

# Analizador Lexico

## Parte 1: Analisador Léxico

Nesta parte do projeto você deve implementar um analisador léxico para a linguagem. Na sua implementação, considere as seguintes classes de tokens na linguagem:

<b>ID</b>	Identificador
<b>NUM</b>	Literal decimal (inteiro)
<b>KEY</b>	Palavra-chave
<b>SYM</b>	Símbolos léxicos
<b>ERROR</b>	Lexema do primeiro erro encontrado

Comentários e espaços em branco devem ser descartados durante a fase de análise léxica do compilador. Deve-se imprimir também o número da linha onde foi encontrado o token, incluindo o token de erro. O analisador léxico deve parar após encontrar o primeiro erro léxico. **No caso de um erro de comentário não finalizado, deve-se usar o lexema "/" para o token ERROR.** O formato de saída é:

(Num\_Linha, Tipo-Token, "Lexema")

### ▪ Exemplo de arquivo de entrada (main.c):

```
void main(void)
{
    int a;
    a = 4 + 5;
}
```

### ▪ Como executar (dois argumentos: entrada e saída)

O programa deve ler a entrada de um arquivo (fonte) e gravar a saída em um outro arquivo (alvo).

```
$ ./lexico main.c main.lex
```

### ▪ Exemplo de arquivo de saída (main.lex)

```
(1,KEY,"void")
(1,ID,"main")
```

(1,SYM,"(")  
MATA61 Compiladores  
(1,SYM,")")  
(2,SYM,"{")  
(3,KEY,"int")  
(3,ID,"a")  
(3,SYM,";")  
(4,ID,"a")  
(4,SYM,"=")  
(4,NUM,"4")  
(4,SYM,"+")  
(4,NUM,"5")  
(4,SYM,";")  
(5,SYM,"}")