Linguagens de Programação

Aula 4

Na aula anterior

- Ambiguidade
- Análise léxica

Na aula de hoje

- Análise léxica – implementação
- Gramática e reconhecedores
- Variáveis:
  - Nomes
  - Vinculações (binding)
  - Verificação de tipo e
  - Escopo

Projeto de um analisador léxico

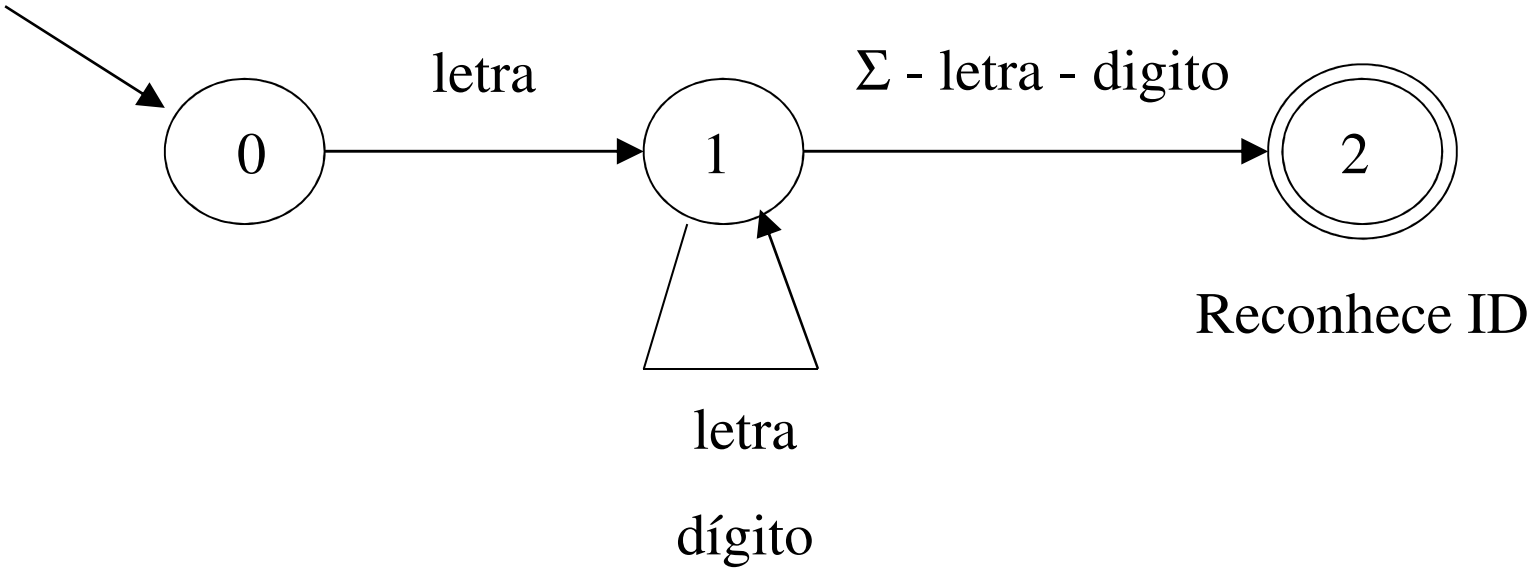
- Existem duas formas básicas para implementar os autômatos: usando a tabela de transição ou **diretamente em código**

Estilo de implementação diretamente em código

- Cada ***token*** listado é codificado em um número natural
- Deve haver uma variável para controlar o **estado** corrente do autômato e outro para indicar o estado de **partida** do autômato em uso
- Uma função **falhar** é usada para desviar o estado corrente para um outro autômato no caso de um estado não reconhecer uma letra

