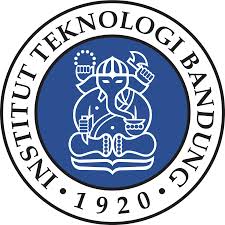
**EBNF and Parser Implementation of PHP Code**

**Task IF5020**

**Algorithm and Programming A**



By:

Maratus Sholihah (23518016)

Muhammad Hanifan (23518026)

Magister Informatika

Sekolah Teknik Elektronika dan Informatika

INSTITUT TEKNOLOGI BANDUNG

2018

1. **BNF Grammar of PHP**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Program** |  |  |
| 1 | <Program> | ::= | '<?php' {<block>} '?>' |
| 2 | <block> | ::= | <statement> | <function\_declaration\_statement> | <class\_declaration> |
|  | **Statement** |  |  |
| 3 | <statement> | ::= | '{' {<block>} '}' |  'switch' '(' <expr> ')' '{' <switch\_case> '}' |  'while' '(' <expr> ')' <inner\_statement> |  'for' '(' <expr\_c> ';' <expr\_c> ';' <expr\_c> ')' < inner\_statement > |  'if' '(' <expr> ')' <statement> {<elseif>} <else\_s> |  'do' < inner\_statement > 'while' '(' <expr> ')' ';' |  'echo' <expr\_c> ';' |  <expr> ';' |  'return' <expr> ';'|  'return' ';'|  ';' |
| 4 | <inner\_statement> | ::= | <statement> | 'break' |
| 5 | <elseif> | ::= | 'elseif' '(' <expr> ')' <statement> |
| 6 | <else\_s> | ::= | '' | 'else' <statement> |
| 7 | <switch\_case> | ::= | '' | {<case>} <default\_s> {<case>} |
| 8 | <case> | ::= | 'case' <expr> ':'{ <inner\_statement>} |
| 9 | <default\_s> | ::= | '' | 'default' ':' {<inner\_statement>} |
|  | **Class and Interface** |  |  |
| 10 | <class\_declaration> | ::= | <class\_entry\_type> < t\_string > <extends\_from> <implements\_list> '{' {<class\_statement>} '}' |'interface' < t\_string > <interface\_extends\_list>  '{' {<function\_declaration>} '}' |
| 11 | <class\_entry\_type> | ::= | 'abstract' 'class' | 'final' 'class' | 'class' |
| 12 | <extends\_from> | ::= | '' | 'extends' < t\_string > |
| 13 | <implement\_list> | ::= | '' | 'implements' <t\_string\_cc> |
| 14 | <interface\_extends\_list> | ::= | '' | 'extends' <t\_string\_cc> |
| 15 | <class\_statement> | ::= | <modifier> <assignment\_expr > ';' | <modifier> <function\_declaration\_statement> |
| 16 | <modifier> | ::= | 'public' | 'protected' | 'private' | 'static' | 'abstract' | 'final' | '' |
|  | **Function** |  |  |
| 17 | <function\_declaration\_statement> | ::= | <function\_declaration> <function\_statement> |
| 18 | <function\_declaration> | ::= | 'function' < t\_string > '(' <function\_parameter\_c> ')' |
| 19 | <function\_statement> | ::= | '{' <statement> '}' |
| 20 | <function\_parameter\_c> | ::= | '' | <function\_parameter\_cc> |
| 21 | <function\_parameter\_cc> | ::= | <function\_parameter>|<function\_parameter>','<function\_parameter\_cc> |
| 22 | <function\_parameter> | ::= | <assignment\_expr> | <t\_variable> |
|  | **Expression** |  |  |
| 23 | <expr\_c> | ::= | '' | <expr\_cc> |
| 24 | <expr\_cc> | ::= | <expr> | <expr> ',' <expr\_cc> |
| 25 | <expr> | ::= | <assignment\_expr> | <right\_hand > |
| 26 | <assignment\_expr> | ::= | <left\_hand> <assignment\_operator> <right\_hand> | <left\_hand> |
