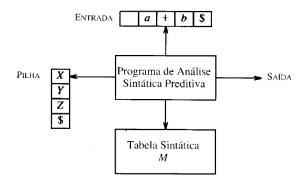
Parsing Descendente (Parte 02)

### Sumário

- 1 Parsing Descendente Não-Recursivo
  - Analisador Preditivo Não-Recursivo
- Função FOLLOWFunção FOLLOW
- Tabela SintáticaTabela Sintática
- 4 Gramática **LL**(*k*)
  - Gramática **LL**(k)
- 5 Tratamento de Erros
  - Tratamento de Erros

#### Modelo de Analisador Preditivo Não-Recursivo



00000

## Parsing Preditivo Não-Recursivo

```
Faça ip apontar para o primeiro símbolo da entrada w$
```

### Código de Análise Preditiva Não-Recursiva

Faça ip apontar para o primeiro símbolo da entrada w\$ repetir

```
Seja X o símbolo do topo da pilha e a o símbolo apontado por ip
até que X = '\$'
```

Parsing Desc. Não-Rec.

### Código de Análise Preditiva Não-Recursiva

Faça *ip* apontar para o primeiro símbolo da entrada *w*\$ **repetir** 

Seja X o símbolo do topo da pilha e a o símbolo apontado por ip se X for um terminal ou '\$' então

Parsing Desc. Não-Rec.

### Código de Análise Preditiva Não-Recursiva

```
Faça ip apontar para o primeiro símbolo da entrada w$
repetir
    Seja X o símbolo do topo da pilha e a o símbolo apontado por ip
    se X for um terminal ou '$' então
         se X = a então
até que X = '\$'
```

Parsing Desc. Não-Rec.

### Código de Análise Preditiva Não-Recursiva

```
Faça ip apontar para o primeiro símbolo da entrada w$
repetir
    Seja X o símbolo do topo da pilha e a o símbolo apontado por ip
    se X for um terminal ou '$' então
         se X = a então
              Remover X da pilha e avançar ip
até que X = '\$'
```

Parsing Desc. Não-Rec.

### Código de Análise Preditiva Não-Recursiva

```
Faça ip apontar para o primeiro símbolo da entrada w$
repetir
    Seja X o símbolo do topo da pilha e a o símbolo apontado por ip
    se X for um terminal ou '$' então
         se X = a então
              Remover X da pilha e avançar ip
         senão erro()
até que X = '\$'
```

Parsing Desc. Não-Rec.

### Código de Análise Preditiva Não-Recursiva

```
Faça ip apontar para o primeiro símbolo da entrada w$
repetir
    Seja X o símbolo do topo da pilha e a o símbolo apontado por ip
    se X for um terminal ou '$' então
         se X = a então
             Remover X da pilha e avançar ip
         senão erro()
                                            // X é um não-terminal
    senão
até que X = '\$'
```

Parsing Desc. Não-Rec.

```
Faça ip apontar para o primeiro símbolo da entrada w$
repetir
    Seja X o símbolo do topo da pilha e a o símbolo apontado por ip
    se X for um terminal ou '$' então
         se X = a então
              Remover X da pilha e avançar ip
         senão erro()
                                              // X é um não-terminal
    senão
         se M[X, a] = 'X \rightarrow Y_1 Y_2 \dots Y_k' então
até que X = '$'
```

```
Faça ip apontar para o primeiro símbolo da entrada w$
repetir
    Seja X o símbolo do topo da pilha e a o símbolo apontado por ip
    se X for um terminal ou '$' então
         se X = a então
              Remover X da pilha e avançar ip
         senão erro()
                                              // X é um não-terminal
    senão
         se M[X, a] = 'X \rightarrow Y_1 Y_2 \dots Y_k' então
              Remover X da pilha
até que X = '$'
```

```
Faça ip apontar para o primeiro símbolo da entrada w$
repetir
    Seja X o símbolo do topo da pilha e a o símbolo apontado por ip
    se X for um terminal ou '$' então
         se X = a então
              Remover X da pilha e avançar ip
         senão erro()
                                               // X é um não-terminal
    senão
         se M[X, a] = 'X \rightarrow Y_1 Y_2 \dots Y_k' então
              Remover X da pilha
              se M[X, a] \neq X \rightarrow X então
até que X = '$'
```

LL(k)

## Parsing Preditivo Não-Recursivo

```
Faça ip apontar para o primeiro símbolo da entrada w$
repetir
    Seja X o símbolo do topo da pilha e a o símbolo apontado por ip
    se X for um terminal ou '$' então
         se X = a então
              Remover X da pilha e avançar ip
         senão erro()
                                               // X é um não-terminal
    senão
         se M[X, a] = 'X \rightarrow Y_1 Y_2 \dots Y_k' então
              Remover X da pilha
              se M[X, a] \neq X \rightarrow X então
                   Empilhar Y_1 Y_2 \dots Y_k (com Y_1 no topo)
até que X = '$'
```

00000

```
Faça ip apontar para o primeiro símbolo da entrada w$
repetir
    Seja X o símbolo do topo da pilha e a o símbolo apontado por ip
    se X for um terminal ou '$' então
          se X = a então
               Remover X da pilha e avançar ip
          senão erro()
                                                 // X é um não-terminal
    senão
          se M[X, a] = 'X \rightarrow Y_1 Y_2 \dots Y_k' então
               Remover X da pilha
               se M[X, a] \neq X \rightarrow X então
                    Empilhar Y_1 Y_2 \dots Y_k (com Y_1 no topo)
               Escrever a regra X \rightarrow Y_1 Y_2 \dots Y_k na saída
até que X = '$'
```

```
Faça ip apontar para o primeiro símbolo da entrada w$
repetir
    Seja X o símbolo do topo da pilha e a o símbolo apontado por ip
    se X for um terminal ou '$' então
          se X = a então
               Remover X da pilha e avançar ip
          senão erro()
                                                // X é um não-terminal
    senão
          se M[X, a] = 'X \rightarrow Y_1 Y_2 \dots Y_k' então
               Remover X da pilha
               se M[X, a] \neq X \rightarrow X então
                    Empilhar 'Y_1Y_2...Y_k' (com Y_1 no topo)
               Escrever a regra X \rightarrow Y_1 Y_2 \dots Y_k na saída
          senão erro()
até que X = '$'
```

#### Parser Preditivo Não-Recursivo – Exemplo

Seja GLC 
$$G: E \rightarrow E + T \mid T$$
  
 $T \rightarrow T * F \mid F$   
 $F \rightarrow id \mid (E)$ 

Eliminando as recursões à esquerda, tem-se:

$$\begin{array}{lll} \textit{G'} \colon \textit{E} & \rightarrow \textit{T} \; \textit{E'} \\ \textit{E'} & \rightarrow + \textit{T} \; \textit{E'} \; \mid \; \lambda \\ \textit{T} & \rightarrow \textit{F} \; \textit{T'} \\ \textit{T'} & \rightarrow * \textit{F} \; \textit{T'} \; \mid \; \lambda \\ \textit{F} & \rightarrow \text{id} \; \mid \; (\textit{E}) \end{array}$$

```
FIRST(E) = FIRST(TE') = FIRST(T) = FIRST(FT') = FIRST(F) = \{ \ '(', i) \} 
FIRST(E') = FIRST(+TE') \cup FIRST(\lambda) = \{ \ '+', \lambda \} 
FIRST(T') = FIRST(+FT') \cup FIRST(\lambda) = \{ \ '+', \lambda \}
```

Parsing Desc. Não-Rec.

00000

Tratamento de Erros

LL(k)

## Parsing Preditivo Não-Recursivo

#### Parser Preditivo Não-Recursivo – Exemplo

Seja GLC 
$$G: E \rightarrow E + T \mid T$$
  
 $T \rightarrow T * F \mid F$   
 $F \rightarrow id \mid (E)$ 

Eliminando as recursões à esquerda, tem-se:

FIRST(
$$E$$
) = FIRST( $TE'$ ) = FIRST( $T$ ) = FIRST( $F$ ) = FIRST( $F$ ) = { '(', id)}  
FIRST( $E'$ ) = FIRST( $+TE'$ )  $\cup$  FIRST( $\lambda$ ) = { '+',  $\lambda$ }  
FIRST( $T'$ ) = FIRST( $*FT'$ )  $\cup$  FIRST( $\lambda$ ) = { '\*',  $\lambda$ }

### Parser Preditivo Não-Recursivo – Exemplo (cont.)

Para GLC 
$$G'$$
:  $E \rightarrow T E'$ 

$$E' \rightarrow + T E' \mid \lambda$$

$$T \rightarrow F T'$$

$$T' \rightarrow * F T' \mid \lambda$$

$$F \rightarrow \text{id} \mid (E)$$

#### Considere a seguinte tabela sintática:

Não-		Símbolo de Entrada					
Terminal	id	+	(	)	\$		
E	E  o TE'			$E \rightarrow TE'$			
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E'  o \lambda$	
T	T  o FT'			$T \rightarrow FT'$			
T'		$T'  o \lambda$	$T' \rightarrow *FT'$		$T'  o \lambda$	$T'  o \lambda$	
F	$ extcolor{F}  ightarrow  extcolor{id}$			$F \rightarrow (E)$			

-									
	Não-		Símbolo de Entrada						
	Terminal	id	+	*	(	)	\$		
	Ε	$3E \rightarrow TE'$			$E \rightarrow TE'$				
	E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$		
	T	$T \rightarrow FT'$			$T \rightarrow FT'$				
	T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$		
	F	$F \rightarrow id$			$F \rightarrow (E)$				

F	$F \rightarrow id$			$F \rightarrow (E)$
Simulação:	Casamento	Pilha	Entrada	Ação
,		E\$	id + id * id\$	
		TE'\$	id + id * id\$	Saída: $E  o TE'$
		FT'E'\$	id + id * id\$	Saída: $T  o FT'$
		id <i>T'E'</i> \$	id + id * id\$	Saída: $F \rightarrow id$
		T'E'\$	+ id * id\$	⇒ Casamento: id
		E'\$	+ id * id\$	Saída: $T'  o \lambda$
		+TE'\$	+ id * id\$	Saída: $E'  o + TE'$
		TE'\$	id ∗ id\$	⇒ Casamento: +
		FT'E'\$	id ∗ id\$	Saída: $T \rightarrow FT'$
		id <i>T'E'</i> \$	id * id\$	Saída: $F  o id$
		T'E'\$	* id\$	⇒ Casamento: id
		*FT'E'\$	* id\$	Saída: $T' \rightarrow *FT'$
		FT'E'\$	id\$	⇒ Casamento: ∗
		id <i>T' E'</i> \$	id\$	Saída: $F  o id$
		T'E'\$	\$	⇒ Casamento: id
		E'\$	\$	Saída: $T'  o \lambda$
		\$	\$	Saída: $E'  ightarrow \lambda$

Não-		Símbolo de Entrada						
Terminal	id	+	*	(	)	\$		
E	$3E \rightarrow TE'$			$E \rightarrow TE'$				
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$		
T	$T \rightarrow FT'$			$T \rightarrow FT'$				
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$		
F	$F \rightarrow id$			$F \rightarrow (E)$				

F	$F \rightarrow id$				$F \rightarrow (E)$
Simulação:	Casame	Casamento		Entrada	Ação
•			E\$		
			TE'\$		Saída: $E \rightarrow TE'$
			FT'E'\$		Saída: $T \to FT'$
			id <i>T'E'</i> \$		Saída: $F  o id$
			T'E'\$	+ id * id\$	⇒ Casamento: id
			E'\$	+ id * id\$	Saída: $T' \to \lambda$
			+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
			TE'\$	id * id\$	⇒ Casamento: +
			FT'E'\$	10	Saída: $T \to FT'$
			id <i>T'E'</i> \$		Saída: $F  o id$
			T'E'\$	* id\$	⇒ Casamento: id
			*FT'E'\$	* id\$	Saída: $T' \to *FT'$
			FT'E'\$		⇒ Casamento: *
			id <i>T' E'</i> \$		Saída: $F  o id$
			T'E'\$		
			E'\$	S	Saída: $T' \to \lambda$

Não-		Símbolo de Entrada					
Terminal	id	+	*	(	)	\$	
E	$3E \rightarrow TE'$			$E \rightarrow TE'$			
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$	
T	$T \rightarrow FT'$			$T \rightarrow FT'$			
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$	
F	$F \rightarrow id$			$F \rightarrow (E)$			

Simul	JACAO.
JIIIIU	iacao.

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E \rightarrow TE'$
	FT' E'\$	id + id * id\$	Saída: $T  o FT'$
	id <i>T'E'</i> \$	id + id * id\$	Saída: $F  o id$
	T'E'\$	+ id * id\$	⇒ Casamento: id
	E'\$	+ id * id\$	Saída: $T'  o \lambda$
	+ <i>TE</i> '\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
	TE'\$	id * id\$	⇒ Casamento: +
	FT'E'\$	id * id\$	Saída: $T  o FT'$
	id <i>T'E'</i> \$	id * id\$	Saída: $F  o  ext{id}$
	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T'  o *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	idT'E'\$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
id + id * id	\$	\$	Saída: $E'  o \lambda$

					,			
Não-		Símbolo de Entrada						
Terminal	id	+	*	(	)	\$		
E	$3E \rightarrow TE'$			$E \rightarrow TE'$				
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$		
T	$T \rightarrow FT'$			$T \rightarrow FT'$				
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$		
F	$F \rightarrow id$			$F \rightarrow (E)$				

Simu	

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T  o FT'$
	id <i>T'E'</i> \$	id + id * id\$	Saída: $F  o id$
	T'E'\$	+ id * id\$	⇒ Casamento: id
	E'\$	+ id * id\$	Saída: $T'  o \lambda$
	+ <i>TE</i> '\$	+ id * id\$	Saída: $E'  o + TE'$
	TE'\$	id * id\$	⇒ Casamento: +
	FT'E'\$	id * id\$	Saída: $T  o FT'$
	id <i>T'E'</i> \$	id * id\$	Saída: $F  o id$
	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T'  o *FT'$
	FT'E'\$	id\$	⇒ Casamento: ∗
	idT'E'\$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  o \lambda$

Não-		Símbolo de Entrada						
Terminal	id	+	*	(	)	\$		
E	$3E \rightarrow TE'$			$E \rightarrow TE'$				
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$		
T	$T \rightarrow FT'$			$T \rightarrow FT'$				
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$		
F	$F \rightarrow id$			$F \rightarrow (E)$				

$\sim$	~
Simil	acão:
Jiiiiu	uçuo.

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E \rightarrow TE'$
	FT'E'\$	id + id * id\$	Saída: $T  o FT'$
	id <i>T'E'</i> \$	id + id * id\$	Saída: $F  o id$
	T'E'\$	+ id * id\$	⇒ Casamento: id
	E'\$	+ id * id\$	Saída: $T'  o \lambda$
	+ <i>TE</i> '\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
	TE'\$	id * id\$	⇒ Casamento: +
	FT'E'\$	id * id\$	Saída: $T  o FT'$
	idT'E'\$	id * id\$	Saída: $F  o id$
	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T' \to *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	idT'E'\$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  o \lambda$

Não-		Símbolo de Entrada					
Terminal	id	+	*	(	)	\$	
E	$3E \rightarrow TE'$			$E \rightarrow TE'$			
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$	
T	$T \rightarrow FT'$			$T \rightarrow FT'$			
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$	
F	$F \rightarrow id$			$F \rightarrow (E)$			

Simu	Jacau.
Jiiiiu	açao.

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T  o FT'$
	id <i>T'E'</i> \$	id + id * id\$	Saída: $F  o id$
	T'E'\$	+ id * id\$	⇒ Casamento: id
	E'\$	+ id * id\$	Saída: $T'  o \lambda$
	+ <i>TE</i> '\$	+ id * id\$	Saída: $E'  o + TE'$
	TE'\$	id * id\$	$\Rightarrow$ Casamento: $+$
	FT'E'\$	id * id\$	Saída: $T  o FT'$
	id <i>T'E'</i> \$	id * id\$	Saída: $F  o \mathbf{id}$
	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T'  o *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	idT'E'\$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  ightarrow \lambda$

#### Parser Preditivo Não-Recursivo – Exemplo (cont.)

								,	
Não-		Símbolo de Entrada							
Terminal	id		+	*		(		)	\$
E	$3E \rightarrow TE'$				$E \rightarrow$	TE'			
E'		E'	$\rightarrow +TE'$				E' -	$\rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$				$T \rightarrow$	·FT′			
T'		7	$\Gamma' \to \lambda$	$T' \rightarrow *FT'$			T' -	$\rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$				$F \rightarrow$	· (E)			
Simulação	Casame	nto	Pilha	Entrada		Aca	ăo.		

