Error detection

EDR · LL(1) Parse

$$E \rightarrow \cdot TE'$$
$$E' \rightarrow +TE' \cdot | \varepsilon$$
$$T \rightarrow \cdot E T$$
$$T' \rightarrow *E T' / \varepsilon$$
$$F \rightarrow \cdot (E) \cdot / id$$

	first	follow
E	$($, id	$)$, $
E'	+, ε	$)$, $
T	$($, id	+, $)$, $
T'	*, ε	+, $)$, $
F	$($, id	*, +, $)$, $

	+	*	(	id	)	$
E			$E \rightarrow T \cdot E'$	$E \rightarrow TE'$		
E'	$E' \rightarrow +TE'$				$E' \rightarrow \varepsilon$	$E' \rightarrow$
T			$T \rightarrow FT'$	$T \rightarrow FT'$		
T'	$T' \rightarrow \varepsilon$	$T \rightarrow *F \cdot T'$			$T' \rightarrow \varepsilon$	$T' \rightarrow$
F			$F \rightarrow (E)$	$F \rightarrow id$		

(2) // STRING : id * + id + * id

INPUT	STACK	ACTION
$ id * + id + * id	E , $	M[E,id] = E → TE'
$ id * + id + * id	T E' $	M[T,id] = T → FT'
$ id * + id + * id	F T' E' $	M[F,id] = F → id
$ id * id + * id	id T' E' $	POP id from stack & move point
$ id * + id + *	T' E' $	[T', *] = T' → * F T'
$ id * + id + +	* F T' E' $	POP * from stack & move pointer
$ i d * + id +	F T' E' $	M[F, +] = synon, POP F
$ id * + id +	T' E' $	M[T', +] = T' → ε
$ id * + id +	E' $	M[E', +] = E' → + TE'
$ id * + id +	+ T E' $	POP + from stack & move pointer
$ id * + id	T E' $	M[T, id] = T → FT'
$ id * + id	F T' E' E' $	M[F, id] = F → id
$ id * + id	id T' E' $	POP id from stack, move point
$ id * +	T' E' $	M[T', +] = T' → ε
$ id * +	E' $	M[E', +] = E' → + TE'
$ id * +	+ T E' $	POP + from stack, move pointer
$ id *	T E' $	M[T, *] = NULL	Skip i/p symbol
$ id	T E' $	M[T, id] = T → FT'
$ id	F T' E' $	M[F, id] = F → id
$ id	id T' E' $	POP id from stack, move pointer
$	T' E' $	M[T', $] = T' → ε
$	E' $	M[E, $] = E' → ε
$	$	ACCEPT

STRING ACCEPTED

STRING : id id + id * id.

INPUT	STACK	ACTION.
$ id * id + id id	E $	M[E, id] = E → TE'
$ id * id + id id	TE' $	M[T, id] = T → FT'
$ id * id + id id	FT'E' $	M[F, id] = F → id
$ id * id + id id	id T'E' $	POP id from stack & move pointer
$ id * id + id	T'E' $	M[T', id] = NULL	skip i/p symbol
$ id * id +	T'E' $	M[T', +] = T' → ε
$ id * id +	E'$	M[E', +] = E' → + TE'
$ id * id +	+TE' $	POP + from stack & move pointer
$ id * id	TE'$	M[T, id] = T → FT'
$ id * id	FT'E' $	M[F, id] = F → id
$ id * id	id T'E'$	POP id from stack & move pointer
$ id *	T'E' $	M[T', *] = T' → * FT'
$ id *	*FT'E'$	POP * from stack & move pointer
$ id	FT'E' $	[F, id] = F → id
$ id	id T'E'$	POP id from stack & move pointer
$	T'E' $	M[T', $] = T' → ε
$	E' $	M[E', $] = E' → ε
$	$	ACCEPT

STRING IS ACCEPTED

$S \longrightarrow AbS \mid e \mid \varepsilon$

$A \longrightarrow a \mid cAd$

string is aab.

	FIRST	FOLLOW	
S	a, c, e, ε	$\$$	
A	a, c	b, d	

	a	b	c	d	e	$\$$
S	$S \longrightarrow AbS$		$S \rightarrow AbS$		$S \rightarrow e$	$S \rightarrow \varepsilon$
A	$A \longrightarrow a$		$A \rightarrow cAd$			

	a	b	c	d	e	$\$$
S	$S \rightarrow AbS$		$S \rightarrow AbS$		$S \rightarrow e$	$S \rightarrow \varepsilon$
A	$A \rightarrow a$	SYNCH	$A \rightarrow cAd$	SYNCH		

// STRING : aab //

INPUT	STACK	ACTION
$\$baa$	$S\$$	$M[S, a] = S \longrightarrow AbS$
$\$baa$	$AbS\$$	$M[A, a] = A \longrightarrow a$
$\$baa$	$abS\$$	POP a from stack
$\$ba$	$bS\$$	NOT ACCEPTED

// STRING "NOT" ACCEPTED. //