| 27 | <left\_hand> | ::= | <t\_variable> | <array\_access> | <object\_access> |
| 28 | <right\_hand\_c> | ::= | '' | <right\_hand\_cc> |
| 29 | <right\_hand\_cc> | ::= | <right\_hand> | <right\_hand> ',' <right\_hand\_cc> |
| 30 | <right\_hand> | ::= | <constant\_value> | <t\_variable> | <function\_call> | <array\_access> | <object\_access> |<t\_variable> '++' | <t\_variable> '--' | '++' <t\_variable> | |
|  |  |  | '--' <t\_variable> | <right\_hand> <operator> <right\_hand> |
| 31 | <function\_call> | ::= | <t\_string> '(' <right\_hand\_c> ')' |
| 32 | <array\_access> | ::= | <t\_variable> '[' ''' <t\_string> ''' ']' | <t\_variable> '[' <t\_variable> ']' | <t\_variable> '[' <number> ']' |
| 33 | <object\_access> | ::= | <t\_variable> '->' <t\_string> |
|  | **Operator** |  |  |
| 34 | <assignment\_operator> | ::= | '=' | '+=' | '-=' | '\*=' | '/=' | '%=' | '.=' |
| 35 | <operator> | ::= | '+' | '-' | '\*' | '/' | '%' | '\*\*' | |
|  |  |  | '==' | '===' | '!=' | '<>' | '!==' | '<' | '<=' | '>' | '>=' | |
|  |  |  | 'and' | 'or' | 'xor' | '&&' | '||' | '!' | '.' |
|  | **Others** |  |  |
| 36 | <t\_variable\_c> | ::= | '' | <t\_variable\_cc> |
| 37 | <t\_variable\_cc> | ::= | <t\_variable> | <t\_variable> ',' <t\_variable\_cc> |
| 38 | <t\_variable> | ::= | '$' < t\_string > |
| 39 | <t\_string\_c> | ::= | '' | <t\_string\_cc> |
| 40 | <t\_string\_cc> | ::= | <t\_string> | <t\_string> ',' <t\_string\_cc> |
| 41 | < t\_string > | ::= | '\_' {<var\_name>} | <letter> {<var\_name>} |
| 42 | <var\_name\_r> | ::= | '' | '\_' <var\_name\_r> | <letter> <var\_name\_r> | <digit> <var\_name\_r> |
| 43 | <constant\_value\_c> | ::= | '' | <constant\_value\_cc> |
| 44 | <constant\_value\_cc> | ::= | <constant\_value> | <constant\_value> ',' <constant\_value\_cc> |
| 45 | <constant\_value> | ::= | <constant\_value\_single> | <array> |
| 46 | <constant\_value\_single> | ::= | <number> | ''' <string> ''' | <boolean> |
| 47 | <array> | ::= | 'array' '(' <constant\_value\_c> ')' |'array' '(' <key\_value\_pair\_c> ')' |
| 48 | <key\_value\_pair\_c> | ::= | '' | <key\_value\_pair\_cc> |
| 49 | <key\_value\_pair\_cc> | ::= | <key\_value\_pair> | <key\_value\_pair> ',' <key\_value\_pair\_cc> |
| 50 | <key\_value\_pair> | ::= | <constant\_value\_single> '=>' <right\_hand> |
| 51 | <letter> | ::= | 'a' | 'b' | 'c' | 'd' | 'e' | 'f' | 'g' | 'h' | 'i' | 'j' | 'k' | 'l' | 'm' |  'n' | 'o' | 'p' | 'q' | 'r' | 's' | 't' | 'u' | 'v' | 'w' | 'x' | 'y' | 'z' |  'A' | 'B' | 'C' | 'D' | 'E' | 'F' | 'G' | 'H' | 'I' | 'J' | 'K' | 'L' | 'M' |  'N' | 'O' | 'P' | 'Q' | 'R' | 'S' | 'T' | 'U' | 'V' | 'W' | 'X' | 'Y' | 'Z' |
| 52 | <digit> | ::= | '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9' |
| 53 | <string> | ::= | {<input>} |
| 54 | <number> | ::= | <number\_sign> {<digit>} | <number\_sign> {<digit>} '.' {<digit>} |
| 55 | <number\_sign> | ::= | '' | '+' | '-' |
| 56 | <boolean> | ::= | 'true' | 'false' |

**Note:**

* \_s : single optional
* \_c : repeat with comma (,) separator