1. All of the productions that exhibit left-recursion:
   * expr\_prefix : expr\_prefix factor addop | ;
   * factor\_prefix : factor\_prefix postfix\_expr mulop | ;
2. To combat not knowing which operator is supposed to be in what order, so ANTLR rewrites the left-recursive rule into a non-recursive rule using precedence of previous operators to judge the precedence of the current operator. A good example of this is for a grammar rule for an expression of basic arithmetic, (e.g. 5+3\*(4-3)), it would come out in the order (4-3), 3\*(1), 5+3 = 8.code
3. ALL(\*) (“all star”), it’s an extension to LL(\*) that parses and analyzes grammar on the fly to have access to input sequences, rather than doing so statically, by accepting individual characters as input symbols. ALL(\*) parsers can recognize the order of input sequences by reading through the grammars. The ALL(\*) prediction mechanism launches subparsers at decision points to explore possible paths in each production.
4. Graphical user interface, text

   Description automatically generatedA screenshot of a computer

   Description automatically generated with medium confidenceANTLR generates methods for each rule defined in the grammar. These methods descend from the root of the generated parse tree to the leaves, comparing and consuming tokens from the top-down using a default match() method as well as other generated methods that define and match tokens. Decisions are made using case logic, to invoke the appropriate method when an alternative token is encountered.