Jiiilulação.	Casamento	l IIIIa	Liitiada	Ação
•		E\$	id + id * id\$	
		TE'\$	id + id * id\$	Saída: $E  o TE'$
		FT'E'\$	id + id * id\$	Saída: $T  o FT'$
		id <i>T' E'</i> \$	id + id * id\$	
		T'E'\$	+ id * id\$	
		E'\$	+ id * id\$	
		+TE'\$	+ id * id\$	
		TE'\$	id * id\$	
		FT'E'\$	id * id\$	
		idT'E'\$	id * id\$	
		T'E'\$	* id\$	
		*FT'E'\$	* id\$	
		FT'E'\$	id\$	

Não-		Símbolo de Entrada					
Terminal	id	+	*	(	)	\$	
E	$3E \rightarrow TE'$			$E \rightarrow TE'$			
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$	
T	$T \rightarrow FT'$			$T \rightarrow FT'$			
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$	
F	$F \rightarrow id$			$F \rightarrow (E)$			

Simil	lação:
JIIIIU	ıacau.
	,
•	.uyuv.

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T  o FT'$
	id <i>T' E'</i> \$	id + id * id\$	Saída: $F \rightarrow id$
	T'E'\$	+ id * id\$	⇒ Casamento: id
	E'\$	+ id * id\$	Saída: $T'  ightarrow \lambda$
	+ <i>TE</i> '\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
	TE'\$	id * id\$	$\Rightarrow$ Casamento: +
	FT'E'\$	id * id\$	Saída: $T  o FT'$
	id <i>T'E'</i> \$	id * id\$	Saída: $F  o id$
	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T'  o *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	idT'E'\$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	5	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  ightarrow \lambda$

#### Parser Preditivo Não-Recursivo – Exemplo (cont.)

Não-		Símbolo de Entrada					
Terminal	id	+	*	(	)	\$	
E	$3E \rightarrow TE'$			$E \rightarrow TE'$			
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$	
T	$T \rightarrow FT'$			$T \rightarrow FT'$			
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$	
F	$F \rightarrow id$			$F \rightarrow (E)$			

Simulação:	Casamento	Pilha	Entrada	
•		E\$	id + id * id\$	
		TE'\$	id + id * id\$	Saída:
		/-/-		

	1 = 3	10 + 10 * 103	Salda: E → IE
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id <i>T'E'</i> \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
	E'\$	+ id * id\$	Saída: $T'  o \lambda$
	+ <i>TE</i> '\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
	TE'\$	id * id\$	⇒ Casamento: +
	FT'E'\$	id * id\$	Saída: $T  o FT'$
	id <i>T'E'</i> \$	id * id\$	Saída: $F  o id$
	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T' \to *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	idT'E'\$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'S	\$	Saída: $T' \rightarrow \lambda$

Ação F → TF'

Não-		Símbolo de Entrada				
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

F	$F \rightarrow id$					$F \rightarrow (E)$	
Simulação	Casame	nto	Pilha		ntrada	Aç	ão
•			E\$		+ id ∗ id\$		
			TE'\$		+ id * id\$	Saída: E -	$\rightarrow TE'$
			FT'E'\$		+ id * id\$	Saída: T -	$\rightarrow$ FT $'$
			id <i>T' E'</i> \$		+ id * id\$	Saída: F -	→ id
	id		T'E'\$		+ id * id\$	⇒ Casame	ento: id
			E'\$	-	+ id * id\$	Saída: $T'$	
						Saída: E'	
						⇒ Casamo	
				l		Saída: T -	
						Saída: F -	
						⇒ Casamo	
				l		Saída: T'	
		(c		İ		⇒ Casamo	
		(c				Saída: F -	
		* id				⇒ Casame	
		* id				Saída: T'	

Não-		Símbolo de Entrada					
Terminal	id	+	*	(	)	\$	
E	$3E \rightarrow TE'$			$E \rightarrow TE'$			
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$	
T	$T \rightarrow FT'$			$T \rightarrow FT'$			
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$	
F	$F \rightarrow id$			$F \rightarrow (E)$			

Simula	

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id T' E' \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
	+ <i>TE</i> '\$	+ id * id\$	Saída: $E'  o + TE'$
	TE'\$	id * id\$	⇒ Casamento: +
	FT'E'\$	id * id\$	Saída: $T  o FT'$
	idT'E'\$	id * id\$	Saída: $F  o id$
	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T'  o *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	idT'E'\$	id\$	Saída: $F  o \mathbf{id}$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  o \lambda$

Não-		Símbolo de Entrada				
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simu	
	7

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id T' E' \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
	TE'\$	id * id\$	⇒ Casamento: +
	FT'E'\$	id * id\$	Saída: $T  o FT'$
	idT'E'\$	id * id\$	Saída: $F  o id$
	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T' \to *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	idT'E'\$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  ightarrow \lambda$

Não-		Símbolo de Entrada					
Terminal	id	+	*	(	)	\$	
E	$3E \rightarrow TE'$			$E \rightarrow TE'$			
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$	
T	$T \rightarrow FT'$			$T \rightarrow FT'$			
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$	
F	$F \rightarrow id$			$F \rightarrow (E)$			

Simu	ação:

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id T' E' \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
	TE'\$	id * id\$	$\Rightarrow$ Casamento: +
	FT'E'\$	id * id\$	Saída: $T  o FT'$
	idT'E'\$	id * id\$	Saída: $F  o \mathbf{id}$
	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T'  o *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	idT'E'\$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	S	\$	Saída: $E' \rightarrow \lambda$

Não-	Símbolo de Entrada					
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simul	Incha.
JIIIIu	iacao.

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E \rightarrow TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id T' E' \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id * id\$	⇒ Casamento: +
	FT'E'\$	id * id\$	Saída: $T  o FT'$
	idT'E'\$	id * id\$	Saída: $F \rightarrow id$
	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T'  o *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	idT'E'\$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  ightarrow \lambda$

#### Parser Preditivo Não-Recursivo – Exemplo (cont.)

Não-	Símbolo de Entrada					
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

A ====

Simu	laçao:

Casamento	Pilha	Entrada	Açao
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T  o FT'$
	id <i>T' E'</i> \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id * id\$	⇒ Casamento: +
id +	FT' E'\$	id * id\$	Saída: $T  o FT'$
id +	id <i>T' E'</i> \$	id * id\$	Saída: $F  o \mathbf{id}$
id + id	T'E'\$	* id\$	⇒ Casamento: id
id + id	*FT'E'\$	* id\$	Saída: $T'  o *FT'$
id + id *	FT'E'\$	id\$	⇒ Casamento: *
id + id *	idT'E'\$	id\$	Saída: $F  o \mathbf{id}$
id + id * id	T'E'\$	\$	⇒ Casamento: id
id + id * id	E'\$	5	Saída: $T'  ightarrow \lambda$
id + id * id	\$	\$	Saída: $E'  ightarrow \lambda$

Não-		Símbolo de Entrada				
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simu	iacao.

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E \rightarrow TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id <i>T'E'</i> \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id * id\$	⇒ Casamento: +
id +	FT'E'\$	id * id\$	Saída: $T \rightarrow FT'$
	id <i>T'E'</i> \$	id * id\$	Saída: $F  o id$
	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T' \to *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	id <i>T'E'</i> \$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  o \lambda$

Não-	Símbolo de Entrada					
			Simbolo de En	trada		
Terminal	id	+	*	(	)	\$
Ε	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simul	

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id <i>T' E'</i> \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id * id\$	⇒ Casamento: +
id +	FT'E'\$	id * id\$	Saída: $T \rightarrow FT'$
id +	id <i>T' E'</i> \$	id * id\$	Saída: $F \rightarrow id$
id + id	T'E'\$		⇒ Casamento: id
id + id	*FT'E'\$		Saída: $T' \to *FT'$
id + id *	FT'E'\$		⇒ Casamento: *
id + id *	idT'E'\$		Saída: $F  o id$
id + id * id	T'E'\$		⇒ Casamento: id
id + id * id	E'\$		Saída: $T'  o \lambda$
id + id * id	\$		Saída: $E'  ightarrow \lambda$

Não-	Símbolo de Entrada					
			Simbolo de En	trada		
Terminal	id	+	*	(	)	\$
Ε	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simul	ISCSO.
Jiiiiu	açao.

Casamen	to Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E \rightarrow TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id T' E' \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id * id\$	⇒ Casamento: +
id +	FT'E'\$	id * id\$	Saída: $T \rightarrow FT'$
id +	id T' E'\$	id * id\$	Saída: $F \rightarrow id$
	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T'  o *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	id <i>T'E'</i> \$	id\$	Saída: $F  o \mathbf{id}$
	id T'E'\$	\$	⇒ Casamento: id
	id E'\$	\$	Saída: $T'  o \lambda$
	id \$	\$	Saída: $E'  o \lambda$

Não-	Símbolo de Entrada					
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simu	

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id T' E' \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id * id\$	⇒ Casamento: +
id +	FT'E'\$	id * id\$	Saída: $T \rightarrow FT'$
id +	id <i>T'E'</i> \$	id * id\$	Saída: $F \rightarrow id$
id + id	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T'  o *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	id <i>T'E'</i> \$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  o \lambda$

Não-	Símbolo de Entrada					
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simu	acan.
JIIIIu	iacao.

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  ightarrow TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id <i>T' E'</i> \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id * id\$	⇒ Casamento: +
id +	FT'E'\$	id * id\$	Saída: $T  o FT'$
id +	id <i>T' E'</i> \$	id * id\$	Saída: $F \rightarrow id$
id + id	T'E'\$	* id\$	⇒ Casamento: id
	*FT'E'\$	* id\$	Saída: $T'  o *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	idT'E'\$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  ightarrow \lambda$

				· · · · · ·	,	
Não-	Símbolo de Entrada					
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simu	acao.
	,

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id <i>T' E'</i> \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id * id\$	⇒ Casamento: +
id +	FT'E'\$	id * id\$	Saída: $T \rightarrow FT'$
id +	id <i>T' E'</i> \$	id * id\$	Saída: $F \rightarrow id$
id + id	T'E'\$	* id\$	⇒ Casamento: id
id + id	*FT'E'\$	* id\$	Saída: $T' \rightarrow *FT'$
	FT'E'\$	id\$	⇒ Casamento: *
	id <i>T' E'</i> \$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  o \lambda$

Não-			Símbolo de En	trada		
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simu	iacao.
	7

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E \rightarrow TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id T' E' \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id * id\$	⇒ Casamento: +
id +	FT'E'\$	id * id\$	Saída: $T \rightarrow FT'$
id +	id <i>T'E'</i> \$	id * id\$	Saída: $F \rightarrow id$
id + id	T'E'\$	* id\$	⇒ Casamento: id
id + id	*FT'E'\$	* id\$	Saída: $T' \rightarrow *FT'$
id + id *	FT'E'\$	id\$	⇒ Casamento: *
	id <i>T'E'</i> \$	id\$	Saída: $F  o id$
	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
id + id * id	\$	\$	Saída: $E'  o \lambda$

					,	
Não-	Símbolo de Entrada					
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simulação:
7

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T  o FT'$
	id <i>T' E'</i> \$	id + id * id\$	Saída: $F  o \mathbf{id}$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E'  ightarrow + TE'$
id +	TE'\$	id * id\$	$\Rightarrow$ Casamento: +
id +	FT'E'\$	id * id\$	Saída: $T  o FT'$
id +	id <i>T' E'</i> \$	id * id\$	Saída: $F  o \mathbf{id}$
id + id	T'E'\$	* id\$	⇒ Casamento: id
id + id	*FT'E'\$	* id\$	Saída: $T'  o *FT'$
id + id *	FT'E'\$	id\$	⇒ Casamento: *
	id <i>T' E'</i> \$	id\$	Saída: $F  o \mathbf{id}$
	T'E'\$	\$	$\Rightarrow$ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  ightarrow \lambda$

Não-			Símbolo de En	trada		
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simul	

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id <i>T' E'</i> \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id * id\$	$\Rightarrow$ Casamento: +
id +	FT'E'\$	id * id\$	Saída: $T \rightarrow FT'$
id +	id <i>T' E'</i> \$	id * id\$	Saída: $F \rightarrow id$
id + id	T'E'\$	* id\$	⇒ Casamento: id
id + id	*FT'E'\$	* id\$	Saída: $T' \rightarrow *FT'$
id + id *	FT'E'\$	id\$	⇒ Casamento: *
id + id *	id <i>T' E'</i> \$	id\$	Saída: $F \rightarrow id$
	T'E'\$	\$	$\Rightarrow$ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  o \lambda$

Não-	Símbolo de Entrada					
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simul	laçao:

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id T' E' \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id * id\$	$\Rightarrow$ Casamento: +
id +	FT'E'\$	id * id\$	Saída: $T \rightarrow FT'$
id +	id T' E' \$	id * id\$	Saída: $F \rightarrow id$
id + id	T'E'\$	* id\$	⇒ Casamento: id
id + id	*FT'E'\$	* id\$	Saída: $T' \rightarrow *FT'$
id + id *	FT'E'\$	id\$	⇒ Casamento: *
id + id *	id <i>T' E'</i> \$	id\$	Saída: $F \rightarrow id$
id + id * id	T'E'\$	\$	⇒ Casamento: id
	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  o \lambda$

Não-	Símbolo de Entrada					
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simu	laçao:

Casamento	Pilha	Entrada	Ação	
	E\$	id + id * id\$		
	TE'\$	id + id * id\$	Saída: $E  ightarrow \mathit{TE}'$	
	FT'E'\$	id + id * id\$	Saída: $T  o FT'$	
	id <i>T' E'</i> \$	id + id * id\$	Saída: $F  o \mathbf{id}$	
id	T'E'\$	+ id * id\$	⇒ Casamento: id	
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$	
id	+TE'\$	+ id * id\$	Saída: $E'  ightarrow + TE'$	
id +	TE'\$	id * id\$	$\Rightarrow$ Casamento: +	
id +	FT'E'\$	id * id\$	Saída: $T  o FT'$	
id +	id <i>T' E'</i> \$	id * id\$	Saída: $F  o \mathbf{id}$	
id + id	T'E'\$	* id\$	⇒ Casamento: id	
id + id	*FT'E'\$	* id\$	Saída: $T'  o *FT'$	
id + id *	FT'E'\$	id\$	⇒ Casamento: *	
id + id *	id <i>T' E'</i> \$	id\$	Saída: $F  o \mathbf{id}$	
id + id * id	T'E'\$	\$	⇒ Casamento: id	
	E'\$	\$	Saída: $T'  o \lambda$	
	\$	\$	Saída: $E'  ightarrow \lambda$	

#### Parser Preditivo Não-Recursivo – Exemplo (cont.)

Não-			Símbolo de En	trada		
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simulação:
7

0000

•	, .u				• • (=)
	Casamento		Pilha	Entrada	Ação
			E\$	id + id * id\$	
			TE'\$	id + id * id\$	Saída: $E \rightarrow TE'$
			FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
			id T' E' \$	id + id * id\$	Saída: $F \rightarrow id$
	id		T'E'\$	+ id * id\$	⇒ Casamento: id
	id		E'\$	+ id * id\$	Saída: $T' \rightarrow \lambda$
	id		+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
	id +		TE'\$	id * id\$	⇒ Casamento: +
	id +		FT'E'\$	id * id\$	Saída: $T \rightarrow FT'$
	id +		id T' E'\$	id * id\$	Saída: $F \rightarrow id$
	id + id		T'E'\$	* id\$	⇒ Casamento: id
	id + id		*FT'E'\$	* id\$	Saída: $T' \rightarrow *FT'$
	id + id	*	FT'E'\$	id\$	⇒ Casamento: *
	id + id	*	id <i>T' E'</i> \$	id\$	Saída: $F \rightarrow id$
	id + id	* id	T'E'\$	\$	⇒ Casamento: id
	id + id	* id	E'\$	\$	Saída: $T'  o \lambda$
			\$	\$	Saída: $E' \rightarrow \lambda$

#### Parser Preditivo Não-Recursivo – Exemplo (cont.)

Não-			Símbolo de En	trada		
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simul	Incha.
JIIIIu	iacao.

