```
<email> ::= Subject: <str>. Dear <str>, <body> Best, <str>
<body> ::= <sentence> | <sentence><body>
<sentence> ::= <stmt>.
<list> ::= <stmt>, <list> | <stmt>
<stmt> ::= <var_assign> | <conditional> | <comp> |
           <while_loop> | <print> | RE: <str> | SEE: <str> with: <str>

<var_assign> ::= B<str> said <bool_expr> |
                 I<str> said <int_expr>  |
                 S<str> said <str_expr>
<conditional> ::= Suppose <bool_expr>: then <list>; otherwise, <list> Thanks

<comp> ::= <int_expr> is on the same page as <int_expr>   |
           <bool_expr> is on the same page as <bool_expr> |
           <int_expr> is less than <int_expr>             |
           <int_expr> is greater than <int_expr>
<while_loop> ::= Keep <bool_expr> in the loop, regarding: <list> Thanks
<print> ::= Highlight <str> | highlight <str>

<int_expr> ::= <int_expr> piggybacks off of <int_expr1> |
               <int_expr> drills down on <int_expr1>    |
               <int_expr1>
<int_expr1> ::= <int_expr1> joins forces with <int>     |
                <int_expr1> leverages <int>             |
                <int_expr1> remains to be seen of <int> |
                <int>

<bool_expr> ::= <comp> | <bool_expr1>
<bool_expr1> ::= <bool_expr1> or <bool_expr2>  | <bool_expr2>
<bool_expr2> ::= <bool_expr2> and <bool_expr3> | <bool_expr3>
<bool_expr3> ::= not <bool> | <bool>
<bool> ::= yep | nope | B<str>

<str> ::= <char> | <char><str> | <digit><str>
<char> ::= a | b | c | ... | x | y | z | A | B | C | ... | X | Y | Z
<int> ::= <digit> | <digit><int>
<digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
```