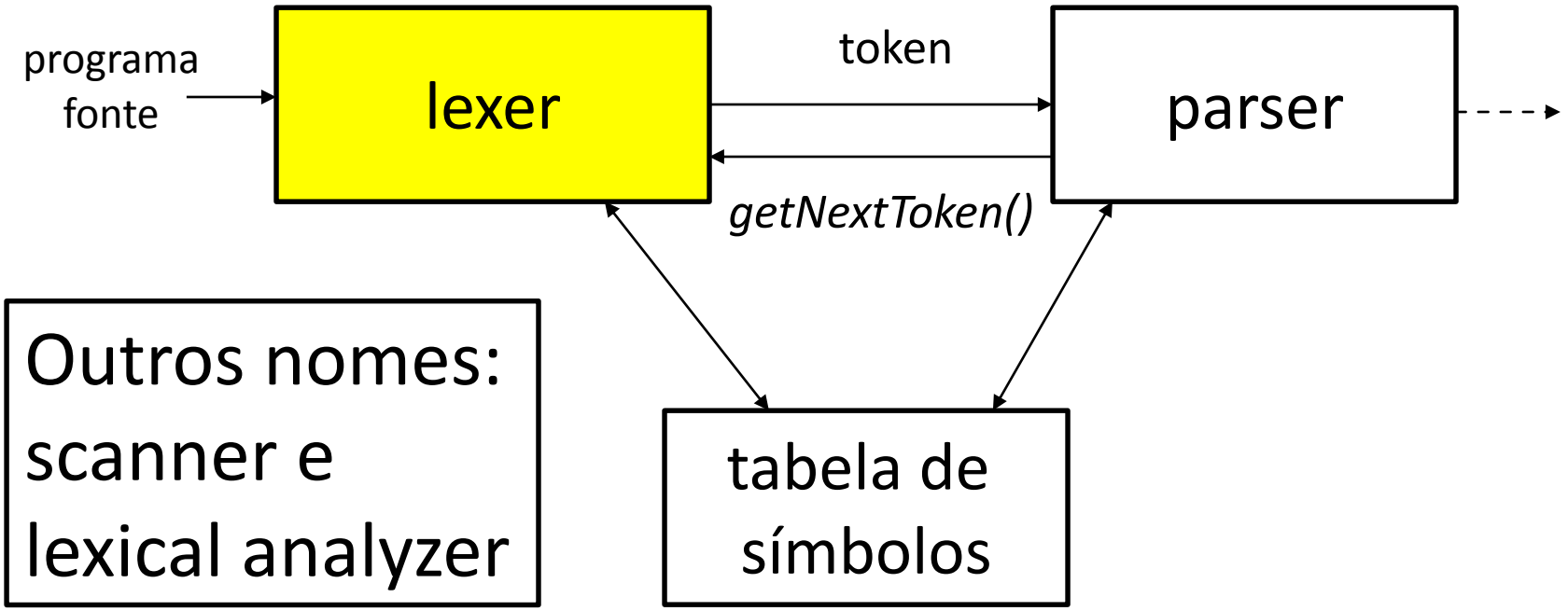
ANÁLISE LÉXICA

Analisador léxico



Outros nomes:
scanner e
lexical analyzer

Tokens, Padrões e Lexemas

- *Tokens*: par com o nome do token e atributos opcionais
- *Padrão*: descrição dos possíveis lexemas associados a um tipo de token
- *Lexema*: sequência de caracteres que casam com o padrão de um tipo de token.

Exemplo

- Para a entrada: “var1 = 23”;
  - Primeiro **token** da entrada: <ID, “var1”>
  - Para este token
    - **Padrão**: “seq. de caracteres seguida por dígito”
    - ID (para identificador) é o tipo do token
    - **Lexema** “var1”

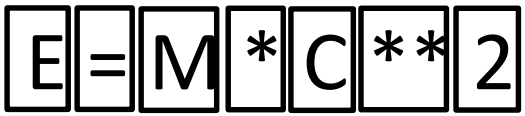
Observações

- Existem conjuntos de strings na entrada que geram o mesmo tipo de token
  - Exemplo: var1, xyz2 são tokens do mesmo tipo ID
- Símbolos terminais de uma gramática correspondem a tokens
  - Exemplo: Palavras reservadas, operadores, identificadores, constantes, parenteses, etc.

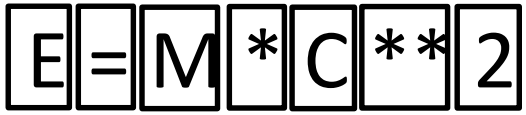
Exemplo

```
E = M * C ** 2
```

Exemplo



Exemplo



<**id**, apontador p/posição de E na tabela de símbolos>
<**assign_op**,>
<**id**, apontador p/posição de M na tabela de símbolos>
<**mult_op**,>
<**id**, apontador p/posição de C na tabela de símbolos>
<**exp_op**,>
<**num**, valor 2>

- Distingue-se instâncias de um token pelos seus atributos
- Usa-se tabela de símbolos para guardar informações auxiliares sobre tokens. E.g., tipo e escopo de identificadores

Expressões regulares

- Usada na especificação dos Tokens
- Expressividade suficiente na prática

Operadores

- * Zero ou mais instâncias
- + Uma ou mais instâncias
- ? Zero ou uma instância
- […] classes de caracteres
  - Exemplos: [A-Za-z][A-Za-z0-9]*
- A | B Instância de A ou instância de B
- AB Instância de A seguida de instância de B