0000

/ → Iu					$I \rightarrow (L)$	
Casame	nto	Pilha	E	ntrada	Açã	io
		E\$	id -	⊢id ∗id\$		
		TE'\$	id -	⊢ id ∗ id\$	Saída: E -	→ TE′
		FT'E'\$	id -	⊢ id ∗ id\$	Saída: T -	$\rightarrow FT'$
		id <i>T' E'</i> \$	id -	⊢ id ∗ id\$	Saída: F -	→ id
id		T'E'\$	-	⊢ id ∗ id\$	⇒ Casame	
id		E'\$	-	⊢ id ∗ id\$	Saída: T'	$\rightarrow \lambda$
id		+TE'\$	-	⊢ id ∗ id\$	Saída: E'	$\rightarrow +TE'$
id +		TE'\$		id * id\$	⇒ Casame	ento: +
id +		FT'E'\$		id * id\$	Saída: T -	$\rightarrow FT'$
id +		id <i>T' E'</i> \$		id * id\$	Saída: F -	→ id
id + id		T'E'\$		* id\$	⇒ Casame	ento: id
id + id		*FT'E'\$		* id\$	Saída: T'	$\rightarrow *FT'$
id + id *	k	FT'E'\$		id\$	⇒ Casame	ento: *
id + id *	k	id <i>T' E'</i> \$		id\$	Saída: F -	→ id
id + id	∗ id	T'E'\$		\$	⇒ Casame	
id + id	∗ id	E'\$		\$	Saída: T'	$\rightarrow \lambda$
	∗ id	\$		\$	Saída: E'	

#### Parser Preditivo Não-Recursivo – Exemplo (cont.)

Não-	Símbolo de Entrada					
Terminal	id	+	*	(	)	\$
E	$3E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$			$F \rightarrow (E)$		

Simu	
•	

Parsing Desc. Não-Rec.

0000

Casamento	Pilha	Entrada	Ação
	E\$	id + id * id\$	
	TE'\$	id + id * id\$	Saída: $E  o TE'$
	FT'E'\$	id + id * id\$	Saída: $T \rightarrow FT'$
	id <i>T'E'</i> \$	id + id * id\$	Saída: $F \rightarrow id$
id	T'E'\$	+ id * id\$	⇒ Casamento: id
id	E'\$	+ id * id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id * id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id * id\$	⇒ Casamento: +
id +	FT'E'\$	id * id\$	Saída: $T \rightarrow FT'$
id +	id <i>T'E'</i> \$	id * id\$	Saída: $F \rightarrow id$
id + id	T'E'\$	* id\$	⇒ Casamento: id
id + id	*FT'E'\$	* id\$	Saída: $T' \rightarrow *FT'$
id + id *	FT'E'\$	id\$	⇒ Casamento: *
id + id *	id <i>T' E'</i> \$	id\$	Saída: $F \rightarrow id$
id + id * id	T'E'\$	\$	⇒ Casamento: id
id + id * id	E'\$	\$	Saída: $T'  o \lambda$
id + id * id	\$	\$	Saída: $E'  o \lambda$
	id id id + id + id + id + id id + id id + id id + id * id	E\$   TE'\$   FT'E'\$   id T'E'\$   id E'\$   id + TE'\$   id + FT'E'\$   id + FT'E'\$   id + id   T'E'\$   id + id   E'\$	$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$

#### **FOLLOW**

Quando um não-terminal X puder gerar  $\lambda$  (ou  $\lambda \in \mathsf{FIRST}(X)$ ), deve-se determinar quais terminais podem aparecer depois dele em uma forma sentencial.

#### **FOLLOW**

Quando um não-terminal X puder gerar  $\lambda$  (ou  $\lambda \in FIRST(X)$ ), deve-se determinar quais terminais podem aparecer depois dele em uma forma sentencial.

#### Define-se **FOLLOW** da seguinte forma:

#### **FOLLOW**

Parsing Desc. Não-Rec.

Quando um não-terminal X puder gerar  $\lambda$  (ou  $\lambda \in \mathsf{FIRST}(X)$ ), deve-se determinar quais terminais podem aparecer depois dele em uma forma sentencial.

#### Define-se **FOLLOW** da seguinte forma:

- Se S é o símbolo de partida, então coloca-se \$ em FOLLOW(S), sendo \$ um marcador de fim da entrada
- Para toda regra da forma  $A \to \alpha X \beta$ , tudo em FIRST( $\beta$ ) exceto  $\lambda$  é colocado em FOLLOW(X)
- Para toda regra da forma  $A \to \alpha X$  ou  $A \to \alpha X \beta$  em que  $\lambda \in \mathsf{FIRST}(\beta)$  tudo em  $\mathsf{FOLLOW}(A)$  é colocado em  $\mathsf{FOLLOW}(X)$

#### **FOLLOW**

Quando um não-terminal X puder gerar  $\lambda$  (ou  $\lambda \in \mathsf{FIRST}(X)$ ), deve-se determinar quais terminais podem aparecer depois dele em uma forma sentencial.

#### Define-se **FOLLOW** da seguinte forma:

- Se S é o símbolo de partida, então coloca-se \$ em FOLLOW(S), sendo \$ um marcador de fim da entrada
- Para toda regra da forma  $A \to \alpha X \beta$ , tudo em FIRST( $\beta$ ) exceto  $\lambda$  é colocado em FOLLOW(X)
- Para toda regra da forma  $A \to \alpha X$  ou  $A \to \alpha X \beta$  em que  $\lambda \in \mathsf{FIRST}(\beta)$  tudo em  $\mathsf{FOLLOW}(A)$  é colocado em  $\mathsf{FOLLOW}(X)$

#### **FOLLOW**

Parsing Desc. Não-Rec.

Quando um não-terminal X puder gerar  $\lambda$  (ou  $\lambda \in \mathsf{FIRST}(X)$ ), deve-se determinar quais terminais podem aparecer depois dele em uma forma sentencial.

#### Define-se **FOLLOW** da seguinte forma:

- Se S é o símbolo de partida, então coloca-se \$ em FOLLOW(S), sendo \$ um marcador de fim da entrada
- Para toda regra da forma  $A \to \alpha X \beta$ , tudo em FIRST( $\beta$ ) exceto  $\lambda$  é colocado em FOLLOW(X)
- Para toda regra da forma  $A \to \alpha X$  ou  $A \to \alpha X \beta$  em que  $\lambda \in \mathsf{FIRST}(\beta)$ , tudo em  $\mathsf{FOLLOW}(A)$  é colocado em  $\mathsf{FOLLOW}(X)$

#### **FOLLOW**

Quando um não-terminal X puder gerar  $\lambda$  (ou  $\lambda \in FIRST(X)$ ), deve-se determinar quais terminais podem aparecer depois dele em uma forma sentencial.

#### Define-se **FOLLOW** da seguinte forma:

- Se S é o símbolo de partida, então coloca-se \$ em FOLLOW(S), sendo \$ um marcador de fim da entrada
- Para toda regra da forma A → αXβ, tudo em FIRST(β) exceto λ é colocado em FOLLOW(X)
- Para toda regra da forma  $A \to \alpha X$  ou  $A \to \alpha X \beta$  em que  $\lambda \in \mathsf{FIRST}(\beta)$ , tudo em  $\mathsf{FOLLOW}(A)$  é colocado em  $\mathsf{FOLLOW}(X)$

Parsing Desc. Não-Rec.

```
FOLLOW - Exemplo
Seja GLC G: E \rightarrow T E'
                  E' \rightarrow + T E' \mid \lambda
                   T \rightarrow F T'
                   T' \rightarrow * F T' \mid \lambda
                  F \rightarrow id \mid (E)
```

### FOLLOW - Exemplo

Seja GLC 
$$G: E \rightarrow T E'$$

$$E' \rightarrow + T E' \mid \lambda$$

$$T \rightarrow F T'$$

$$T' \rightarrow * F T' \mid \lambda$$

$$F \rightarrow \text{id} \mid (E)$$

$$\begin{aligned} &\mathsf{FIRST}(E) = \mathsf{FIRST}(TE') = \mathsf{FIRST}(T) = \mathsf{FIRST}(FT') = \mathsf{FIRST}(F) = \{\ '(',\mathsf{id}) \\ &\mathsf{FIRST}(E') = \mathsf{FIRST}(+TE') \cup \mathsf{FIRST}(\lambda) = \{\ '+',\lambda\} \\ &\mathsf{FIRST}(T') = \mathsf{FIRST}(*FT') \cup \mathsf{FIRST}(\lambda) = \{\ '*',\lambda\} \end{aligned}$$

### FOLLOW - Exemplo

Seja GLC 
$$G: E \rightarrow T E'$$

$$E' \rightarrow + T E' \mid \lambda$$

$$T \rightarrow F T'$$

$$T' \rightarrow * F T' \mid \lambda$$

$$F \rightarrow \operatorname{id} \mid (E)$$

$$\begin{aligned} \mathsf{FIRST}(E) &= \mathsf{FIRST}(TE') = \mathsf{FIRST}(T) = \mathsf{FIRST}(FT') = \mathsf{FIRST}(F) = \{\ '(',\mathsf{id}) \\ \mathsf{FIRST}(E') &= \mathsf{FIRST}(+TE') \cup \mathsf{FIRST}(\lambda) = \{\ '+',\lambda\} \\ \mathsf{FIRST}(T') &= \mathsf{FIRST}(*FT') \cup \mathsf{FIRST}(\lambda) = \{\ '*',\lambda\} \end{aligned}$$

$$FOLLOW(E) = \{\$\}$$

### FOLLOW – Exemplo

Seja GLC 
$$G: E \rightarrow T E'$$

$$E' \rightarrow + T E' \mid \lambda$$

$$T \rightarrow F T'$$

$$T' \rightarrow * F T' \mid \lambda$$

$$F \rightarrow \operatorname{id} \mid (E)$$

$$\begin{aligned} & \mathsf{FIRST}(E) &= \mathsf{FIRST}(TE') = \mathsf{FIRST}(T) = \mathsf{FIRST}(FT') = \mathsf{FIRST}(F) = \{ \ '(', \mathsf{id}) \\ & \mathsf{FIRST}(E') = \mathsf{FIRST}(+TE') \cup \mathsf{FIRST}(\lambda) = \{ \ '+', \lambda \} \\ & \mathsf{FIRST}(T') = \mathsf{FIRST}(*FT') \cup \mathsf{FIRST}(\lambda) = \{ \ '*', \lambda \} \end{aligned}$$

**FOLLOW**(E) = { ')', \$}

### FOLLOW - Exemplo

Seja GLC 
$$G: E \rightarrow T E'$$

$$E' \rightarrow + T E' \mid \lambda$$

$$T \rightarrow F T'$$

$$T' \rightarrow * F T' \mid \lambda$$

$$F \rightarrow \text{id} \mid (E)$$

$$\begin{split} & \mathsf{FIRST}(E) = \mathsf{FIRST}(TE') = \mathsf{FIRST}(T) = \mathsf{FIRST}(FT') = \mathsf{FIRST}(F) = \{ \ '(', \mathsf{id}) \\ & \mathsf{FIRST}(E') = \mathsf{FIRST}(+TE') \cup \mathsf{FIRST}(\lambda) = \{ \ '+', \lambda \} \\ & \mathsf{FIRST}(T') = \mathsf{FIRST}(*FT') \cup \mathsf{FIRST}(\lambda) = \{ \ '*', \lambda \} \end{split}$$

$$FOLLOW(E) = FOLLOW(E') = \{ ')', \$ \}$$

# FOLLOW - Exemplo

 $FOLLOW(T) = \{ '+' \}$ 

Seja GLC 
$$G: E \rightarrow T E'$$
  
 $E' \rightarrow + T E' \mid \lambda$   
 $T \rightarrow F T'$   
 $T' \rightarrow * F T' \mid \lambda$   
 $F \rightarrow \operatorname{id} \mid (E)$ 

$$\begin{aligned} & \mathsf{FIRST}(E) = \mathsf{FIRST}(TE') = \mathsf{FIRST}(T) = \mathsf{FIRST}(FT') = \mathsf{FIRST}(F) = \{\ '(',\mathsf{id}) \\ & \mathsf{FIRST}(E') = \mathsf{FIRST}(+TE') \cup \mathsf{FIRST}(\lambda) = \{\ '+',\lambda\} \\ & \mathsf{FIRST}(T') = \mathsf{FIRST}(*FT') \cup \mathsf{FIRST}(\lambda) = \{\ '*',\lambda\} \\ & \mathsf{FOLLOW}(E) = \mathsf{FOLLOW}(E') = \{\ ')',\$ \} \end{aligned}$$

# FOLLOW – Exemplo

```
Seja GLC G: E \rightarrow T E'

E' \rightarrow + T E' \mid \lambda

T \rightarrow F T'

T' \rightarrow * F T' \mid \lambda

F \rightarrow \operatorname{id} \mid (E)
```

$$\begin{aligned} & \mathsf{FIRST}(E) &= \mathsf{FIRST}(TE') = \mathsf{FIRST}(T) = \mathsf{FIRST}(FT') = \mathsf{FIRST}(F) = \{\ '(',\mathsf{id}) \\ & \mathsf{FIRST}(E') = \mathsf{FIRST}(+TE') \cup \mathsf{FIRST}(\lambda) = \{\ '+',\lambda\} \\ & \mathsf{FIRST}(T') = \mathsf{FIRST}(*FT') \cup \mathsf{FIRST}(\lambda) = \{\ '*',\lambda\} \\ & \mathsf{FOLLOW}(E) = \mathsf{FOLLOW}(E') = \{\ ')',\$ \} \\ & \mathsf{FOLLOW}(T) = \{\ '+',\ ')',\$ \} \end{aligned}$$

### FOLLOW - Exemplo

Seja GLC 
$$G: E \rightarrow T E'$$
  
 $E' \rightarrow + T E' \mid \lambda$   
 $T \rightarrow F T'$   
 $T' \rightarrow * F T' \mid \lambda$   
 $F \rightarrow \operatorname{id} \mid (E)$ 

$$\begin{aligned} & \mathsf{FIRST}(E) &= \mathsf{FIRST}(TE') = \mathsf{FIRST}(T) = \mathsf{FIRST}(FT') = \mathsf{FIRST}(F) = \{\ '(',\mathsf{id}) \\ & \mathsf{FIRST}(E') = \mathsf{FIRST}(+TE') \cup \mathsf{FIRST}(\lambda) = \{\ '+',\lambda\} \\ & \mathsf{FIRST}(T') = \mathsf{FIRST}(*FT') \cup \mathsf{FIRST}(\lambda) = \{\ '*',\lambda\} \end{aligned}$$

$$FOLLOW(E) = FOLLOW(E') = \{ ')', \$ \}$$

$$FOLLOW(T) = FOLLOW(T') = \{ '+', ')', \$ \}$$

LL(k)

## Função FOLLOW

```
FOLLOW - Exemplo
Seja GLC G: E \rightarrow T E'
                E' \rightarrow + T E' \mid \lambda
                T \rightarrow F T'
                T' \rightarrow *FT' \mid \lambda
                F \rightarrow id \mid (E)
FIRST(E) = FIRST(TE') = FIRST(T) = FIRST(FT') = FIRST(F) = \{ '(', id) \}
FIRST(E') = FIRST(+TE') \cup FIRST(\lambda) = \{ '+', \lambda \}
FIRST(T') = FIRST(*FT') \cup FIRST(\lambda) = \{ \ '*', \lambda \}
FOLLOW(E) = FOLLOW(E') = \{ ')', \$ \}
FOLLOW(T) = FOLLOW(T') = \{ '+', ')', \$ \}
FOLLOW(F) = \{ '*' \}
```

Tratamento de Erros

## Função FOLLOW

Parsing Desc. Não-Rec.

```
FOLLOW - Exemplo
Seja GLC G: E \rightarrow T E'
                E' \rightarrow + T E' \mid \lambda
                T \rightarrow F T'
                T' \rightarrow *FT' \mid \lambda
                F \rightarrow id \mid (E)
FIRST(E) = FIRST(TE') = FIRST(T) = FIRST(FT') = FIRST(F) = \{ '(', id) \}
FIRST(E') = FIRST(+TE') \cup FIRST(\lambda) = \{ '+', \lambda \}
FIRST(T') = FIRST(*FT') \cup FIRST(\lambda) = \{ \ '*', \lambda \}
FOLLOW(E) = FOLLOW(E') = \{ ')', \$ \}
FOLLOW(T) = FOLLOW(T') = \{ '+', ')', \$ \}
FOLLOW(F) = { '*', '+', ')', $}
```

### Parsing Não-Recursivo – Tabela Sintática

#### Tabela Sintática Preditiva

A tabela sintática preditiva M é uma matriz bidimensional cuja entrada M[A,a] indica qual regra do não-terminal A deve ser usada para se obter uma derivação mais à esquerda iniciada pelo terminal a.

- Para cada produção  $A \rightarrow \alpha$ , faca:
  - ① Para cada terminal a em **FIRST**( $\alpha$ ), adicionar a regra  $A \to \alpha$  à entrada M[A,a]
  - ② Se  $\lambda \in \mathsf{FIRST}(\alpha)$ , adicionar a regra  $A \to \alpha$  à entrada M[A, b] para cada terminal b em  $\mathsf{FOLLOW}(A)$
  - **③** Se  $\lambda \in \mathsf{FIRST}(\alpha)$  e  $\$ \in \mathsf{FOLLOW}(A)$ , adicionar a regra  $A \to \alpha$  à entrada M[A,\$]

## Parsing Não-Recursivo – Tabela Sintática

#### Tabela Sintática Preditiva

A tabela sintática preditiva M é uma matriz bidimensional cuja entrada M[A,a] indica qual regra do não-terminal A deve ser usada para se obter uma derivação mais à esquerda iniciada pelo terminal a.

