BNF Grammar for the language “**Lua**”

Shivam Bhagat 2015B5A70460H
Abhinav Kumar 2015B5A70674H
Yashdeep Thorat 2015B5A70675H
Rohan Jain 2015B4A70676H

```
<Program> ::= <Statements>
<Statements> ::= <Statement> `;´ <Statements> | <Statement>

<Statement> ::= <Conditional-statement> | <Loop-statement> | <Assign-statement> |
                <Function-statement> | <expression>

<Conditional-statement> ::= `if´( <expression> ) <Statements> `end´ |
                            `if´ ( <expression> ) <Statements> `else´ <Statements> `end´

<Loop-statement> ::= <while-stmt> | <for-stmt> | <repeat-stmt>
<while-stmt> ::= `while´ `(´ <expression> `)´ `do´ <Statements> `end´
<repeat-stmt> ::= `repeat´ <Statements> `until´ `(´ <expression> `)´

<Assign-statement> ::= <identifier> `=´ <expression>

<expression> ::= <expression> `or´ <or-term> | <or-term>
<or-term> ::= <or-term> `and´ <and-term> | <and-term>
<and-term> ::= <and-term> `<´ <rel-term> | <and-term> `>´ <rel-term> |
               <and-term> `<=´ <rel-term> | <and-term> `>=´ <rel-term> |
               <and-term> `~=´ <rel-term> | <and-term> `==´ <rel-term> | <rel-term>
<rel-term> ::= <rel-term>  `+´ <term> | <rel-term>  `-´ <term> | <term>
<term> ::= <term> `*´ <factor> | <term> `/´ <factor> | <term> `%´ <factor> | <factor>
<factor> ::= `not´ <un-term> | `#´ <un-term> | `-´ <un-unterm> | <un-term>
<un-term> ::= <un-term> `^´ <identifier> | <un-term> `^´ <literal> | <identifier> | <literal> |
              <Function-call>

<Function-statement> ::= `function´ <identifier> <func_body>
<func_body> ::= `(´<arguments>`)´ <Statements> <return-statement> `end´ |
                `(´ `)´ <Statements> <return-statement> `end´

<arguments> ::= <identifier> `,´ <arguments> | <identifier>
<return-statement> ::= `return´ <expression-list>
<expression-list> ::= <expression>`,´<expression-list> | <expression>

<Function-call> ::= <identifier> `(´<arguments>`)´ | <identifier> `(´ `)´

<identifier> ::= <letter><word>
<word> ::= <letter><word> | <digit><word> | ε

<literal> ::= <number-literal> | `true´ | `false´
<number-literal> ::= <digit><number-literal> | <digit>
<letter> ::= `a´ | `b´ | `c´ | ……… | `z´ | `A´ | `B´ | `C´ | ……… | `Z´ | `_´
<digit> ::= `0´ | `1´ | `2´ | …. | `9´
```