IV.2 – Aspectos Léxicos Convencionais

- **Classes de símbolos**
  - **Genéricos**
    - Token único para diferentes sequências de caracteres
    - Sequências possuem uma lei de formação bem definida
    - Podem possuir limitação de tamanho e/ou valor
    - Possuem valor semântico – o token possui uma informação adicional: a representação ou o valor da sequência reconhecida
    - **Exemplos:**
      - Constantes
        - numéricas (inteira, real, ...)
        - literais (cadeias ou sequências de caracteres)
      - Identificadores
        - Identificadores convencionais
  - **Específicos**
    - Token específico para cada símbolo (formado por 1 ou mais caracteres)
    - Não possuem valor semântico
    - **Exemplos:**
      - Símbolos especiais
        - simples, duplos, triplos , ...
        - pontuação, operadores, separadores ...
      - palavras reservadas
        - casos especiais de identificadores
  - **Sem valor sintático**
    - Não constituem token
    - Devem ser reconhecidos, porém ignorados
    - **Exemplos:**
      - Comentários
      - Espaços em branco
      - Caracteres de controle

IV.3 – Especificação dos aspectos Léxicos

- **Informal – descrição textual**
- **Formal – GR, AF e/ou ER**

Escolha do Mecanismo de Especificação

- **Depende da forma de implementação :**
  - **Quando o Analisador Léxico é gerado automaticamente**
    - **O gerador usado determina o mecanismo de especificação (geralmente ER ou variações)**
    - **Exemplos de Geradores:**
      - **LEX, GALS, Micro-LEX, GALEX**
  - **Quando o Analisador Léxico é programado manualmente**
    - **Qualquer mecanismo**
    - **Especificação usada como documentação**
    - **Normalmente usa-se:**
      - **ER para notação/especificação**
      - **AF como base para reconhecimento**
      - **Informal (textual) para exceções**

IV.4 – Analisador Léxico

IV.4.1 - Funções

- Ler o programa fonte
- Agrupar caracteres em itens léxicos (tokens)
  - **Identificadores / Palavras Reservadas**
  - **Constantes (numéricas e literais)**
  - **Símbolos especiais (pontuação, operadores, etc)**
- Ignorar "brancos", comentários e car. de controle
- Detectar e diagnosticar erros léxicos
  - **Símbolos (caracteres) inválidos**
  - **Identificadores, constantes e literais mal formados**
  - **Tamanho/valor inválido de constantes**
- Exemplo:

Programa Fonte	Tokens Reconhecidos	
program exemplo;	program	1 - PR
	exemplo	2 - ID
	;	3 - SEsp
var a, b : integer;	var	4 - PR
	a	2 - ID
	,	5 - SEsp
...	...	...
begin	begin	6 - PR
(* Inicio do programa *)		
b := 2.5 + a;	b	2 –ID
escreva ('tot = ', B);	:=	7 – SEsp duplo
...	2.5	8 – Cte Real
	...	...
end.	end	9 – PR
	.	10 - SEsp

IV.4.2 - Estratégias de Implementação do Analisador Léxico

- **Procedimento Independente**
  - Analisa o programa fonte inteiro
  - Gera arquivo (lista) de TOKENS
  - Constitui um PASSO do processo de Compilação
- **Função do Analisador Sintático**
  - A cada ativação, reconhece um símbolo e retorna:
    - o token correspondente
    - a representação literal – lexeme – quando o token é genérico
    - localização no programa fonte (linha, coluna)
- **Vantagens x Desvantagens**

IV.5 – Especificação Léxica da LSI-182

- **Identificadores**
  - **caracteres válidos : letras, dígitos, "@" e "_"**
  - **regras de formação:**
    - **começa com letra ou "@"**
    - **não pode terminar com "_" e "@"**
    - **não possui caracteres especiais "_" e "@"consecutivos (ou seja, não possui "@@", "__", "@_" e "_@")**
    - **não possui limite de tamanho**

- **Palavras reservadas**
  - **casos especiais de identificadores**

    - **programa, var, caracter, cadeia, procedimento, inicio, fim, inteiro, booleano, funcao, se, entao, senao, leia, escreva, ou, e, nao, falso, verdadeiro, de, faca, real, vetor, enquanto**

      OBS.: lista a ser atualizada quando da esp. sintática da LSI-182

- **Constantes numéricas**
  - Inteiras e reais sem sinal (com e sem parte exponencial)
    - **Reais na forma .25, 2.5 e 25.**
    - **Expoente composto por: "E" ou "e", sinal opcional ("+" ou "-") e pelo menos 2 digitos: Exemplos: 2.5e-10, 3E10, 123.e+99**
    - **fornecer tokens distintos para constantes inteiras e para constantes reais**

- **Constantes literais**
  - Usar ‘ (caracter aspa) como delimitador
  - Sem limite de tamanho
  - No meio de um literal, o caracter ‘ (aspa) deve ser representado por duas aspas simples justapostas
    Ex. ‘pato d’’agua’
  - Permitir continuação em outra linha
  - Literal não fechado – erro léxico
  - Literais podem conter quaisquer caracteres (mesmo os caracteres inválidos para outros fins)

- **Comentários de linha**
  - Começa com “#”
  - Termina no final da linha

- **Comentários de bloco**
  - notação: qualquer sequência de caracteres entre os delimitadores /* e */ - pode conter “*” e “/” no interior do comentário, mas não “*/”.
  - não analisar sequências de caracteres internas
  - sem limite de tamanho
  - comentário não fechado = erro léxico