- Para cada produção  $A \rightarrow \alpha$ , faca:
  - ① Para cada terminal a em **FIRST**( $\alpha$ ), adicionar a regra  $A \to \alpha$  à entrada M[A,a]
  - ② Se  $\lambda \in \mathsf{FIRST}(\alpha)$ , adicionar a regra  $A \to \alpha$  à entrada M[A, b] para cada terminal b em  $\mathsf{FOLLOW}(A)$
  - **3** Se  $\lambda \in \mathsf{FIRST}(\alpha)$  e  $\$ \in \mathsf{FOLLOW}(A)$ , adicionar a regra  $A \to \alpha$  à entrada M[A,\$]

## Parsing Não-Recursivo - Tabela Sintática

#### Tabela Sintática Preditiva

A tabela sintática preditiva M é uma matriz bidimensional cuja entrada M[A,a] indica qual regra do não-terminal A deve ser usada para se obter uma derivação mais à esquerda iniciada pelo terminal a.

- Para cada produção  $A \rightarrow \alpha$ , faça:
  - ① Para cada terminal a em  $FIRST(\alpha)$ , adicionar a regra  $A \to \alpha$  à entrada M[A,a]
  - ② Se  $\lambda \in \mathsf{FIRST}(\alpha)$ , adicionar a regra  $A \to \alpha$  à entrada M[A, b], para cada terminal b em  $\mathsf{FOLLOW}(A)$
  - **③** Se  $\lambda \in \mathsf{FIRST}(\alpha)$  e  $\$ \in \mathsf{FOLLOW}(A)$ , adicionar a regra  $A \to \alpha$  à entrada M[A,\$]

### Parsing Não-Recursivo – Tabela Sintática

#### Tabela Sintática Preditiva

A tabela sintática preditiva M é uma matriz bidimensional cuja entrada M[A, a] indica qual regra do não-terminal A deve ser usada para se obter uma derivação mais à esquerda iniciada pelo terminal a.

- Para cada produção  $A \rightarrow \alpha$ , faça:
  - **1** Para cada terminal a em **FIRST**( $\alpha$ ), adicionar a regra  $A \rightarrow \alpha$ à entrada M[A, a]

# Parsing Não-Recursivo — Tabela Sintática

#### Tabela Sintática Preditiva

A tabela sintática preditiva M é uma matriz bidimensional cuja entrada M[A,a] indica qual regra do não-terminal A deve ser usada para se obter uma derivação mais à esquerda iniciada pelo terminal a.

- Para cada produção  $A \rightarrow \alpha$ , faça:
  - Para cada terminal a em  $FIRST(\alpha)$ , adicionar a regra  $A \to \alpha$  à entrada M[A, a]
  - **2** Se  $\lambda \in \mathsf{FIRST}(\alpha)$ , adicionar a regra  $A \to \alpha$  à entrada M[A, b], para cada terminal b em  $\mathsf{FOLLOW}(A)$
  - ③ Se  $\lambda \in \mathsf{FIRST}(\alpha)$  e  $\$ \in \mathsf{FOLLOW}(A)$ , adicionar a regra $A \to \alpha$  à entrada M[A,\$]

## Parsing Não-Recursivo - Tabela Sintática

#### Tabela Sintática Preditiva

A tabela sintática preditiva M é uma matriz bidimensional cuja entrada M[A,a] indica qual regra do não-terminal A deve ser usada para se obter uma derivação mais à esquerda iniciada pelo terminal a.

- Para cada produção  $A \rightarrow \alpha$ , faça:
  - Para cada terminal a em **FIRST**( $\alpha$ ), adicionar a regra  $A \rightarrow \alpha$  à entrada M[A, a]
  - **2** Se  $\lambda \in \mathsf{FIRST}(\alpha)$ , adicionar a regra  $A \to \alpha$  à entrada M[A, b], para cada terminal b em  $\mathsf{FOLLOW}(A)$
  - **3** Se  $\lambda \in \mathsf{FIRST}(\alpha)$  e  $\$ \in \mathsf{FOLLOW}(A)$ , adicionar a regra  $A \to \alpha$  à entrada M[A,\$]

## Parsing Não-Recursivo - Tabela Sintática

### Tabela Sintática Preditiva – Exemplo

```
G: E \rightarrow T E' 
E' \rightarrow + T E' \mid \lambda 
T \rightarrow F T' 
T' \rightarrow * F T' \mid \lambda 
F \rightarrow id \mid (E) 
FIRS
```

FIRST(E) = FIRST(TE') = { '(', id)}  
FIRST(E') = FIRST(+TE') 
$$\cup$$
 FIRST( $\lambda$ ) = { '+',  $\lambda$ }  
FIRST( $T$ ) = FIRST( $FT'$ ) = { '(', id)}  
FIRST( $T'$ ) = FIRST(\* $FT'$ )  $\cup$  FIRST( $\lambda$ ) = { '\*',  $\lambda$ }  
FIRST( $F$ ) = FIRST(id)  $\cup$  FIRST(( $E$ )) = { id, '(')}  
FOLLOW( $E$ ) = FOLLOW( $E'$ ) = { ')', \$}  
FOLLOW( $T$ ) = FOLLOW( $T'$ ) = { '+', ')', \$}  
FOLLOW( $T$ ) = { '\*', '+', ')', \$}

#### Tabela Sintática

Parsing Desc. Não-Rec.

## Tabela Sintática Preditiva – Exemplo

```
 \begin{array}{lll} \textit{G:} \ E \ \to \ T \ E' \\ E' \ \to \ + \ T \ E' \ \mid \ \lambda \\ T \ \to \ F \ T' \\ F' \ \to \ * \ F \ T' \ \mid \ \lambda \\ F \ \to \ \mathsf{id} \ \mid \ (E) \\ \end{array} \qquad \begin{array}{lll} \mathsf{FIRST}(E) \ = \ \mathsf{FIRST}(\mathsf{TE'}) \ = \ \{\ '(', \mathsf{id}) \\ \mathsf{FIRST}(E') \ = \ \mathsf{FIRST}(\mathsf{FT'}) \ \cup \ \mathsf{FIRST}(\lambda) \ = \ \{\ '+', \ \lambda\} \\ \mathsf{FIRST}(T') \ = \ \mathsf{FIRST}(\mathsf{FT'}) \ \cup \ \mathsf{FIRST}(\lambda) \ = \ \{\ '*', \lambda\} \\ \mathsf{FIRST}(F') \ = \ \mathsf{FIRST}(\mathsf{id}) \ \cup \ \mathsf{FIRST}(\lambda) \ = \ \{\ '*', \lambda\} \\ \mathsf{FOLLOW}(E) \ = \ \mathsf{FOLLOW}(E') \ = \ \{\ ')', \$ \} \\ \mathsf{FOLLOW}(T) \ = \ \mathsf{FOLLOW}(T') \ = \ \{\ '+', \ ')', \$ \} \\ \mathsf{FOLLOW}(F) \ = \ \{\ '*', \ '+', \ ')', \$ \} \\ \end{array}
```

#### Tabela Sintática

### Tabela Sintática Preditiva – Exemplo

```
 \begin{array}{lll} \textit{G:} \ E \ \to \ T \ E' \\ E' \ \to \ + \ T \ E' \ \mid \ \lambda \\ T \ \to \ F \ T' \\ F' \ \to \ * \ F \ T' \ \mid \ \lambda \\ F \ \to \ \text{id} \ \mid \ (E) \\ \end{array} \begin{array}{lll} \textbf{FIRST}(E) \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T) \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T) \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \ = \ \{\ '(',
```

#### Tabela Sintática

Não-		Símbolo de Entrada							
Terminal	id	+	*	(	)	\$			
E	E  o TE'			E  o TE'					
E'		$E' \rightarrow + TE'$			$E' \rightarrow \lambda$	$E'  o \lambda$			
T	T  o FT'			T  o FT'					
T'		$T'  o \lambda$	$T' \rightarrow *FT'$		$T'  o \lambda$	$T'  o \lambda$			
F	F  o id			$F \rightarrow (E)$					

## Tabela Sintática Preditiva – Exemplo

```
\begin{array}{lll} \textit{G: } E \to \textit{T} \ E' \\ E' \to + \textit{T} \ E' \ | \ \lambda \\ T \to \textit{F} \ T' \\ F' \to * \ \textit{F} \ T' \ | \ \lambda \\ F \to \text{id} \ | \ (E) \end{array} \qquad \begin{array}{ll} \textit{FIRST}(E) \ = \ \textit{FIRST}(TE') \ = \ \{ \ '(', \text{id}) \\ \textit{FIRST}(E') \ = \ \textit{FIRST}(+\text{TE}') \cup \ \textit{FIRST}(\lambda) \ = \ \{ \ '+', \ \lambda \} \\ \textit{FIRST}(T') \ = \ \textit{FIRST}(FT') \ = \ \{ \ '(', \text{id}) \\ \textit{FIRST}(T') \ = \ \textit{FIRST}(*FT') \cup \ \textit{FIRST}(\lambda) \ = \ \{ \ '*', \lambda \} \\ \textit{FOLLOW}(E) \ = \ \textit{FOLLOW}(E') \ = \ \{ \ ')', \$ \} \\ \textit{FOLLOW}(T) \ = \ \textit{FOLLOW}(T') \ = \ \{ \ '+', \ ')', \$ \} \\ \textit{FOLLOW}(F) \ = \ \{ \ '*', \ '+', \ ')', \$ \} \end{array}
```

#### Tabela Sintática

Não-		Símbolo de Entrada					
Terminal	id	+	*	(	)	\$	
E	$E \rightarrow TE'$			$E \rightarrow TE'$			
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$		
T	T  o FT'			T  o FT'			
T'		$T'  o \lambda$	$T' \to *FT'$		$T'  o \lambda$		
F	F  o id			$F \rightarrow (E)$			

### Tabela Sintática Preditiva – Exemplo

```
G: E \rightarrow T \ E'
E' \rightarrow + T \ E' \mid \lambda
T \rightarrow F \ T'
FIRST(E) = FIRST(TE') = \{ \ '(', id) \}
FIRST(E') = FIRST(+TE') \cup FIRST(\lambda) = \{ \ '+', \lambda \}
FIRST(T') = FIRST(FT') = \{ \ '(', id) \}
FIRST(T') = FIRST(*FT') \cup FIRST(\lambda) = \{ \ '*', \lambda \}
FIRST(F) = FIRST(id) \cup FIRST(E) = \{ \ id, \ '(') \}
FOLLOW(E) = FOLLOW(E') = \{ \ '+', \ '+', \ ')', \$ \}
FOLLOW(F) = \{ \ '*', \ '+', \ '+', \ ')', \$ \}
```

#### Tabela Sintática

Não-	Símbolo de Entrada					
Terminal	id	+	*	(	)	\$
E	E  o TE'			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E'  o \lambda$
T	T  o FT'			T  o FT'		
T'		$T'  o \lambda$	$T' \rightarrow *FT'$		$T'  o \lambda$	$T' \to \lambda$
F	F  o id			$F \rightarrow (E)$		

### Tabela Sintática Preditiva – Exemplo

```
G: E \rightarrow T \ E'
E' \rightarrow + T \ E' \mid \lambda
T \rightarrow F \ T'
FIRST(E) = FIRST(TE') = \{ \ '(', id) \}
FIRST(E') = FIRST(+TE') \cup FIRST(\lambda) = \{ \ '+', \lambda \}
FIRST(T') = FIRST(FT') = \{ \ '(', id) \}
FIRST(T') = FIRST(*FT') \cup FIRST(\lambda) = \{ \ '*', \lambda \}
FIRST(F) = FIRST(id) \cup FIRST(E) = \{ \ id, \ '(') \}
FOLLOW(E) = FOLLOW(E') = \{ \ '+', \ '+', \ ')', \$ \}
FOLLOW(F) = \{ \ '*', \ '+', \ '+', \ ')', \$ \}
```

#### Tabela Sintática

Não-		Símbolo de Entrada					
Terminal	id	+	*	(	)	\$	
E	E  o TE'			E  o TE'			
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E'  o \lambda$	
T	T  o FT'			T  o FT'			
T'		$T'  o \lambda$	$T' \rightarrow *FT'$		$T'  o \lambda$	$T'  o \lambda$	
F	F  o id			$F \rightarrow (E)$			

#### Tabela Sintática Preditiva – Exemplo

```
 \begin{array}{lll} \textit{G:} \ E \ \to \ T \ E' \\ E' \ \to \ + \ T \ E' \ \mid \ \lambda \\ T \ \to \ F \ T' \\ F' \ \to \ * \ F \ T' \ \mid \ \lambda \\ F \ \to \ \text{id} \ \mid \ (E) \\ \end{array} \begin{array}{lll} \textbf{FIRST}(E) \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T) \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ TRST(TE') \ = \ TRS
```

#### Tabela Sintática

Não-		Símbolo de Entrada					
Terminal	id	+	*	(	)	\$	
E	E  o TE'			$E \rightarrow TE'$			
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E'  o \lambda$	
T	T  o FT'			$T \rightarrow FT'$			
T'		$T' \rightarrow \lambda$	T'  o *FT'		$T'  o \lambda$	$T'  o \lambda$	
F	F  o id			F  o (E)			

### Tabela Sintática Preditiva – Exemplo

```
G: E \rightarrow T \ E'
E' \rightarrow + T \ E' \mid \lambda
T \rightarrow F \ T'
FIRST(E) = FIRST(TE') = \{ \ '(', id) \}
FIRST(E') = FIRST(+TE') \cup FIRST(\lambda) = \{ \ '+', \lambda \}
FIRST(T') = FIRST(FT') = \{ \ '(', id) \}
FIRST(T') = FIRST(*FT') \cup FIRST(\lambda) = \{ \ '*', \lambda \}
FIRST(F) = FIRST(id) \cup FIRST(E) = \{ \ id, \ '(') \}
FOLLOW(E) = FOLLOW(E') = \{ \ '+', \ '+', \ ')', \$ \}
FOLLOW(F) = \{ \ '*', \ '+', \ '+', \ ')', \$ \}
```

#### Tabela Sintática

Não-	Símbolo de Entrada					
Terminal	id	+	*	(	)	\$
Е	E  o TE'			$E \rightarrow TE'$		
E'		E'  ightarrow + TE'			$E' \rightarrow \lambda$	$E'  o \lambda$
T	T  o FT'			$T \rightarrow FT'$		
T'		${\cal T}'  o \lambda$	$T' \rightarrow *FT'$		$T'  o \lambda$	$T'  o \lambda$
F	F  o id			$F \rightarrow (E)$		

### Tabela Sintática Preditiva – Exemplo

```
 \begin{array}{lll} \textit{G:} \ E \ \to \ T \ E' \\ E' \ \to \ + \ T \ E' \ \mid \ \lambda \\ T \ \to \ F \ T' \\ F' \ \to \ * \ F \ T' \ \mid \ \lambda \\ F \ \to \ \text{id} \ \mid \ (E) \\ \end{array} \begin{array}{lll} \textbf{FIRST}(E) \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T) \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T) \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \ = \ \{\ '(',
```

#### Tabela Sintática

Não-		Símbolo de Entrada				
Terminal	id	+	*	(	)	\$
E	E  o TE'			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	T  o FT'			$T \rightarrow FT'$		
T'		$T'  o \lambda$	$T' \rightarrow *FT'$		$T'  o \lambda$	$T'  o \lambda$
F	F  o id			$F \rightarrow (E)$		

## Tabela Sintática Preditiva – Exemplo

```
 \begin{array}{lll} \textit{G:} \ E \ \to \ T \ E' \\ E' \ \to \ + \ T \ E' \ \mid \ \lambda \\ T \ \to \ F \ T' \\ F' \ \to \ * \ F \ T' \ \mid \ \lambda \\ F \ \to \ \text{id} \ \mid \ (E) \\ \end{array} \begin{array}{lll} \textbf{FIRST}(E) \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T) \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T) \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(T') \ = \ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \\ \textbf{FIRST}(TE') \ = \ \{\ '(', \text{id}) \ = \ \{\ '(',
```

#### Tabela Sintática

Não-		Símbolo de Entrada				
Terminal	id	+	*	(	)	\$
E	E  o TE'			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E'  o \lambda$
T	T  o FT'			$T \rightarrow FT'$		
T'		$T'  o \lambda$	$T' \rightarrow *FT'$		$T'  o \lambda$	$T'  o \lambda$
F	F  o id			$F \rightarrow (E)$		

### Tabela Sintática Preditiva – Exemplo

```
\begin{array}{lll} \textit{G: } E \to \textit{T} \; E' & \textit{FIRST}(E) \; = \; \textit{FIRST}(TE') \; = \; \{\; '(', id) \} \\ E' \to + \; \textit{T} \; E' \; \mid \; \lambda & \textit{FIRST}(E') \; = \; \textit{FIRST}(+TE') \cup \; \textit{FIRST}(\lambda) \; = \; \{\; '+', \; \lambda \} \\ T \to F \; T' & \textit{FIRST}(T) \; = \; \textit{FIRST}(FT') \; = \; \{\; '(', id) \} \\ F \to id \; \mid \; (E) & \textit{FIRST}(T') \; = \; \textit{FIRST}(*FT') \cup \; \textit{FIRST}(\lambda) \; = \; \{\; '*', \lambda \} \\ F \to id \; \mid \; (E) & \textit{FIRST}(F) \; = \; \textit{FIRST}(id) \cup \; \textit{FIRST}(E)) \; = \; \{\; id, \; '(') \} \\ FOLLOW(E) = \; FOLLOW(E') \; = \; \{\; '+', \; '+', \; ')', \$ \} \\ FOLLOW(F) \; = \; \{\; '*', \; '+', \; ')', \$ \} \end{array}
```

#### Tabela Sintática

Não-	Símbolo de Entrada					
Terminal	id	+	*	(	)	\$
Е	E  o TE'			E  o TE'		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E'  o \lambda$
T	T  o FT'			T  o FT'		
T'		$T'  o \lambda$	$T' \rightarrow *FT'$		$T'  o \lambda$	$T'  o \lambda$
F	F  o id			$F \rightarrow (E)$		

# Gramática **LL(**k**)**

# Gramática LL(k)

Uma gramática  $\mathbf{LL}(k)$  é aquela que pode ser analisada por um *parser*  $\mathbf{LL}(k)$ , isto é, uma analisador sintático que faz uma varredura da entrada da esquerda para direita (1°  $\mathbf{L}$ ) produzindo uma derivação mais à esquerda (2°  $\mathbf{L}$ ) utlizando-se de k símbolos (ou *tokens*) da entrada como *lookahead*.