Identificadores em Pascal

letter = [A- Za-z]
digit = [0-9]
id = **letter** (**letter** | **digit**)*

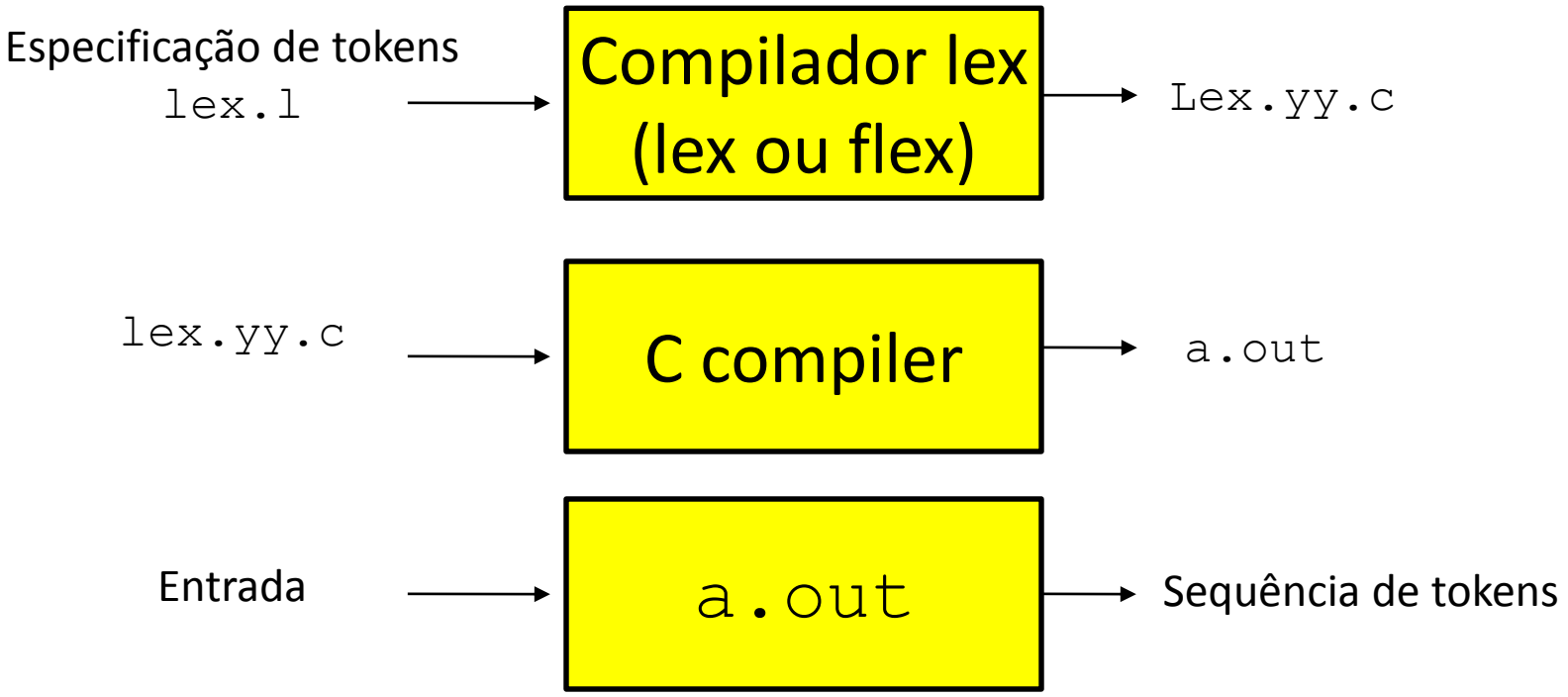
Números em Pascal

digits = **digit digit***
opt_fraction = . **digits** | ε
opt_exponent = (E (+|-|ε) **digits**) | ε
num = **digits opt_fraction opt_exponent**

Implementação do lexer

- Expressões regulares (ER) como entrada
- Ferramentas geram autômatos reconhecedores para ERs
- Exemplos: Lex, Flex, JLex, Alex, etc.

Lex (para C)


Especificação de tokens
lex.l
Compilador lex
(lex ou flex)
Lex.yy.c
lex.yy.c
C compiler
a.out
Entrada
a.out
Sequência de tokens

Descrição em Lex - estrutura

Declarações – variáveis, constantes, defs.regulares

%%

regras de tradução – expr. regulares e ações em C

    Padrão   { ação }

%%

procedimentos auxiliares

Descrição em Lex - exemplo

```
%% declarações
%%

delim       [ \t\n]
ws          {delim}+
letter      [A-Za-z]
digit       [0-9]
id          {letter}({letter}|{digit})*
number      {digit}+(\.{digit}+)?
                    (E[+\-]?{digit}+)?
…
```

Descrição em Lex – exemplo (cont.)

```
%% regras de tradução
%%

{ws}        {/* no action and no return */}
if          {return(IF);}
then        {return(THEN);}
else        {return(ELSE);}
{id}        {yylval=install_id();return(ID);}
{number}    {yylval=install_num();
            return(NUMBER);}
"<"         {yylval = LT; return(RELOP);}
"<="        {yylval = LE; return(RELOP);}
…
```

Descrição em Lex – exemplo (cont.)

```
%% funções auxiliares
%%

int install_id() {
  Copia lexema para a tabela de símbolos.
  Primeiro caracter do lexema é apontado
  pela variável yytext e o comprimento é
  definido pela variável yylength.
}

int install_num() { … }
```