if, else, switch, for, return, null, int, float, double, string, bool* e convertê-las para os respectivos tokens:

Exemplo:

- Trecho de código:

```
if( x == 10 )
```

- Tokens gerados:

```
[reserved_word, if] [id, 1] [Relational_Op, ==] [num, 10]
```

- **Operadores relacionais:** este analisador léxico deve ser capaz de reconhecer os operadores relacionais: *<, <=, ==, !=, >=, >* e convertê-los para os respectivos tokens:

Exemplo:

- Trecho de código:

```
while( x != 0)
```

- Tokens gerados:

```
[reserved_word, while] [id, 1] [Relational_Op, !=] [num, 0]
```

- **Números de ponto flutuante (números reais):** este analisador léxico deve ser capaz de reconhecer números reais quaisquer e convertê-los para os respectivos tokens:

Exemplo:

- Trecho de código:

```
int x = 7.15 - 2.13;
```

- Tokens gerados:

```
[reserved_word, int] [id, 1] [Equal_Op, =] [num, 7.15] [Arith_Op, -] [num, 2.13]
```

- **Funções ou métodos de uma classe:** este analisador léxico deve ser capaz de reconhecer funções e métodos em um programa e convertê-los para os respectivos tokens:

Exemplo:

- Trecho de código:

```
int CadastrarProduto()
```

```
{
```

```
.....
```

```
}
```

- Tokens gerados:

```
[reserved_word, int] [function, CadastrarProduto]
```

- **Remoção de espaços em branco e comentários:** este analisador léxico deve ser capaz de reconhecer espaços em branco e comentários no código fonte e removê-los (ignorá-los) .

Exemplo:

- Trecho de código:

```
//Comentário 1
```

```
/* Comentário 2 */
```

- **Strings:** este analisador léxico deve ser capaz de reconhecer os strings e convertê-las para seus respectivos tokens:

Exemplo:

- Trecho de código:

```
String sexo = "masculino";
```

- Tokens gerados:

```
[reserved_word, String] [id, 1] [equal, =] [string_literal, masculino]
```

- **Operadores lógicos:** este analisador léxico deve ser capaz de reconhecer os operadores lógicos: | | && e convertê-los para os respectivos tokens:

Exemplo:

- Trecho de código:

```
if(idade > 70 && sexo == "masculino")
```

- Tokens gerados:

```
[reserved_word, if] [id, 1] [Relational_Op, >] [num, 70] [logic_op, &&] [id, 2]
```

```
[Relational_Op, ==] [Relational_Op, string_literal]
```

- **Demais caracteres:** este analisador léxico deve ser capaz de reconhecer os caracteres: = () { } , ; e convertê-los para seus respectivos tokens:

Exemplo:

```
[equal, =] [l_paren, (] [r_paren, )] [l_bracket, {] [r_bracket, }]
```

```
[r_bracket, }] [comma, ,] [semicolon, ;]
```

O trabalho pode ser realizado em grupos de até **3 alunos**, bem como deverá ser entregue pelo Moodle até o dia **21/09** e apresentado ao professor nessa mesma data. A seguir, o código que o analisador léxico deve receber para gerar o conjunto de tokens descrito anteriormente:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
void funcao1()
```

```
{
```

```
    float NotaDaP1, NotaDaP2;
```

```
    float Media;
```

```
    clrscr(); // Limpa a tela
```

```
    NotaDaP1 = 6.6; // Atribuição do Valores das médias
```

```
    NotaDaP2 = 8.2;
```

```
    Media = (NotaDaP1 + NotaDaP2) / 2.0;
```

```
    printf("Média Final : %6.3f", Media);
```

```
    // No momento da execução sinal %6.3f vai ser substituído
```

```
    // pelo valor da variável Media
```

```
    // Média Final : 7.400
```

```
    getch();          // espera que o usuário pressione uma tecla
```

```
}
```

```
int funcao2()
```

```
{
```

```
    int num;
```

```
    string s;
```

```
    printf ("Digite um número: ");
```

```

scanf ("%d",&num);
if (num>10)
{
    printf ("\n\n O número e maior que 10");
    s = "errou";
}
if (num==10)
{
    printf ("\n\n Você acertou!\n");
    printf ("O numero é igual a 10.");
    s = "acertou";
}
if (num<10)
{
    printf ("\n\n O número é menor que 10");
    s = "errou";
}
if(num == 10 && s == "acertou")
{
    return 1;
}
return 0;
}

int main()
{
    funcao1();
    funcao2();
    return 1;
}

```

O trabalho pode ser elaborado na linguagem que o grupo definir, C, C++, Java.

Um ótimo trabalho a todos!