#### Gramática **LL(1)**

Seja G uma gramática **LL(1)** então para todo par de regras distintas de um não terminal A em que  $A \rightarrow \alpha \mid \beta$  valem as seguintes condições:

- $\alpha$  e  $\beta$  não derivam ao mesmo tempo sentenças que se iniciam pelo mesmo terminal a, isto é,  $\mathsf{FIRST}(\alpha) \cap \mathsf{FIRST}(\beta) = \emptyset$
- No máximo uma das formas sentenciais produz vazio, isto é, ou  $\lambda \in \mathsf{FIRST}(\alpha)$  ou  $\lambda \in \mathsf{FIRST}(\beta)$  mas não ambos
- Se  $\beta \stackrel{*}{\Rightarrow} \lambda$ , então  $\alpha$  não produz nenhuma sentença que se inicie por um terminal em **FOLLOW**(A), isto é, **FIRST**( $\alpha$ )  $\cap$  **FOLLOW**(A) =  $\emptyset$

# Gramática **LL(***k***)**

# Gramática LL(k)

Uma gramática  $\mathbf{LL}(k)$  é aquela que pode ser analisada por um *parser*  $\mathbf{LL}(k)$ , isto é, uma analisador sintático que faz uma varredura da entrada da esquerda para direita (1°  $\mathbf{L}$ ) produzindo uma derivação mais à esquerda (2°  $\mathbf{L}$ ) utlizando-se de k símbolos (ou *tokens*) da entrada como *lookahead*.

# Gramática **LL(1)**

Seja G uma gramática **LL(1)** então para todo par de regras distintas de um não terminal A em que  $A \to \alpha \mid \beta$  valem as seguintes condições:

- $\alpha$  e  $\beta$  não derivam ao mesmo tempo sentenças que se iniciam pelo mesmo terminal a, isto é,  ${\sf FIRST}(\alpha) \cap {\sf FIRST}(\beta) = \emptyset$
- No máximo uma das formas sentenciais produz vazio, isto é, ou  $\lambda \in \mathsf{FIRST}(\alpha)$  ou  $\lambda \in \mathsf{FIRST}(\beta)$  mas não ambos
- Se  $\beta \stackrel{*}{\Rightarrow} \lambda$ , então  $\alpha$  não produz nenhuma sentença que se inicie por um terminal em **FOLLOW**(A), isto é, **FIRST**( $\alpha$ )  $\cap$  **FOLLOW**(A) =  $\emptyset$

LL(k)

Seja GLC  $G: E \rightarrow T E'$ 

# Gramática **LL(***k***)**

# Gramática **LL(1)** – Exemplo

```
E' \rightarrow + T E' \mid \lambda
                T \rightarrow F T'
                T' \rightarrow *FT' \mid \lambda
                F \rightarrow id \mid (E)
FIRST(E) = FIRST(TE') = FIRST(T) = FIRST(FT') = FIRST(F) = \{ '(', id) \}
FIRST(E') = FIRST(+TE') \cup FIRST(\lambda) = \{ '+', \lambda \}
FIRST(T') = FIRST(*FT') \cup FIRST(\lambda) = \{ '*', \lambda \}
FOLLOW(E) = FOLLOW(E') = \{ ')', \$ \}
FOLLOW(T) = FOLLOW(T') = \{ '+', ')', \$ \}
FOLLOW(F) = { '*', '+', ')', $}
```

# Gramática **LL(***k*)

# Gramática **LL(1)** – Exemplo

```
Seja GLC G: E \rightarrow T E'
                E' \rightarrow + T E' \mid \lambda
                T \rightarrow F T'
                T' \rightarrow *FT' \mid \lambda
                F \rightarrow id \mid (E)
FIRST(E) = FIRST(TE') = FIRST(T) = FIRST(FT') = FIRST(F) = \{ '(', id) \}
FIRST(E') = FIRST(+TE') \cup FIRST(\lambda) = \{ '+', \lambda \}
FIRST(T') = FIRST(*FT') \cup FIRST(\lambda) = \{ '*', \lambda \}
FOLLOW(E) = FOLLOW(E') = \{ ')', \$ \}
FOLLOW(T) = FOLLOW(T') = \{ '+', ')', \$ \}
FOLLOW(F) = { '*', '+', ')', $}
```

Como todas as regras de G atendem as condições, então G é **LL(1)**!

LL(k)

000

# Gramática LL(1) – Contra-Exemplo

```
Seja GLC G: S \rightarrow i E t S S' \mid a

S' \rightarrow e S \mid \lambda

E \rightarrow b

FIRST(S) = \{i, a\}

FIRST(S') = \{e, \lambda\}

FIRST(E) = \{b\}

FOLLOW(S) = \{e, \$\}

FOLLOW(S') = \{e, \$\}

FOLLOW(E) = \{t\}
```

Então G não é LL(1), pois FIRST(eS)  $\cap$  FOLLOW(S')  $\neq \emptyset$ 

LL(k)

000

# Gramática LL(1) – Contra-Exemplo

```
Seja GLC G: S \rightarrow i E t S S' \mid a

S' \rightarrow e S \mid \lambda

E \rightarrow b

FIRST(S) = \{i, a\}

FIRST(S') = \{e, \lambda\}

FIRST(E) = \{b\}

FOLLOW(S) = \{e, \$\}

FOLLOW(S') = \{e, \$\}

FOLLOW(E) = \{t\}
```

Então G não é LL(1), pois FIRST(eS)  $\cap$  FOLLOW(S')  $\neq \emptyset$ 

A pilha de uma analisador preditivo não-recursivo torna explícitos os terminais e não-terminais que o mesmo espera reconhecer com o restante da entrada.

Um erro pode ser detectado durante a análise preditiva quando:

- o terminal do topo da pilha não coincide com o próximo da entrada ou
- a entrada da tabela sintática M[A, a] está vazia, em que A representa o não-terminal que está no topo da pilha e a é o próximo terminal na entrada.

A recuperação de erros na modalidade do desespero (panic mode) está baseada na ideia de se ignorar símbolos na entrada até que surja um token pertencente a um conjunto pré-selecionado de tokens de sincronização.

A efetividade dessa abordagem depende da escolha de tais conjuntos de sincronização.

A pilha de uma analisador preditivo não-recursivo torna explícitos os terminais e não-terminais que o mesmo espera reconhecer com o restante da entrada.

Um erro pode ser detectado durante a análise preditiva quando:

- o terminal do topo da pilha não coincide com o próximo da entrada ou
- a entrada da tabela sintática M[A, a] está vazia, em que A representa o não-terminal que está no topo da pilha e a é o próximo terminal na entrada.

A recuperação de erros na modalidade do desespero (panic mode) está baseada na ideia de se ignorar símbolos na entrada até que surja um token pertencente a um conjunto pré-selecionado de tokens de sincronização.

A efetividade dessa abordagem depende da escolha de tais conjuntos de sincronização.

A pilha de uma analisador preditivo não-recursivo torna explícitos os terminais e não-terminais que o mesmo espera reconhecer com o restante da entrada.

Um erro pode ser detectado durante a análise preditiva quando:

- o terminal do topo da pilha não coincide com o próximo da entrada ou
- a entrada da tabela sintática M[A, a] está vazia, em que A representa o não-terminal que está no topo da pilha e a é o próximo terminal na entrada.

A recuperação de erros na modalidade do desespero (panic mode) está baseada na ideia de se ignorar símbolos na entrada até que surja um token pertencente a um conjunto pré-selecionado de tokens de sincronização.

A efetividade dessa abordagem depende da escolha de tais conjuntos de sincronização.

A pilha de uma analisador preditivo não-recursivo torna explícitos os terminais e não-terminais que o mesmo espera reconhecer com o restante da entrada.

Um erro pode ser detectado durante a análise preditiva quando:

- o terminal do topo da pilha não coincide com o próximo da entrada ou
- a entrada da tabela sintática M[A, a] está vazia, em que A representa o não-terminal que está no topo da pilha e a é o próximo terminal na entrada.

A recuperação de erros na modalidade do desespero (panic mode) está baseada na ideia de se ignorar símbolos na entrada até que surja um token pertencente a um conjunto pré-selecionado de tokens de sincronização.

A efetividade dessa abordagem depende da escolha de tais conjuntos de sincronização.

A pilha de uma analisador preditivo não-recursivo torna explícitos os terminais e não-terminais que o mesmo espera reconhecer com o restante da entrada.

Um erro pode ser detectado durante a análise preditiva quando:

- o terminal do topo da pilha não coincide com o próximo da entrada ou
- a entrada da tabela sintática M[A, a] está vazia, em que A representa o não-terminal que está no topo da pilha e a é o próximo terminal na entrada.

A recuperação de erros na modalidade do desespero (panic mode) está baseada na ideia de se ignorar símbolos na entrada até que surja um token pertencente a um conjunto pré-selecionado de tokens de sincronização.

A efetividade dessa abordagem depende da escolha de tais conjuntos de sincronização.

#### Eis algumas heurísticas para escolha do conjunto de sincronização:

① Como ponto de partida, pode-se usar todos os símbolos de **FOLLOW**(A) no conjunto de *tokens* de sincronização para o não-terminal A.

Assim se ignorarmos todas as *tokens* até que um elemento de FOLLOW(A) seja visto e removermos o não-terminal A da pilha, e provável que a análise possa continuar.

Eis algumas heurísticas para escolha do conjunto de sincronização:

1 Como ponto de partida, pode-se usar todos os símbolos de **FOLLOW**(A) no conjunto de *tokens* de sincronização para o não-terminal A.

Assim se ignorarmos todas as *tokens* até que um elemento de FOLLOW(A) seja visto e removermos o não-terminal A da pilha, é provável que a análise possa continuar.

Eis algumas heurísticas para escolha do conjunto de sincronização:

1 Como ponto de partida, pode-se usar todos os símbolos de **FOLLOW**(A) no conjunto de *tokens* de sincronização para o não-terminal A.

Assim se ignorarmos todas as *tokens* até que um elemento de FOLLOW(A) seja visto e removermos o não-terminal A da pilha, é provável que a análise possa continuar.

#### Eis algumas heurísticas para escolha do conjunto de sincronização (cont.):

② Geralmente, não é suficiente usar **FOLLOW**(*A*) como conjunto de sincronização para *A*.

Por exemplo, se pontos-e-vírgulas terminarem os enunciados (como em C C++ e Java), então as palavras-chave que iniciam os enunciados não devem aparecer no conjunto **FOLLOW** do não terminal que gera expressões.

Assim, um ponto-e-vírgula ausente após uma atribuição poderia resultar na desconsideração da palavra-chave que inicia o próximo enunciado.

**Solução:** explorar a estrutura hierárquica que existe nas construções de uma linguagem (por exemplo: expressões aparecem em enunciados que figuram dentro de blocos e assim por diante).

Pode-se adicionar oo conjunto de sincronização de uma estrutura mais baixa os símbolos que começam as construções mais altas (por exemplo, adicionar as palavras-chave que iniciam comandos aos conjuntos de sincronização dos não-terminais que geram expressões).

Eis algumas heurísticas para escolha do conjunto de sincronização (cont.):

Geralmente, não é suficiente usar FOLLOW(A) como conjunto de sincronização para A.

Por exemplo, se pontos-e-vírgulas terminarem os enunciados (como em C C++ e Java), então as palavras-chave que iniciam os enunciados não devem aparecer no conjunto **FOLLOW** do não terminal que gera expressões.

Assim, um ponto-e-vírgula ausente após uma atribuição poderia resultar na desconsideração da palavra-chave que inicia o próximo enunciado.

**Solução:** explorar a estrutura hierárquica que existe nas construções de uma linguagem (por exemplo: expressões aparecem em enunciados que figuram dentro de blocos e assim por diante).

Pode-se adicionar oo conjunto de sincronização de uma estrutura mais baixa os símbolos que começam as construções mais altas (por exemplo adicionar as palavras-chave que iniciam comandos aos conjuntos de sincronização dos não-terminais que geram expressões).

Eis algumas heurísticas para escolha do conjunto de sincronização (cont.):

② Geralmente, não é suficiente usar FOLLOW(A) como conjunto de sincronização para A.

Por exemplo, se pontos-e-vírgulas terminarem os enunciados (como em C, C++ e Java), então as palavras-chave que iniciam os enunciados não devem aparecer no conjunto **FOLLOW** do não terminal que gera expressões.

Assim, um ponto-e-vírgula ausente após uma atribuição poderia resultar na desconsideração da palavra-chave que inicia o próximo enunciado.

**Solução:** explorar a estrutura hierárquica que existe nas construções de uma linguagem (por exemplo: expressões aparecem em enunciados que figuram dentro de blocos e assim por diante).

Pode-se adicionar oo conjunto de sincronização de uma estrutura mais baixa os símbolos que começam as construções mais altas (por exemplo adicionar as palavras-chave que iniciam comandos aos conjuntos de sincronização dos não-terminais que geram expressões).

Eis algumas heurísticas para escolha do conjunto de sincronização (cont.):

Geralmente, não é suficiente usar FOLLOW(A) como conjunto de sincronização para A.

Por exemplo, se pontos-e-vírgulas terminarem os enunciados (como em C, C++ e Java), então as palavras-chave que iniciam os enunciados não devem aparecer no conjunto **FOLLOW** do não terminal que gera expressões.

Assim, um ponto-e-vírgula ausente após uma atribuição poderia resultar na desconsideração da palavra-chave que inicia o próximo enunciado.

**Solução:** explorar a estrutura hierárquica que existe nas construções de uma linguagem (por exemplo: expressões aparecem em enunciados que figuram dentro de blocos e assim por diante).

Pode-se adicionar oo conjunto de sincronização de uma estrutura mais baixa os símbolos que começam as construções mais altas (por exemplo adicionar as palavras-chave que iniciam comandos aos conjuntos de sincronização dos não-terminais que geram expressões).

Eis algumas heurísticas para escolha do conjunto de sincronização (cont.):

Geralmente, não é suficiente usar FOLLOW(A) como conjunto de sincronização para A.

Por exemplo, se pontos-e-vírgulas terminarem os enunciados (como em C, C++ e Java), então as palavras-chave que iniciam os enunciados não devem aparecer no conjunto **FOLLOW** do não terminal que gera expressões.

Assim, um ponto-e-vírgula ausente após uma atribuição poderia resultar na desconsideração da palavra-chave que inicia o próximo enunciado.

**Solução:** explorar a estrutura hierárquica que existe nas construções de uma linguagem (por exemplo: expressões aparecem em enunciados que figuram dentro de blocos e assim por diante).