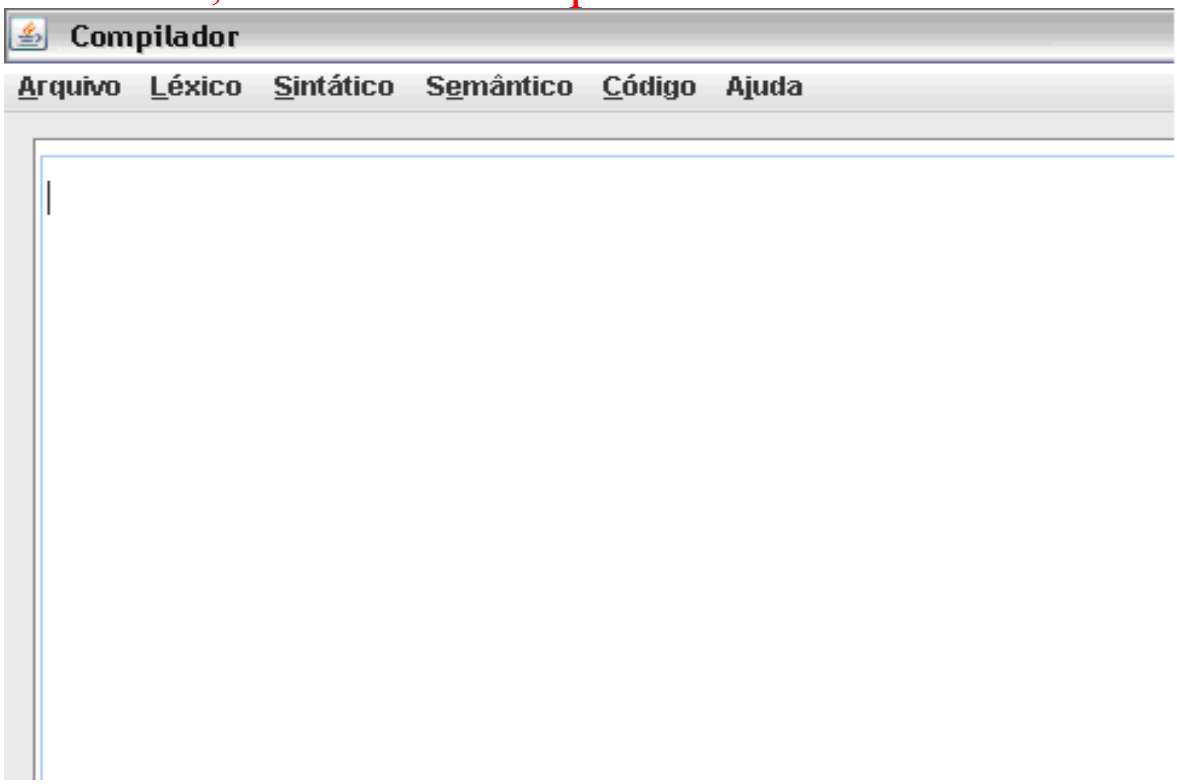
- **Símbolos especiais**
  (lista a ser atualizada de acordo com a especificação sintática)
  - Token específico para cada símbolo
    - **Simples: ; , . > < = ( ) [ ] + - * / :**
    - **Duplos: := .. <> <= >=**

IV.6 – Implementação do Anal. Léxico

- **Especificação Formal / Implementação**
  - Especificar os aspectos Léxicos da LSI-182 usando ER (de acordo com as regras do GALS)
  - Uso do Gerador Automático (GALS)
  - Estratégia de Implementação – Livre escolha
    - procedimento independente
    - função do Analisador Sintático
- **Linguagem para implementação**
  - livre escolha
    - GALS: Java, C++ e Delphi
  - interface gráfica
- **Equipes :** 2 alunos
- **Prazo de entrega**
  - junto com o analisador sintático (a ser definido)
- **Leitura complementar**
  - capítulo sobre análise léxica de um dos livros indicados na bibliografia da disciplina.

IV.7 – Implementação do Compilador

- Criar um Módulo de Integração, usando Interface Gráfica, conforme exemplo abaixo:



- **Observações gerais:**
  - **Permitir a edição, leitura e gravação de programas (permitir a extensão .lsi ou .txt)**
  - **Permitir a ativação independente de cada analisador**
  - **Fornecer mensagens de erro em uma janela separada**
  - **Posicionar o cursor na posição do token que causou o erro**
  - **Substituir as mensagens de erro padrão do gerador por mensagens personalizadas adequadas**
  - **Documentar o programa fonte adequadamente**
  - **Registrar data/autor em todos os módulos/classes**

LEX – Gerador de Analisadores Léxicos
(Lesk & Schmith, 1975)

- **Linguagem de especificação:**

  Definições
  %%
  Regras de tradução
  %%
  Procedimento auxiliares

- **Exemplo de uma especificação:**

```
letra A | B | C | ... | Z   ou [A-Z]
digito     0 | 1 | 2 | ... | 9   ou [0-9]
%%
=      return (1)
:=     return (2)
(      return (3)
       …
       …
       …
letra {letra | digito}  {Se epal-reservada (id, token)
                         então return (token)
                         senao insere-ts(id); return(20)});

dígito {digito}   {insere-tc (num); return (21)}
%%
epal-reservada (...)
insere-ts (...)
insere-tc (...)
```

GALS

(Gesser, 2002 CCO - UFSC)