

Given Grammar (Left Recursive)	Left Recursion Eliminated Grammar
$E \rightarrow E + T \mid T$ $T \rightarrow T * F \mid F$ $F \rightarrow (E) \mid \text{id}$	$E \rightarrow T E'$ $E' \rightarrow + T E' \mid \varepsilon$ $T \rightarrow F T'$ $T' \rightarrow * F T' \mid \varepsilon$ $F \rightarrow (E) \mid \text{ID}$

Nonterminal	Nullable	FIRST	FOLLOW
E	No	{ (, id }	{ \$,) }
E'	Yes	{ + }	{ \$,) }
T	No	{ (, id }	{ \$,) , + }
T'	Yes	{ * }	{ \$,) , + }
F	No	{ (, id }	{ *, \$,) , + }

Parse Table

	+	*	()	id	\$
<i>E</i>			<i>E</i> → <i>T E'</i>		<i>E</i> → <i>T E'</i>	
<i>E'</i>	<i>E'</i> →+ <i>T E'</i>			<i>E'</i> →ε		<i>E'</i> →ε
<i>T</i>			<i>T</i> → <i>F T'</i>		<i>T</i> → <i>F T'</i>	
<i>T'</i>	<i>T'</i> →ε	<i>T'</i> →* <i>F T'</i>		<i>T'</i> →ε		<i>T'</i> →ε
<i>F</i>			<i>F</i> →(<i>E</i>)		<i>F</i> → id	

Example (Predictive Parsing)

Stack	Input
\$ E	(id + id) * id \$
\$ E' T	(id + id) * id \$
\$E'T'F	(id + id) * id \$
\$E'T')E((id + id) * id \$
\$ E' T') E	id + id) * id \$
\$ E' T') E' T	Id + id) * id \$
\$ E' T') E' T' F	id + id) * id \$
\$ E' T') E' T' id	id + id) * id \$
\$ E' T') E' T'	+ id) * id \$
\$ E' T') E'	+ id) * id \$
\$ E' T') E' T +	+ id) * id \$
\$ E' T') E' T	id) * id \$
\$ E' T') E' T' F	id) * id \$
\$ E' T') E' T' id	id) * id \$
\$ E' T') E' T') * id \$
\$ E' T') E') * id \$
\$ E' T')) * id \$
\$ E' T'	* id \$
\$ E' T' F *	* id \$
\$ E' T' F	id \$
\$ E' T' id	id \$
\$ E' T'	\$
\$ E'	\$
\$	\$