Pode-se adicionar oo conjunto de sincronização de uma estrutura mais baixa os símbolos que começam as construções mais altas (por exemplo, adicionar as palavras-chave que iniciam comandos aos conjuntos de sincronização dos não-terminais que geram expressões).

Eis algumas heurísticas para escolha do conjunto de sincronização (cont.):

Geralmente, não é suficiente usar FOLLOW(A) como conjunto de sincronização para A.

Por exemplo, se pontos-e-vírgulas terminarem os enunciados (como em C, C++ e Java), então as palavras-chave que iniciam os enunciados não devem aparecer no conjunto **FOLLOW** do não terminal que gera expressões.

Assim, um ponto-e-vírgula ausente após uma atribuição poderia resultar na desconsideração da palavra-chave que inicia o próximo enunciado.

**Solução:** explorar a estrutura hierárquica que existe nas construções de uma linguagem (por exemplo: expressões aparecem em enunciados que figuram dentro de blocos e assim por diante).

Pode-se adicionar oo conjunto de sincronização de uma estrutura mais baixa os símbolos que começam as construções mais altas (por exemplo, adicionar as palavras-chave que iniciam comandos aos conjuntos de sincronização dos não-terminais que geram expressões).

- Se adicionarmos os símbolos de FIRST(A) ao conjunto de sincronização para o não-terminal A, pode ser possível retornar a análise a partir de A caso um símbolo de FIRST(A) apareca na entrada.
- $\odot$  Se um não-terminal puder gerar a cadeia vazia, então a produção que deriva  $\lambda$  pode ser usada como default.
  - Agindo-se assim, pode-se postergar a detecção de algum erro, mas não se pode fazer com que um erro seja perdido.
  - Essa abordagem reduz o número de não-terminais que devem ser considerados na recuperação de erros.
- Se um terminal do topo da pilha não puder ser reconhecido, uma ideia é simplesmente desempilhá-lo, emitir uma mensagem de que o terminal foi inserido e prosseguir na análise.
  - Consequentemente, essa abordagem faz que com o conjunto de sincronização de um token consista em todos os demais tokens

- ② Se adicionarmos os símbolos de FIRST(A) ao conjunto de sincronização para o não-terminal A, pode ser possível retornar a análise a partir de A, caso um símbolo de FIRST(A) apareça na entrada.
- $\odot$  Se um não-terminal puder gerar a cadeia vazia, então a produção que deriva  $\lambda$  pode ser usada como default.
  - Agindo-se assim, pode-se postergar a detecção de algum erro, mas não se pode fazer com que um erro seja perdido.
  - Essa abordagem reduz o número de não-terminais que devem ser considerados na recuperação de erros.
- Se um terminal do topo da pilha não puder ser reconhecido, uma ideia é simplesmente desempilhá-lo, emitir uma mensagem de que o terminal foi inserido e prosseguir na análise.
  - Consequentemente, essa abordagem faz que com o conjunto de sincronização de um token consista em todos os demais tokens

- ② Se adicionarmos os símbolos de FIRST(A) ao conjunto de sincronização para o não-terminal A, pode ser possível retornar a análise a partir de A, caso um símbolo de FIRST(A) apareca na entrada.
- Se um não-terminal puder gerar a cadeia vazia, então a produção que deriva  $\lambda$  pode ser usada como default.
  - Agindo-se assim, pode-se postergar a detecção de algum erro, mas não se pode fazer com que um erro seja perdido.
  - Essa abordagem reduz o número de não-terminais que devem ser considerados na recuperação de erros.
- Se um terminal do topo da pilha não puder ser reconhecido, uma ideia é simplesmente desempilhá-lo, emitir uma mensagem de que o terminal foi inserido e prosseguir na análise.
  - Consequentemente, essa abordagem faz que com o conjunto de sincronização de um token consista em todos os demais tokens

- ② Se adicionarmos os símbolos de FIRST(A) ao conjunto de sincronização para o não-terminal A, pode ser possível retornar a análise a partir de A, caso um símbolo de FIRST(A) apareca na entrada.
- $oldsymbol{3}$  Se um não-terminal puder gerar a cadeia vazia, então a produção que deriva  $\lambda$  pode ser usada como *default*.
  - Agindo-se assim, pode-se postergar a detecção de algum erro, mas não se pode fazer com que um erro seja perdido.
  - Essa abordagem reduz o número de não-terminais que devem ser considerados na recuperação de erros.
- Se um terminal do topo da pilha não puder ser reconhecido, uma ideia é simplesmente desempilhá-lo, emitir uma mensagem de que o terminal foi inserido e prosseguir na análise.
  - Consequentemente, essa abordagem faz que com o conjunto de sincronização de um token consista em todos os demais tokens

- ② Se adicionarmos os símbolos de FIRST(A) ao conjunto de sincronização para o não-terminal A, pode ser possível retornar a análise a partir de A, caso um símbolo de FIRST(A) apareca na entrada.
- $oldsymbol{3}$  Se um não-terminal puder gerar a cadeia vazia, então a produção que deriva  $\lambda$  pode ser usada como *default*.
  - Agindo-se assim, pode-se postergar a detecção de algum erro, mas não se pode fazer com que um erro seja perdido.
  - Essa abordagem reduz o número de não-terminais que devem ser considerados na recuperação de erros.
- Se um terminal do topo da pilha não puder ser reconhecido, uma ideia é simplesmente desempilhá-lo, emitir uma mensagem de que o terminal foi inserido e prosseguir na análise.
  - Consequentemente, essa abordagem faz que com o conjunto de sincronização de um token consista em todos os demais tokens.

- 2 Se adicionarmos os símbolos de **FIRST**(*A*) ao conjunto de sincronização para o não-terminal *A*, pode ser possível retornar a análise a partir de *A*, caso um símbolo de **FIRST**(*A*) apareça na entrada.
- $oldsymbol{3}$  Se um não-terminal puder gerar a cadeia vazia, então a produção que deriva  $\lambda$  pode ser usada como *default*.
  - Agindo-se assim, pode-se postergar a detecção de algum erro, mas não se pode fazer com que um erro seja perdido.
  - Essa abordagem reduz o número de não-terminais que devem ser considerados na recuperação de erros.
- Se um terminal do topo da pilha não puder ser reconhecido, uma ideia é simplesmente desempilhá-lo, emitir uma mensagem de que o terminal foi inserido e prosseguir na análise.
  - Consequentemente, essa abordagem faz que com o conjunto de sincronização de um *token* consista em todos os demais *tokens*

## Tratamento de Erros

Eis algumas heurísticas para escolha do conjunto de sincronização (cont.):

- ② Se adicionarmos os símbolos de FIRST(A) ao conjunto de sincronização para o não-terminal A, pode ser possível retornar a análise a partir de A, caso um símbolo de FIRST(A) apareca na entrada.
- $oldsymbol{3}$  Se um não-terminal puder gerar a cadeia vazia, então a produção que deriva  $\lambda$  pode ser usada como *default*.
  - Agindo-se assim, pode-se postergar a detecção de algum erro, mas não se pode fazer com que um erro seja perdido.
  - Essa abordagem reduz o número de não-terminais que devem ser considerados na recuperação de erros.
- Se um terminal do topo da pilha não puder ser reconhecido, uma ideia é simplesmente desempilhá-lo, emitir uma mensagem de que o terminal foi inserido e prosseguir na análise.
  - Consequentemente, essa abordagem faz que com o conjunto de sincronização de um *token* consista em todos os demais *tokens*.

Não-				Símbolo de Ei	ntrada		
Terminal	id		+	*	(	)	\$
Ε	$E \rightarrow TE'$				$E \rightarrow TE'$	sinc	sinc
E'		E' -	$\rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$		sinc		$T \rightarrow FT'$	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$		sinc	sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha	Entrada	Aç	ăo	
•			ES				
			ES	id * + id\$			
			TE'S	id * + id\$			
			FT'E'S	id * + id\$	Saída: T -		
			id <i>T' E'</i> \$	id * + id\$	Saída: F -		
			T'E'	* + id\$	⇒ Casamo		
			*FT'E'	* + id\$	Saída: T'		
			FT'E'S	+ id\$	⇒ Casamo		
			T'E'S	+ id\$	⇒ Desem		
			E' 9	+ id\$	Saída: T'		
			+TE'	+ id\$	Saída: E'		
			TE'S	id\$	⇒ Casamo		
			FT'E'S	id\$	Saída: T -		
			idT'E'	id\$	Saída: F -		
			T'E'S				
			E' 9				

	Não-					Símbolo de En	trada		
	Terminal	id		+		*	(	)	\$
	Ε	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
	E'		E' -	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
	T	$T \rightarrow FT'$		sinc			$T \rightarrow FT'$	sinc	sinc
	T'		T	$' \rightarrow \lambda$		$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
	F	$F \rightarrow id$		sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Si	mulação:	Casam	ento	Pilha		Entrada	Aç	ão .	
	,			E:	5	+id*+id\$			
				ES		id * + id\$	⇒ Pular:		
				TE'S		id * + id\$	Saída: E -		
				FT'E'S		id * + id\$	Saída: T -		
				idT'E'		id * + id\$	Saída: F -		
				T'E'		* + id\$	⇒ Casame		
				*FT'E'S		* + id\$	Saída: T'		
				FT'E'S		+ id\$	⇒ Casame		
				T'E'		+ id\$	⇒ Desem		
				E'S		+ id\$	Saída: $T^\prime$		
				+TE'		+ id\$	Saída: E'		
				TE'S		id\$	⇒ Casame		
				FT'E'S		id\$	Saída: T -		
				idT'E'		id\$	Saída: F -		
				T'E'		\$	⇒ Casame		
				E'S		\$	Saída: T'		
							~/		

Não-				Símbolo de Ei	ntrada		
Terminal	id		+	*	(	)	\$
E	$E \rightarrow TE'$				$E \rightarrow TE'$	sinc	sinc
E'		E' -	$\rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$	İ	sinc		$T \rightarrow FT'$	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$	İ	sinc	sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha	Entrada	Aç	ão	
,			E	+id * + id\$			
			ES				
			TE'S	id * + id\$			
			FT'E'S	id * + id\$	Saída: T -		
			idT'E'	id * + id\$	Saída: F -		
			T'E'	* + id\$	⇒ Casamo		
			*FT'E'	* + id\$	Saída: T'		
			FT'E'S	+ id\$	⇒ Casamo		
			T'E'				
			E' 9	+ id\$			
			+TE'	+ id\$	Saída: E'		
			TE'S	id\$	⇒ Casamo		
			FT'E'S	id\$	Saída: T -		
			idT'E'	id\$	Saída: F -		
			T'E'\$				
			E' 9		Saída: T'		
			0		Salda. E		

Não-				Símbolo de Er	ntrada		
Terminal	id		+	*	(	)	\$
E	$E \rightarrow TE'$				$E \rightarrow TE'$	sinc	sinc
E'		E' -	$\rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$		sinc		$T \rightarrow FT'$	sinc	sinc
T'		T	$' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$		sinc	sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha	Entrada	Aç	ão	
•			ES				
			ES				
			TE'S	id * + id\$			
			FT'E'S				
			idT'E'		Saída: F -		
			T'E'				
			*FT'E'		Saída: T'		
			FT'E'S		⇒ Casamo		
			T'E'				
			E'				
			+TE'				
			TE'S		⇒ Casamo		
			FT'E'S		Saída: T -		
			id <i>T'E'</i> \$		Saída: F -		
			T'E'S		⇒ Casamo		
			E' 9		Saída: T'		
					Saída: E'		

· · u cu · · · c				p.o			
Não-				Símbolo de E	ntrada		
Terminal	id		+	*	1 (	)	\$
E	$E \rightarrow TE'$				$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$		sinc		$T \rightarrow FT'$	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$	İ	sinc	sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha	Entrada	Aç	ão	
,			ES		5		
			ES				
			TE'S				
			FT'E'S		Saída: T		
			idT'E'		Saída: F		
			T'E'S				
			*FT'E'S		Saída: T'		
			FT'E'S		⇒ Casam		
			T'E'S				
			E'S	+ id9			
			+TE'	+ id9	Saída: $E'$		
			TE'S	id9	⇒ Casam		
			FT'E'S	ids	Saída: T		
			idT'E'S		Saída: F -		
			T'E'S				
			E'S	5	Saída: $T'$		
			(	9	Saída: E'		

Tratamic	iito ac L	. 1 1 0 .	) L/(	ilibio			
Não-				Símbolo de Ei	ntrada		
Terminal	id		+	*	(	)	\$
E	$E \rightarrow TE'$				$E \rightarrow TE'$	sinc	sinc
E'		E' -	$\rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$		sinc		$T \rightarrow FT'$	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	$T' \to *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$		sinc	sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha	Entrada	Aç	ão	
•			ES				
			ES				
			TE'S				
			FT'E'S	id * + id\$	Saída: T -		
			idT'E'		Saída: F -		
			T'E'	* + id\$	⇒ Casamo		
			*FT'E'				
			FT'E'S	+ id\$	⇒ Casamo		
			T'E'				
			E' 9	+ id\$			
			+TE'	+ id\$	Saída: <i>E'</i>		
			TE'S	id\$	⇒ Casamo		
			FT'E'	id\$	Saída: T -		
			id <i>T′ E′</i> \$	id\$	Saída: F -		
			T'E'	\$	⇒ Casamo		
			E' \$				
			0		Salda E		

· · u cu · · · c				۰۲	. •			
Não-				Sín	nbolo de En	trada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$	İ	sinc			T  o FT'	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	T' -	$\rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$	İ	sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha		Entrada	Açã	ão .	
,			ES	+	id * + id\$			
			ES		id * + id\$	⇒ Pular: ·		
			TE'S		id * + id\$	Saída: E -		
			FT'E'S		id * + id\$	Saída: T -		
			idT'E'S			Saída: F -		
			T'E'			⇒ Casame		
			*FT'E'S	5		Saída: $T'$		
			FT'E'S			⇒ Casame		
			T'E'			⇒ Desemp		
			E'S			Saída: $T'$		
			+TE'			Saída: E'		
			TE'S			⇒ Casame		
			FT'E'S			Saída: T -		
			idT'E'			Saída: F -		
			T'E'			$\Rightarrow$ Casame		
			E'S	5		Saída: $T'$		
						Saída: F'		

Hataiiic	iito uc L	.1103		ilibio				
Não-				Símbolo	de En	trada		
Terminal	id		+	*		(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E' -	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$		sinc			$T \rightarrow FT'$	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	$T' \rightarrow *F$	T'		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$		sinc	sinc		$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha	Entr	ada	Açâ	ão .	
•			E\$	+id *	+ id\$			
			E\$		+ id\$	⇒ Pular: -		
			TE'\$		+ id\$	Saída: E -		
			FT'E'\$		+ id\$	Saída: T -	$\rightarrow$ FT $'$	
			id T' E' \$	id *		Saída: F -		
			T'E'\$	100		⇒ Casame		
			*FT'E'\$	100		Saída: T'		
			FT'E'\$			⇒ Casame		
			T'E'\$			⇒ Desemp		
			E' \$			Saída: T'		
			+TE'\$			Saída: E'		
			TE'\$			⇒ Casame		
			FT' E' \$			Saída: T -		
			id <i>T' E'</i> \$			Saída: F -		
			T'E'\$			⇒ Casame		
			E'\$			Saída: $T'$		
			0			Salda E		

· · u cu · · · c				,,,,p,,	•			
Não-				Símbo	olo de En	trada		
Terminal	id		+	*		(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$	İ	sinc			$T \rightarrow FT'$	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	$T' \rightarrow$	*FT'		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$	İ	sinc	sir	ıc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha	E	ntrada	Açâ	ío	
•			ES		* + id\$			
			ES		* + id\$	⇒ Pular: -		
			TE'S		* + id\$	Saída: E –		
			FT'E'S		* + id\$	Saída: T -	$\rightarrow FT'$	
			id <i>T' E'</i> \$	id	* + id\$	Saída: F -		
			T'E'9			⇒ Casame		
			*FT'E'	5		Saída: $T'$		
			FT'E'S			⇒ Casame		
			T'E'	5		⇒ Desemp		
			E' 9			Saída: T'		
			+TE'	5		Saída: E'		
			TE'S	5		⇒ Casame		
			FT'E'	5		Saída: T -		
			id <i>T'E'</i> \$	5		Saída: F -		
			T'E'	5		⇒ Casame		
			E' 9	5		Saída: $T^\prime$		
			4	: 1		Saída: F'		

