INTEGER	→	0 | [1-9][0-9]*
LPAREN	→	(
RPAREN	→	)
PLUS	→	+
MINUS	→	-
STAR	→	*
SLASH	→	/

Expression	→	*Term* { *AddOp Term* } *
Term	→	*Factor* { *MulOp Factor* } *
Factor	→	*IntLiteral* | LPAREN *Expression* RPAREN
IntLiteral	→	INTEGER
AddOp	→	PLUS | MINUS
MulOp	→	STAR | SLASH

```
options {
  JAVA_UNICODE_ESCAPE = true;
}

PARSER_BEGIN(MathexParser)

public class MathexParser
{}

PARSER_END(MathexParser)

/* WHITE SPACE */

SKIP :
{
  " "
| "\t"
| "\n"
| "\r"
| "\f"
}


/* TOKENS */

TOKEN:
{
  < INTEGER : ("0" | ["1"-"9"](["0"-"9"])*) >
|  < LPAREN: "(" >
| < RPAREN: ")" >
| < PLUS: "+" >
| < MINUS: "-" >
| < STAR: "x" >
| < SLASH: "/" >
}


/* MathEx Language Grammar */
```

```
void Expression():
{}
{
   Term() (AddOp() Term())*
}

void Term():
{}
{
 Factor() (MulOp() Factor())*
}

void Factor():
{}
{
   IntLiteral()
| <LPAREN> Expression() <RPAREN>
}

void IntLiteral():
{}
{
   <INTEGER>
}

void AddOp():
{}
{
   <PLUS>
|  <MINUS>
}

void MulOp():
{}
{
   <STAR>
|  <SLASH>
}
```