EBNF Grammar for C--

(corrected)

```
1.  TranslationUnit     = { TypeSpecifier, identifier,
                            ( [ "[", integer, "]"],
                              { ",", identifier, [ "[", integer, "]" ] }, ";"
                            | "(", [ Parameter, { ",", Parameter } ], ")",
                              CompoundStatement
                            )
                          }
                        ;
2.  TypeSpecifier       = "void"
                        | "int"
                        | "float"
                        | "char"
                        ;
3.  Parameter           = TypeSpecifier, identifier, [ "[", "]" ]
                        ;
4.  CompoundStatement   = "{", { Declaration | Statement }, "}"
                        ;
5.  Declaration         = TypeSpecifier, identifier, [ "[", integer, "]" ],
                            { ",", identifier, [ "[", integer, "]" ] }, ";"
                        ;
6.  Statement           = ExpressionStatement
                        | CompoundStatement
                        | SelectionStatement
                        | RepetitionStatement
                        | ReturnStatement
                        ;
7.  ExpressionStatement = [ Expression, [ "=", Expression ] ], ";"
                        ;
8.  SelectionStatement  = "if", "(", Expression, ")", Statement, [ "else",
                          Statement ]
                        ;
9.  RepetitionStatement = "while", "(", Expression, ")", Statement
                        ;
10. ReturnStatement     = "return", [ Expression ], ";"
                        ;
11. Expression          = SimpleExpression,
                          [ ( "<=" | "<" | ">=" | ">" | "==" | "!=" ),
                            SimpleExpression ]
                        ;
12. SimpleExpression    = Term, { ( "+" | "-" ),  Term }
                        ;
13. Term                = Factor, { ( "*" | "/" ), Factor }
                        ;
14. Factor              = "(", Expression1, ")"
                        | identifier,
                              [ "[", Expression, "]"
                              | "(", [ Expression, { ",", Expression } ], ")"
                              ]
                        | integer
                        | float
                        | char
                        | string
                        ;
```