· · u cu · · · c				5111P10				
Não-				Símbolo de	Ent	rada		
Terminal	id		+	*	Т	(	)	\$
E	$E \rightarrow TE'$				T	$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$	İ	sinc			$T \rightarrow FT'$	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	$T' \rightarrow *FT'$			$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$	İ	sinc	sinc		$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha	Entrada		Açã	ío	
,			ES		d\$			
			ES		đ\$	⇒ Pular: -		
			TE'S		đ\$	Saída: E -		
			FT'E'S		d\$	Saída: T -	→ FT′	
			idT'E'			Saída: F -	→ id	
			T'E'			⇒ Casame		
			*FT'E'S			Saída: $T'$		
			FT'E'S			⇒ Casame		
			T'E'			⇒ Desemp		
			E'S			Saída: T'		
			+TE'S			Saída: E'		
			TE'S	1		⇒ Casame		
			FT'E'S			Saída: T -		
			idT'E'			Saída: F -		
			T'E'			⇒ Casame		
			E'S	Б		Saída: T'		
						Saída: E'		

· · u cu · · · c				٠				
Não-					Símbolo de En	trada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$		sinc			$T \rightarrow FT'$	sinc	sinc
T'		Т	$\lambda' \rightarrow \lambda$	7	$\Gamma' \to *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$		sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha	Т	Entrada	Açã	ão .	
•			ES		+id*+id\$			
			ES		id * + id\$	⇒ Pular: ·		
			TE'S		id * + id\$	Saída: E -		
			FT'E'S		id * + id\$	Saída: T -	$\rightarrow$ FT $'$	
			id T' E' 9		id * + id\$	Saída: F -	→ id	
	id		T'E'S		* + id\$	⇒ Casame		
			*FT'E'S	5		Saída: $T'$		
			FT'E'S			⇒ Casame		
			T'E'			⇒ Desemp		
			E'S			Saída: $T'$		
			+TE'	5		Saída: E'		
			TE'S			⇒ Casame		
			FT'E'S	5		Saída: T -		
			idT'E'S			Saída: F -		
			T'E'	5		⇒ Casame		
			E'S	5		Saída: $T'$		
			9	8		Saída: F'		

· · u cu · · · c					P. 0			
Não-				S	ímbolo de En	trada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$	İ	sinc			T  o FT'	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	T'	$\rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$	İ	sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha		Entrada	Açã	ão .	
•			ES		+id * + id\$			
			ES		id * + id\$	⇒ Pular: ·		
			TE'S		id * + id\$	Saída: E -		
			FT'E'S		id * + id\$	Saída: T -	$\rightarrow$ FT $'$	
			id T' E' 9		id * + id\$	Saída: F -	→ id	
	id		T'E'S		* + id\$	⇒ Casame		
			*FT'E'	5	* + id\$	Saída: T'		
			FT'E'S			⇒ Casame		
			T'E'			⇒ Desemp		
			E'S			Saída: T'		
			+TE'	5		Saída: E'		
			TE'S	5		⇒ Casame		
			FT'E'S	5		Saída: T -		
			idT'E'	5		Saída: F -		
			T'E'	5		⇒ Casame		
			E'S	5		Saída: $T'$		
			(			Saída: E'		

Tracamic	iito ac E	0		- I I I				
Não-				Sím	ibolo de En	itrada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$ sinc			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$	
T	$T \rightarrow FT'$	İ			$T \rightarrow FT'$	sinc	sinc	
T'		T	$\lambda' \rightarrow \lambda$	T' -	$\rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F			sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	ção: Casamento		Pilha		Entrada	Aç	io	
,			ES	6 +	id * + id\$			
			ES		id*+id	⇒ Pular:	+	
			TE'S		id*+id	Saída: E -		
			FT'E'S		id*+id	Saída: T -	$\rightarrow FT'$	
			id <i>T' E'</i> \$		id*+id	Saída: F -	→ id	
	id		T'E'		* + id\$	⇒ Casame	ento: id	
	id		*FT'E'\$	5	* + id\$	Saída: T'	$\rightarrow *FT'$	
			FT'E'9			⇒ Casame		
			T'E'	5		⇒ Desem		
			E' 9	5		Saída: $T'$		
			+TE'	5		Saída: E'		
			TE'	5		⇒ Casame		
			FT'E'	5		Saída: T -		
			id <i>T'E'</i> \$	5		Saída: F -		
			T'E'	5		⇒ Casame		
			E' \$	5		Saída: $T'$		
			9	: [		Saída: E'		

· · u cu · · · c					3.0			
Não-				Si	mbolo de En	trada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$	'			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$		sinc			$T \rightarrow FT'$	sinc	sinc
T'		7	$\lambda' \rightarrow \lambda$	T'	$\rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \qquad F  o id$		sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	mulação: Casamento		Pilha	$\top$	Entrada	Açã	ão .	
•			ES		+id * + id\$			
			ES		id * + id\$	⇒ Pular: ·		
			TE'S		id * + id\$	Saída: E -		
			FT'E'S		id * + id\$	Saída: T -	$\rightarrow$ FT $'$	
			id <i>T' E'</i> \$		id*+id\$	Saída: F -	→ id	
	id		T'E'		* + id\$	⇒ Casame		
	id		*FT'E'\$		* + id\$	Saída: T'	$\rightarrow *FT'$	
	id *		FT'E'S		+ id\$	⇒ Casame	ento: *	
			T'E'9	5		⇒ Desemp		
			E' 9			Saída: T'		
			+TE'	5		Saída: E'		
			TE'S	5		$\Rightarrow$ Casame		
			FT' E' \$	5		Saída: T -		
			id <i>T'E'</i> \$	5		Saída: F -		
			T'E'	5		⇒ Casame		
			E' \$	5		Saída: $T^\prime$		
			4			Saída: E'		

Não-				Símbolo de E	ntrada		
Terminal	id		+	*	(	)	\$
E	$E \rightarrow TE'$				$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$	sinc			$T \rightarrow FT'$	sinc	sinc
T'	T'		$\lambda' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$		sinc	sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha	Entrada	Aç	ão	
,			ES	+id * + id	6		
			ES		⇒ Pular:	+	
			TE'S		Saída: E	$\rightarrow TE'$	
			FT'E'S	id * + id	Saída: T	$\rightarrow$ FT $'$	
			id T' E' S	id * + id	Saída: F	→ id	
	id		T'E'S	* + id	⇒ Casam	ento: id	
	id		*FT'E'S	* + id9	Saída: T'	$\rightarrow *FT'$	
	id *		FT'E'S	+ id9	⇒ Casam	ento: *	
			T'E'S	+ id9	⇒ Desem		
			E' 9	+ id9	Saída: T'		
			+TE'	+ id9	Saída: E'		
			TE'S	id9	⇒ Casam		
			FT'E'S	ids	Saída: T		
			idT'E'S	ids	Saída: F		
			T'E'S	5 9	⇒ Casam		
			E'S	5 9	Saída: T'		
			5	5 9	Saída: E'		

Tracamic	iito ac E	0.			PIO			
Não-				S	ímbolo de En	trada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$		sinc		$T \rightarrow FT'$	sinc	sinc	
T'		7	$\lambda' \rightarrow \lambda$	T	$' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F			sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	ulação: Casamento		Pilha		Entrada	Açã	io	
,			E	5	+id * + id\$			
			ES		id * + id\$	⇒ Pular: ·	+	
			TE'S		id * + id\$	Saída: E -		
			FT'E'S		id * + id\$	Saída: T -	$\rightarrow FT'$	
			id <i>T' E'</i> \$		id * + id\$	Saída: F -	→ id	
	id		T'E'		* + id\$	⇒ Casame	ento: id	
	id		*FT'E'\$		* + id\$	Saída: T'	$\rightarrow *FT'$	
	id *		FT'E'S		+ id\$	⇒ Casame	ento: *	
	id		T'E'	<b>5</b>	+ id\$	⇒ Desemp	oilhar: F	
			E' 9	5		Saída: T'		
			+TE'	5		Saída: E'		
			TE'	5		$\Rightarrow$ Casame		
			FT'E'	5		Saída: T -		
			id <i>T'E'</i> \$	5		Saída: F -		
			T'E'	5		⇒ Casame		
			E' \$	5		Saída: $T^\prime$		
			9			Saída: F		

Não-					Símbolo de En	trada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'	T   T   FT'		$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T			sinc			$T \rightarrow FT'$	sinc	sinc
T'			$\lambda' \rightarrow \lambda$	7	$\Gamma'  o *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$		sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha		Entrada	Açã	io	
,			ES	5	+id * + id\$			
			ES	5	id * + id\$	⇒ Pular: ·	+	
			TE'S		id * + id\$	Saída: E -	→ TE′	
			FT'E'S	5	id * + id\$	Saída: T -	$\rightarrow FT'$	
			id T' E' 9	5	id * + id\$	Saída: F -	→ id	
	id		T'E'S	5	* + id\$	⇒ Casame	ento: id	
	id		* $FT'E'$ \$   * + id\$   Saída: $T' \rightarrow *FT'$				$\rightarrow *FT'$	
	id *		FT'E'S	5	+ id\$	⇒ Casame	ento: *	
	id		T'E'S	5	+ id\$	⇒ Desemp	oilhar: F	
			E' 9	5	+ id\$	Saída: T'		
			+TE'	5		Saída: E'		
			TE'S	5		⇒ Casame		
			FT'E'S	5		Saída: T -		
			idT'E'	8		Saída: F -		
			T'E'S	8		⇒ Casame		
			E'S	8		Saída: T'		
			5	5		Saída: E'		

Tracamic	iito ac E	0			PIO			
Não-				5	Símbolo de En	trada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$	İ	sinc			T  o FT'	sinc	sinc
T'	$F \qquad F  o id$		$\lambda' \rightarrow \lambda$		$' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
			sinc		sinc	$F \rightarrow (E)$		
Simulação:	Casamento		Pilha		Entrada	Açã	io	
,			ES	5	+id * + id\$			
			ES		id * + id\$	⇒ Pular: ·	+	
			TE'S		id * + id\$	Saída: E -		
			FT'E'S		id * + id\$	Saída: T -	$\rightarrow FT'$	
			id <i>T' E'</i> \$		id * + id\$	Saída: F -	→ id	
	id		T'E'		* + id\$	⇒ Casame	ento: id	
	id		*FT'E'\$		* + id\$	Saída: T'	$\rightarrow *FT'$	
	id *		FT'E'S		+ id\$	⇒ Casame	ento: *	
	id		T'E'		+ id\$	⇒ Desemp	oilhar: F	
	id		E' 9		+ id\$	Saída: T'	$\rightarrow \lambda$	
			+TE'	5		Saída: E'		
			TE'S	5		⇒ Casame		
			FT'E'	5		Saída: T -		
			id <i>T′ E′</i> \$	5		Saída: F -		
			T'E'	5		$\Rightarrow$ Casame		
			E' 9	5		Saída: $T'$		
			9			Saída: F		

				-۱۰۰۳	. •			
Não-				Sín	nbolo de En	ntrada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$		sinc			$T \rightarrow FT'$	sinc	sinc
T'			$\lambda' \rightarrow \lambda$	T' -	$\rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$		sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha		Entrada	Açã	io	
,			ES	+ 1	-id * + id\$			
			ES		$\mathrm{id} * + \mathrm{id} \$$	⇒ Pular: ·	+	
			TE'S		id*+id\$	Saída: E -		
			FT'E'S		$\mathrm{id} * + \mathrm{id} \$$	Saída: T -	$\rightarrow FT'$	
			id T' E' 9		$\mathrm{id} * + \mathrm{id} \$$	Saída: F -	→ id	
	id		T'E'S		* + id\$	⇒ Casame		
	id		*FT'E'S		* + id\$	Saída: T'	$\rightarrow *FT'$	
	id *		FT'E'S		+ id\$	⇒ Casame	ento: *	
	id		T'E'S		+ id\$	⇒ Desemp		
	id		E' 9		+ id\$	Saída: T'	$\rightarrow \lambda$	
			+TE'S	5	+ id\$	Saída: E'		
			TE'S	5		⇒ Casame		
			FT'E'S	5		Saída: T -		
			idT'E'S	5		Saída: F -		
			T'E'	5		⇒ Casame		
			E'S	5		Saída: $T'$		
			5	5		Saída: E'		

#### Tratamento de Erros – Exemplo

Tracamic	iito ac E	0			ipio			
Não-					Símbolo de En	trada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$	İ	sinc			T  o FT'	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	7	$\Gamma' \to *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$		sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha	П	Entrada	Açã	ão .	
•			ES	5	+id * + id\$			
			ES		id * + id\$	⇒ Pular: ·	+	
			TE'S		id * + id\$	Saída: E -	→ TE′	
			FT'E'	5	id * + id\$	Saída: T -	$\rightarrow FT'$	
			id <i>T' E'</i> 9	5	id * + id\$	Saída: F -	→ id	
	id		T'E'9		* + id\$	⇒ Casame	ento: id	
	id		*FT'E'\$		* + id\$	Saída: T'	$\rightarrow *FT'$	
	id *		FT'E'S		+ id\$	⇒ Casame	ento: *	
	id		T'E'	5	+ id\$	⇒ Desemp	oilhar: F	
	id		E' 9		+ id\$	Saída: T'		
	id		+TE'	5	+ id\$	Saída: E'	$\rightarrow +TE'$	
			TE'S	5		⇒ Casame		
			FT'E'S	5		Saída: T -		
			id <i>T′E′</i> \$	5		Saída: F -		
			T'E'	5		⇒ Casame		
			E' 9	5		Saída: $T'$		
			9			Saída: F		

## Tratamento de Erros – Exemplo

Não-				:	Símbolo de En	trada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$	İ	sinc			T  o FT'	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	7	$^{\prime} \rightarrow *FT^{\prime}$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$		sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casamento		Pilha	Т	Entrada	Aç	ão .	
,			ES		+id*+id\$			
			ES		id * + id\$	⇒ Pular:	+	
			TE'S		id * + id\$	Saída: E -	$\rightarrow TE'$	
			FT'E'S		id * + id\$	Saída: T -	$\rightarrow FT'$	
			id T' E' 9	5	id * + id\$	Saída: F -	→ id	
	id		T'E'S	\$	* + id\$	⇒ Casame	ento: id	
	id		*FT'E'	5	* + id\$	Saída: T'	$\rightarrow *FT'$	
	id *		FT'E'S	\$	+ id\$	⇒ Casame	ento: *	
	id		T'E'S	5	+ id\$	⇒ Desemp	oilhar: F	
	id		E'S	\$	+ id\$	Saída: T'		
	id		+TE'S	5	+ id\$	Saída: E'	$\rightarrow +TE'$	
	id +		TE'S	5	id\$	⇒ Casame	ento: +	
			FT'E'S	5		Saída: T -		
			idT'E'	5		Saída: F -		
			T'E'	5		⇒ Casame		
			E'S	5		Saída: $T'$		
			9	8		Saída: E'		

#### Tratamento de Erros – Exemplo

	ratarric	iito ac L			יקייי				
	Não-				Sím	ibolo de En	itrada		
	Terminal	id		+		*	(	)	\$
	E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
	E'		E' -	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
	T	$T \rightarrow FT'$		sinc			$T \rightarrow FT'$	sinc	sinc
	T'		T	$^{\prime} \rightarrow \lambda$	T' -	$\rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
	F	$F \rightarrow id$		sinc		sinc	$F \rightarrow (E)$	sinc	sinc
S	imulação:	Casam	ento	Pilha		Entrada	Açã	ão .	
	•			E\$	+	id * + id\$			
				E\$		id * + id\$	⇒ Pular: ·	+	
				TE'\$		id * + id\$	Saída: E -	$\rightarrow TE'$	
				FT'E'\$		id * + id\$	Saída: T -	$\rightarrow FT'$	
				id <i>T' E'</i> \$		id * + id\$	Saída: F -	→ id	
		id		T'E'\$		* + id\$	⇒ Casame	ento: id	
		id		*FT'E'\$		* + id\$	Saída: T'	$\rightarrow *FT'$	
		id *		FT'E'\$		+ id\$	⇒ Casame	ento: *	
		id		T'E'\$	İ	+ id\$	⇒ Desemp	oilhar: F	
		id		E'\$		+ id\$	Saída: T'	$\rightarrow \lambda$	
		id		+TE'\$		+ id\$	Saída: E'	$\rightarrow +TE'$	
		id +		TE'\$	İ	id\$	⇒ Casame	ento: +	
				FT'E'\$		id\$	Saída: T -		
				id <i>T' E'</i> \$			Saída: F -		
				T'E'\$			⇒ Casame		
				E'\$			Saída: $T'$		
				8			Saída: F		

