Syntax of BIOLA Programming Language

```
1. <S>   →  <program>

2. <program> →          an empty string         |
                        function < Function-name > ( < parameter-list> )
                        {
                                  <stmts>
                        }
                        <program>

3. <stmts>  →           an empty string  |   <stmt>   |   <stmt> <stmts>

4. <stmt>   →           <simple-stmt> |   <compound-stmt>

5. <simple-stmt> →      return <expr> ;

6. <simple-stmt> →      read   <var>;

7. <simple-stmt> →      display   <expr-or-string-list> ;          |

8. <simple-stmt> →      <var> = <expr>;

9. <simple-stmt> →      <var> = <Function-name> ( <expr-list> );

10. <simple-stmt> →     <Function-name> ( <expr-list> );

11. <simple-stmt> →     // <string>

12. <compound-stmt> →
                        while (<logic-expression>)
                             <simple-stmt>                 |
                        while (<logic-expression>)
                        {
                             <stmts>
                        }

13. <compound-stmt> →             <if_part_if_stmt>
                                  <else_part_if_stmt>
```

```
[1]14. <if_part_if_stmt>→
                    if (<logic-expression>)
                              <simple-stmt>          |
                    if (<logic-expression>)
                    {
                              <stmts>
                    }

16. <else_part_if_stmt> →
                    else
                              <simple-stmt>          |
                    else
                    {
                              <stmts>
                    }                                |

                    an empty string


17. <var>     →     < ID-name>

18. <var-List>      →         an empty string                  |
                              <var>,   < var-list>

19. <parameter-list>   →      an empty string                  |
                              <var>,   < parameter -list>

20. <expr-list>     →         an empty string                  |
                              <expr>,   < expr -list>

21. <expr-or-string-list> →   an empty string                  |
                              <expr>,   <expr-or-string-list>  |
                              <string-literal>,   <expr-or-string-list>

12. <Function-name> →  <ID-name>

22. <ID-name>   →   one letter plus any number of alphanumerical characters
```

[1] It is for simplicity that we use rules 14 and 15 to describe the syntax of *if-else* statements. The well know ambiguity of *if-else* statements resulted from rule 14 and 15 is resolved by associating an *else* to the closest *if*.

22. <expr> → **any valid numerical expression composed of variables, numerical constants, arithmetic operators + , -- , *, /, % and parentheses**

23. <logic-expression> →
any valid logic expression composed of variables, numerical constants, and relational operators: >,==,<,>=,<= and logical operators !, &&, | |

24. <numerical-literal> → any numerical constant

25. <string-literal> → any character string enclosed in a pair of double quotes