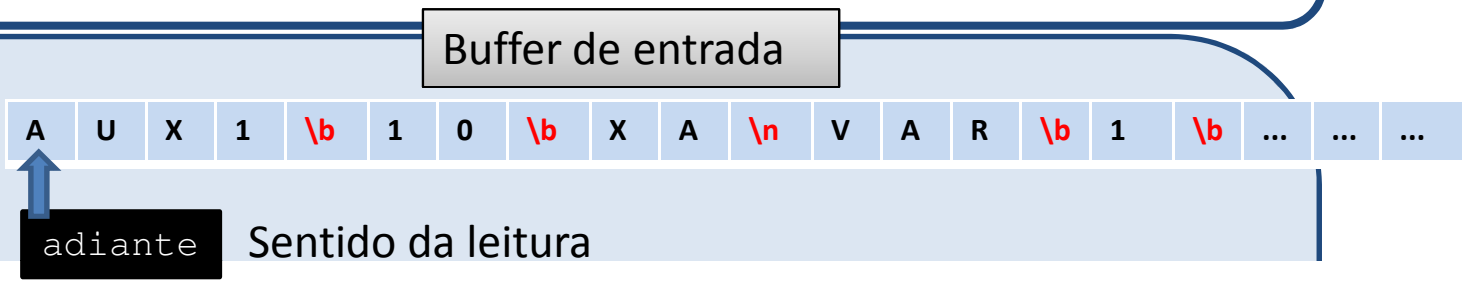
Estilo de implementação

- Cada estado é analisado individualmente em uma estrutura do tipo **switch...case**



Estilo de implementação

```
//exemplo de arquivo
AUX1  10 XA
VAR  1 ...
```

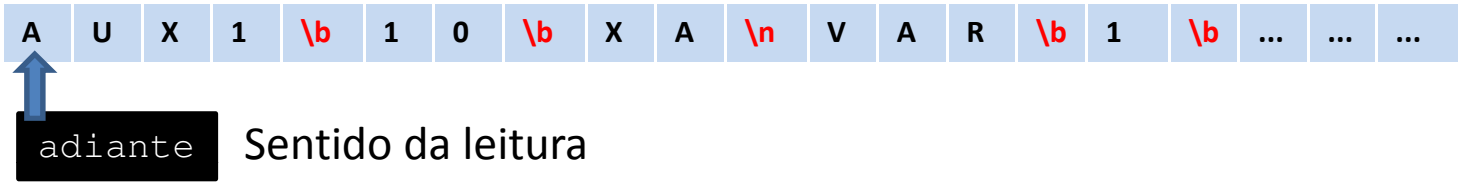


1) A entrada de dados é obtida a partir de um arquivo texto.
2) Cria-se uma estrutura (buffer) para armazená-los em memória (principal)
3) Cada caractere é avaliado individualmente → são separados por “\b” espaço, “\t” tabulação ou “\n” quebra de linha

Estilo de implementação

AUX1 10 XA
VAR 1 ...