#### Tratamento de Erros – Exemplo

Tracarric	iito ac E	0			PIO			
Não-					Símbolo de En	trada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E'	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$	İ	sinc			$T \rightarrow FT'$	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	T	$' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$		sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha		Entrada	Açã	ão .	
•			ES	5	+id * + id\$			
			ES		id * + id\$	⇒ Pular: ·		
			TE'S		id * + id\$	Saída: E -	→ TE′	
			FT'E'	5	id * + id\$	Saída: T -	$\rightarrow FT'$	
			id <i>T' E'</i> 9	5	id * + id\$	Saída: F -	→ id	
	id		T'E'		* + id\$	⇒ Casame	ento: id	
	id		*FT'E'	5	* + id\$	Saída: T'	$\rightarrow *FT'$	
	id *		FT'E'		+ id\$	⇒ Casame	ento: *	
	id		T'E'	5	+ id\$	⇒ Desemp	oilhar: F	
	id		E' 9		+ id\$	Saída: T'		
	id		+TE'	\$	+ id\$	Saída: E'	$\rightarrow +TE'$	
	id +		TE'S	5	id\$	⇒ Casame	ento: +	
	id +		FT'E'	\$	id\$	Saída: T -	$\rightarrow FT'$	
			id <i>T'E'</i> \$	5		Saída: F -		
			T'E'	5		⇒ Casame		
			E' \$	5		Saída: $T'$		
			9	8		Saída: F'		

#### Tratamento de Erros – Exemplo

riatarric	iito ac L				ipio				
Não-					Símbolo de En	itrada			
Terminal	id		+		*	(	)	\$	
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc	
E'		E' -	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$	
T	$T \rightarrow FT'$		sinc			$T \rightarrow FT'$	sinc	sinc	
T'		T	$^{\prime} \rightarrow \lambda$	7	$\Gamma'  o *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$	
F	$F \rightarrow id$		sinc		sinc $F \rightarrow (E)$ sinc		sinc	sinc	
Simulação:	Casam	ento	Pilha		Entrada	Aç	Ação		
,			ES	5	+id * + id\$				
			ES		id * + id\$	⇒ Pular:	+		
			TE'		id * + id\$	Saída: E -	$\rightarrow TE'$		
			FT'E'		id * + id\$	Saída: T -	$\rightarrow FT'$		
			id <i>T' E'</i> 9	5	id * + id\$	Saída: F -	→ id		
	id		T'E'	5	* + id\$	⇒ Casame	ento: id		
	id		*FT'E'	5	* + id\$	Saída: T'	$\rightarrow *FT'$		
	id *		FT'E'		+ id\$	⇒ Casame	ento: *		
	id		T'E'	5	+ id\$	⇒ Desem	oilhar: F		
	id		E' \$		+ id\$	Saída: T'	$\rightarrow \lambda$		
	id		+TE'	5	+ id\$	Saída: E'	$\rightarrow +TE'$		
	id +		TE'S	5	id\$	⇒ Casame	ento: +		
	id +		FT'E'	5	id\$	Saída: T -	$\rightarrow FT'$		
			id <i>T' E'</i> \$	5	id\$	Saída: F -			
			T'E'	5		⇒ Casame			
			E' 9	5		Saída: $T'$			
			9	5		Saída: E'			

#### Tratamento de Erros – Exemplo

Não-					Símbolo de En	ntrada		
Terminal	id		+		*	(	)	\$
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
E'		E' -	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
T	$T \rightarrow FT'$		sinc			$T \rightarrow FT'$	sinc	sinc
T'		T	$\lambda' \rightarrow \lambda$	7	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
F	$F \rightarrow id$		sinc		sinc	$F \rightarrow (E)$	sinc	sinc
Simulação:	Casam	ento	Pilha		Entrada	Aç	io	
•			ES	5	+id * + id\$			
			ES		id * + id\$	⇒ Pular:	+	
			TE'	5	id * + id\$	Saída: E -	→ TE′	
			FT'E'		id * + id\$	Saída: T -	$\rightarrow FT'$	
			id <i>T' E'</i> \$	5	id * + id\$	Saída: F -	→ id	
	id		T'E'		* + id\$	⇒ Casame	ento: id	
	id		*FT'E'	5	* + id\$	Saída: T'	$\rightarrow *FT'$	
	id *		FT'E'	5	+ id\$	⇒ Casame	ento: *	
	id		T'E'	5	+ id\$	⇒ Desem	oilhar: F	
	id		E' \$	5	+ id\$	Saída: T'		
	id		+TE'	5	+ id\$	Saída: E'	$\rightarrow +TE'$	
	id +		TE'S	5	id\$	⇒ Casame	ento: +	
	id +		FT'E'	5	id\$	Saída: T -	$\rightarrow FT'$	
	id +		id <i>T' E'</i> \$	5	id\$	Saída: F -	→ id	
			T'E'	5		⇒ Casame		
			E' 9	5		Saída: $T'$		

#### Tratamento de Erros – Exemplo

Não-		Símbolo de Entrada							
Terminal	id		+		*	(	)	\$	
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc	
E'		E' -	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$	
T	$T \rightarrow FT'$		sinc			T  o FT'	sinc	sinc	
T'		T	$' \rightarrow \lambda$	T	$' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$	
F	$F \rightarrow id$		sinc		sinc	$F \rightarrow (E)$	sinc	sinc	
Simulação:	Casam	ento	Pilha	Т	Entrada	Açã	ão .		
•			E\$	5	+id * + id\$				
			E\$		id * + id\$	⇒ Pular: ·	+		
			TE'\$	5	id * + id\$	Saída: E -	→ TE′		
			FT'E'\$	5	id * + id\$	Saída: T -	$\rightarrow FT'$		
			id <i>T′ E′</i> \$		id*+id\$	Saída: F -	→ id		
	id		T'E'\$	5	* + id\$	⇒ Casame	ento: id		
	id		*FT'E'\$		* + id\$	Saída: T'	$\rightarrow *FT'$		
	id *		FT'E'\$	5	+ id\$	⇒ Casame	ento: *		
	id		T'E'\$		+ id\$	⇒ Desemp	oilhar: F		
	id		E'\$		+ id\$	Saída: T'	$\rightarrow \lambda$		
	id		+TE'\$		+ id\$	Saída: E'	$\rightarrow +TE'$		
	id +		TE'\$		id\$	⇒ Casame	ento: +		
	id +		FT'E'\$		id\$	Saída: T -	$\rightarrow$ FT $'$		
	id +		id <i>T' E'</i> \$	5	id\$	Saída: F -	→ id		
	id + id		T'E'\$	5	\$	⇒ Casame	ento: id		
			E' \$	5		Saída: T'			

#### Tratamento de Erros – Exemplo

Não-		Símbolo de Entrada							
Terminal	id		+		*	(	)	\$	
E	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc	
E'		E'	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$	
T	$T \rightarrow FT'$		sinc			T  o FT'	sinc	sinc	
T'		T	$\lambda' \rightarrow \lambda$	T'	$' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$	
F	$F \rightarrow id$		sinc		sinc	$F \rightarrow (E)$	sinc	sinc	
Simulação:	Casam	ento	Pilha		Entrada	Açã	ão .		
•			E\$	5	+id*+id\$				
			E\$		id * + id\$	⇒ Pular: ·	+		
			TE'\$	5	id * + id\$	Saída: E -	$\rightarrow TE'$		
			FT'E'\$		id * + id\$	Saída: T -	$\rightarrow FT'$		
			id <i>T′ E′</i> \$	5	id * + id\$	Saída: F -	→ id		
	id		T'E'\$	5	* + id\$	⇒ Casame	ento: id		
	id		*FT'E'\$		* + id\$	Saída: T'	$\rightarrow *FT'$		
	id *		FT'E'\$		+ id\$	⇒ Casame	ento: *		
	id		T'E'\$		+ id\$	⇒ Desemp	oilhar: <i>F</i>		
	id		E'\$		+ id\$	Saída: T'			
	id		+TE'\$		+ id\$	Saída: E'	$\rightarrow +TE'$		
	id +		TE'\$		id\$	⇒ Casame	ento: +		
	id +		FT'E'\$		id\$	Saída: T -	$\rightarrow$ FT $'$		
	id +		id <i>T' E'</i> \$	5	id\$	Saída: F -	→ id		
	id + id		T'E'\$	5	\$	⇒ Casame	ento: id		
			E'\$	5	\$	Saída: T'			

#### Tratamento de Erros – Exemplo

Não-		Símbolo de Entrada								
Terminal	id		+	*	(	)	\$			
E	$E \rightarrow TE'$				$E \rightarrow TE'$	sinc	sinc			
E'		E' -	$\rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$			
T	$T \rightarrow FT'$		sinc		$T \rightarrow FT'$	sinc	sinc			
T'		T	$\lambda' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$			
F	$F \rightarrow id$		sinc	sinc	$F \rightarrow (E)$	sinc	sinc			
Simulação:	Casam	ento	Pilha	Entrada	Ac	ão.				

$\sim$	~
Simul	コーコー
Jiiiiu	acao.

Casamento	i iiiia	Liitiaua	nça0
	E\$	+id * + id\$	
	E\$	id * + id\$	⇒ Pular: +
	TE'\$	id * + id\$	Saída: $E \rightarrow TE'$
	FT'E'\$	id * + id\$	Saída: $T \rightarrow FT'$
	id <i>T' E'</i> \$	id * + id\$	Saída: $F \rightarrow id$
id	T'E'\$	* + id\$	⇒ Casamento: id
id	*FT'E'\$	* + id\$	Saída: $T' \rightarrow *FT'$
id *	FT'E'\$	+ id\$	⇒ Casamento: *
id	T'E'\$	+ id\$	⇒ Desempilhar: F
id	E'\$	+ id\$	Saída: $T'  o \lambda$
id	+TE'\$	+ id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id\$	⇒ Casamento: +
id +	FT'E'\$	id\$	Saída: $T \rightarrow FT'$
id +	id <i>T' E'</i> \$	id\$	Saída: $F \rightarrow id$
id + id	T'E'\$	\$	⇒ Casamento: id
id + id	E'\$	\$	Saída: $T'  o \lambda$
	\$	\$	Saída: $E'  o \lambda$

#### Tratamento de Erros – Exemplo

Tratamento de Erros Exemplo											
Não-		Símbolo de Entrada									
Terminal	id	+	*	(	)	\$					
Ε	$E \rightarrow TE'$			$E \rightarrow TE'$	sinc	sinc					
E'		$E' \rightarrow +TE'$			$E' \rightarrow \lambda$	$E' \rightarrow \lambda$					
T	$T \rightarrow FT'$	sinc		$T \rightarrow FT'$	sinc	sinc					
T'		$T' \rightarrow \lambda$	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$					
F	$F \rightarrow id$	sinc	sinc	$F \rightarrow (E)$	sinc	sinc					
	Não- Terminal	$ \begin{array}{c c} \textbf{N\"{ao}-} \\ \hline \textbf{Terminal} & \textbf{id} \\ \hline \textbf{\textit{E}} & \textbf{\textit{E}} \rightarrow \textbf{\textit{TE}}' \\ \textbf{\textit{E'}} & \\ \textbf{\textit{T}} & \textbf{\textit{T}} \rightarrow \textbf{\textit{FT}}' \\ \textbf{\textit{T'}} & \\ \textbf{\textit{F}} & \textbf{\textit{F}} \rightarrow \textbf{id} \\ \end{array} $	$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$	$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$	$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$	$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$					

Simulação:

Casamento	Pilha	Entrada	Ação
	E\$	+id * + id\$	
	E\$	id * + id\$	⇒ Pular: +
	TE'\$	id * + id\$	Saída: $E \rightarrow TE'$
	FT'E'\$	id * + id\$	Saída: $T \rightarrow FT'$
	id <i>T' E'</i> \$	id * + id\$	Saída: $F \rightarrow id$
id	T'E'\$	* + id\$	⇒ Casamento: id
id	*FT'E'\$	* + id\$	Saída: $T' \rightarrow *FT'$
id *	FT'E'\$	+ id\$	⇒ Casamento: *
id	T'E'\$	+ id\$	⇒ Desempilhar: F
id	E'\$	+ id\$	Saída: $T' \rightarrow \lambda$
id	+TE'\$	+ id\$	Saída: $E' \rightarrow +TE'$
id +	TE'\$	id\$	⇒ Casamento: +
id +	FT'E'\$	id\$	Saída: $T \rightarrow FT'$
id +	id <i>T' E'</i> \$	id\$	Saída: $F \rightarrow id$
id + id	T'E'\$	\$	⇒ Casamento: id
id + id	E'\$	\$	Saída: $T'  o \lambda$
id + id	\$	\$	Saída: $E'  o \lambda$

Saída:  $T \rightarrow FT'$ 

⇒ Casamento: id

Saída:  $T' o \lambda$ 

Saída:  $E' \rightarrow \lambda$ 

Saída:  $F \rightarrow id$ 

## Parsing Preditivo Não-Recursivo

#### Tratamento de Erros – Exemplo

id +

id + id

id + id

id + id

al			+		*	(	)	\$
	$E \rightarrow TE'$					$E \rightarrow TE'$	sinc	sinc
		E' -	$\rightarrow +TE'$				$E' \rightarrow \lambda$	$E' \rightarrow \lambda$
	$T \rightarrow FT'$		sinc			$T \rightarrow FT'$	sinc	sinc
İ		T	$' \rightarrow \lambda$	7	$T' \rightarrow *FT'$		$T' \rightarrow \lambda$	$T' \rightarrow \lambda$
	F  o id		sinc		sinc	$F \rightarrow (E)$	sinc	sinc
ão:	Casam	ento	Pilha		Entrada	Aç	ão .	
			ES	5	+id * + id\$			
			ES	5	id * + id\$	⇒ Pular:	+	
			TE'S	5	id * + id\$	Saída: E -	$\rightarrow TE'$	
			FT'E'	5	id * + id\$	Saída: T -	$\rightarrow FT'$	
			id T' E' 9	;	id * + id\$	Saída: F -	→ id	
	id		T'E'	5	* + id\$	⇒ Casame	ento: id	
	id		*FT'E'	5	* + id\$	Saída: T'	$\rightarrow *FT'$	
	id *		FT'E'S	;	+ id\$	⇒ Casame	ento: *	
	id		T'E'	;	+ id\$	⇒ Desemp	oilhar: F	
	id		E' 9	5	+ id\$	Saída: T'	$\rightarrow \lambda$	
	id		+TE'	;	+ id\$	Saída: E'	$\rightarrow +TE'$	
	id +		TE'	;	id\$	⇒ Casame	ento: +	
	áo:	E  o TE' $T  o FT'$ $F  o id$ O: Casame  id  id  id  id  id  id  id  id	$E \rightarrow TE'$ $T \rightarrow FT'$ $F \rightarrow id$ $Casamento$ $id$ $id$ $id$ $id$ $id$ $id$ $id$ $id$	$E \rightarrow TE'$ $T \rightarrow FT'$ $F \rightarrow id$ $Casamento$ $Casamento$ $ES$ $F'' = S$ $id$ $T' = S$ $T' $	$E \rightarrow TE' \\ T \rightarrow FT' \\ F \rightarrow id \\ Sinc \\ T' \rightarrow \lambda \\ Sinc \\ Sinc \\ T' \rightarrow \lambda \\ Sinc \\ OS$ $Casamento                                    $	$E \rightarrow TE' \\ T \rightarrow FT' \\ F \rightarrow id \\ \hline Casamento \\ Casamento \\ \hline Casamento \\ Casamento \\ \hline Casamento \\ Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ \hline Casamento \\ Casamento \\ \hline Casamento \\ Casamento \\ \hline Casamento \\ C$	$E \rightarrow TE' \\ T \rightarrow FT' \\ F \rightarrow id \\ Sinc \\ T' \rightarrow \lambda \\ Sinc \\ T' \rightarrow *FT' \\ F \rightarrow id \\ Sinc \\ T' \rightarrow \lambda \\ Sinc \\ T' \rightarrow *FT' \\ F \rightarrow (E) \\ \hline AC: Casamento   Pilha   Entrada   Açi   Es   +id * +id *   +id * +id *   +id * +id *   +id * +id *   +id * +id *   +id * +id *   +id *   +id * +id *   +id *$	$E \rightarrow TE' \\ T \rightarrow FT' \\ Sinc \\ T' \rightarrow \lambda \\ F \rightarrow id \\ Sinc \\ T' \rightarrow \lambda \\ Sinc \\ T' \rightarrow \lambda \\ Sinc \\ T' \rightarrow \lambda \\ Sinc \\ T' \rightarrow \lambda \\ Sinc \\ T' \rightarrow \lambda \\ F \rightarrow (E) \\ Sinc \\ T' \rightarrow \lambda \\ Sinc \\ T' \rightarrow \lambda \\ Sinc \\ F \rightarrow (E) \\ Sinc \\ $

FT'E'\$

idT'E'\$

T'E'\$

E'\$

Símbolo de Entrada

Parsing Desc. Não-Rec.

Não-

id\$