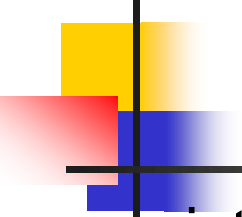


# Análise Léxica

**Generalidades** [KOWALTOWSKI, 83; pag.76-78]:

- Um programa em Pascal é uma sentença de uma GLC, cujo vocabulário terminal inclui vários tipos de símbolos :
  - letras: a, b, c, ..., z      - dígitos: 0, 1, 2, ..., 9
  - símbolos especiais: ; , : , . , ( , ) , = , < , > , + , - , \* ,  
[ , ] , { , } , ' , \_
  - símbolos especiais compostos: := , ( \* , \* ) , < = , > = , < > , ..
  - símbolos compostos em negrito: **program**, **label**,  
**type**, **array**, **of**, **var**, **procedure**, **function**, **begin**,  
**end**, **if**, **then**, **else**, **while**, **do**, **or**, **and**, **div**, **not**.

# Análise Léxica

- 
- sistemas comuns não dispõem de caracteres em negrito, sendo necessário algum tipo de representação. O mais freqüente é adotar caracteres comuns, resultando representações como *begin* e *end* (ou *BEGIN* e *END*). As seqüências de caracteres símbolos compostos em negrito são chamados de *palavras-chaves*.

A substituição dos símbolos em negrito sugerida acima pode acarretar certos problemas, dificultando a distinção, por exemplo, entre a palavra-chave **begin** e o identificador *begin*.

Sugestão p/ isso (adotada em muitas implementações): tratar as palavras-chaves como palavras reservadas que não podem ser usadas como identificadores.



# Análise Léxica

**Razões que levam uma separação da análise de programas em duas partes [KOWALTOWSKI, 83]:**

- **análise léxica:** fica encarregada de isolar as palavras-chaves, símbolos especiais, símbolos especiais compostos, identificadores e constantes, transformando-os em códigos convenientes;
- **análise sintática propriamente dita:** utiliza os resultados da análise léxica toda vez que necessita de mais de um símbolo terminal da cadeia de entrada.



# Análise Léxica

---

**A separação das análises léxicas e sintática traz várias vantagens [KOWALTOWSKI, 83]:**

- toda parte referente à representação dos terminais está concentrada numa única rotina do compilador, tornando mais simples as modificações da representação;
- as regras de formação dos átomos são mais simples do que o resto da sintaxe, permitindo o uso de técnicas de análise mais simples e mais eficiente;
- certos problemas criados pelas idiossincrasias das linguagens de programação, como o uso de palavras-chaves como identificadores, podem ser resolvidos pelo analisador léxico, mantendo a clareza conceitual do analisador sintático;



# Análise Léxica

---

- uma parte apreciável do tempo de compilação corresponde à análise léxica. A sua implementação como uma rotina separada do compilador facilita a introdução de certas otimizações, ou o uso de linguagens de montagem, quando o resto do compilador está escrito numa linguagem de alto nível. Muitos sistemas possuem instruções de máquina especialmente apropriadas para realizar análise léxica. Neste caso, a parte do compilador que depende do sistema utilizado poderá estar concentrada numa única rotina.