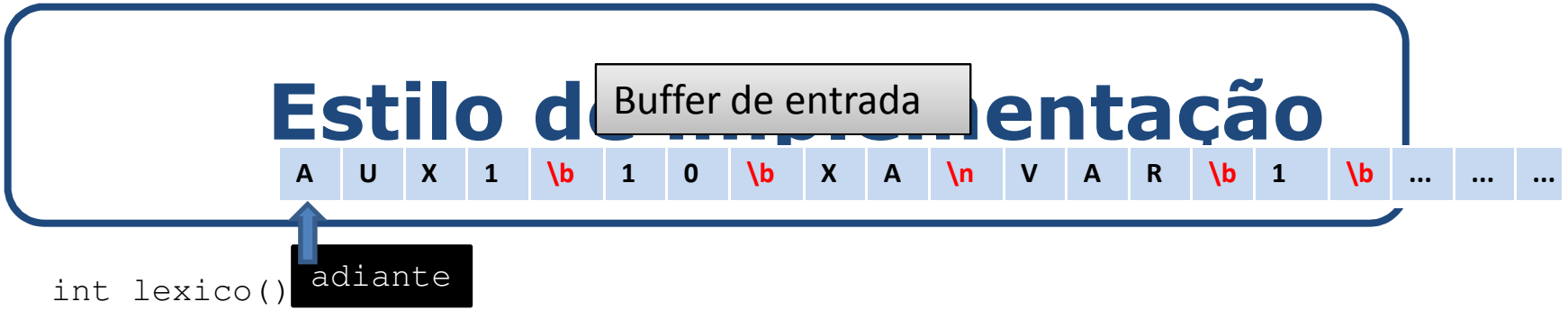
Buffer de entrada



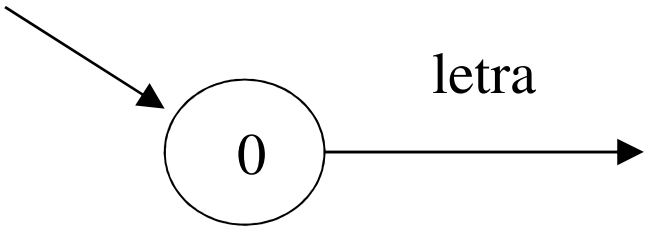
➔O analisador léxico deverá ler e classificar as lexemas considerando o AF e retornar:

1. AUX1 → ID
2. 10 → ID inválido
3. XA → ID
4. VAR → ID
5. 1 → ID inválido

Estilo de implementação




```
int lexico()
{
   while (1)
   {
      switch (estado)
      {
         case 0: c= proximo_caracter();
                  if (isalpha(c))
                  {
                     estado= 1;
                     adiante++;
                  }
                  else
                  {
                      falhar(); //identificador inválido
                  }
                  break;
         …
      }
}
```



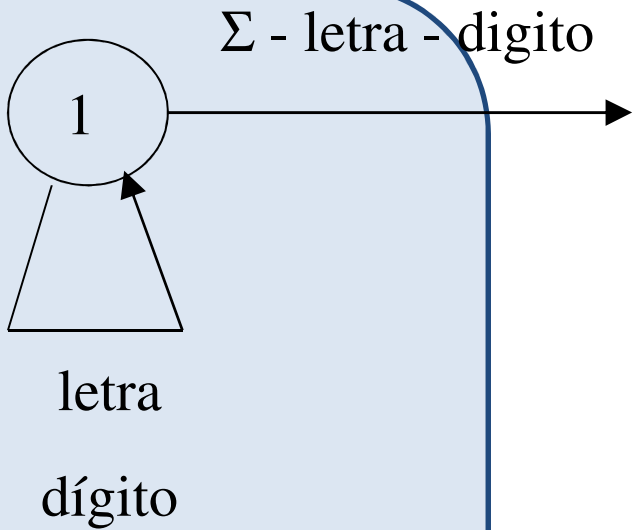
Estilo de implementação

Buffer de entrada

A	U	X	1	\b	1	0	\b	X	A	\n	V	A	R	\b	1	\b	…	…	…

adiante

```
        …
        case 1: c= proximo_caracter();
                    if (isalpha(c) || isdigit(c))
                    {
                       estado= 1;
                       adiante++;
                    }
                    else
                    {


                    if ((c == '\n') || (c == '\t') || (c == '\b'))
                           estado= 2;
                           else falhar();
                    }
                    break;
        …
    }
}
```



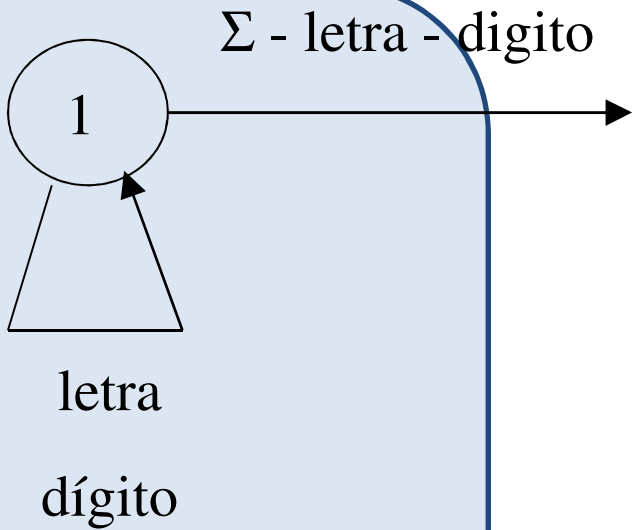
Estilo de implementação

Buffer de entrada

A	U	X	1	\b	1	0	\b	X	A	\n	V	A	R	\b	1	\b	…	…	…

adiante

```
        …
        case 1: c= proximo_caracter();
                    if (isalpha(c) || isdigit(c))
                    {
                       estado= 1;
                       adiante++;
                    }
                    else
                    {


                    if ((c == '\n') || (c == '\t') || (c == '\b'))
                           estado= 2;
                           else falhar();
                    }
                    break;
        …
    }
}
```



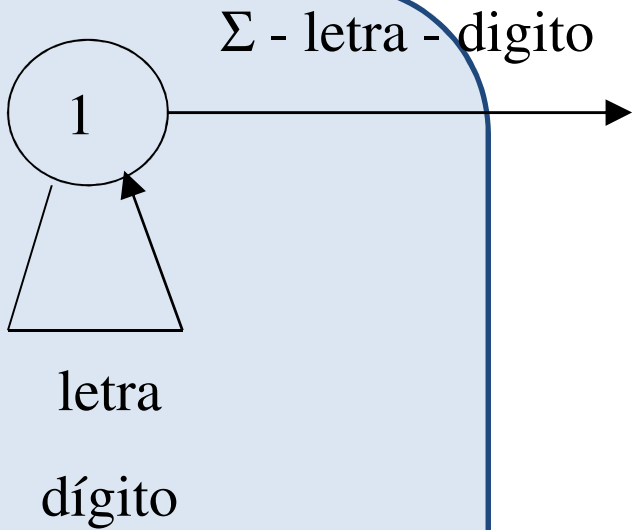
Estilo de implementação

Buffer de entrada

A	U	X	1	\b	1	0	\b	X	A	\n	V	A	R	\b	1	\b	…	…	…

adiante

```
        …
        case 1: c= proximo_caracter();
                    if (isalpha(c) || isdigit(c))
                    {
                       estado= 1;
                       adiante++;
                    }
                    else
                    {

                    if ((c == '\n') || (c == '\t') || (c == '\b'))
                           estado= 2;
                           else falhar();
                    }
                    break;
        …
    }
}
```



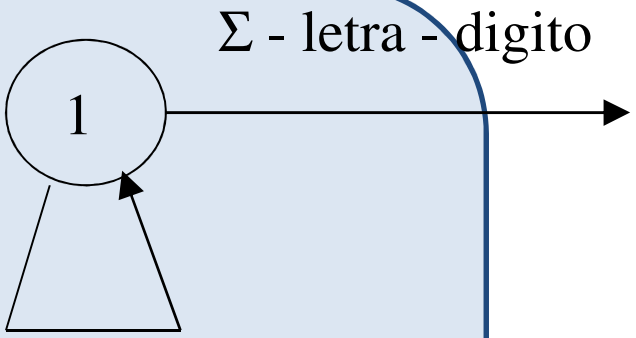
Estilo de implementação

Buffer de entrada

A	U	X	1	\b	1	0	\b	X	A	\n	V	A	R	\b	1	\b	…	…	…

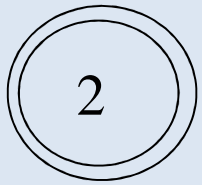
adiante

```
    …
    case 1: c= proximo_caracter();
                if (isalpha(c) || isdigit(c))
                {
                   estado= 1;
                   adiante++;
                }
                else
                {

                if ((c == '\n') || (c == '\t') || (c == '\b'))
                       estado= 2;
                       else falhar();
                }
                break;
    …
  }
}
```



Estilo de implementação

```
    …
            case 2: estado= 0;
                        partida= 0;
                        return ID;
                        break;

          }
} }
```



Reconhece ID

Exercícios

- Referentes ao capítulo 3
  - 1, 6, 10, 11, 12 (pág 184)
  - 1, 2, 3, 6, 7, 8, 10, 11, 12 (págs. 185 e 186)
- Referentes ao capítulo 4
  - 1, 2, 3, 4, 5, 6 (pág 223)