Expression Lexer

©1995-2001, Garth Santor

# Lexer

* Identifies token in the input stream.

# Algorithm – Initialization

1. **Set** the previous token to **Open-Parenthesis**.

# Algorithm – GetNextToken

1. **Get** the next token from in the internal analyzer.
2. **If** the **previous token** is a **Function** **and** the next token is not an **Opening-Parenthesis** **then**,
3. **Exception** “Function not followed by (”.
4. **Set** the **previous token** to the next token.
5. **Return** the next token.

# Algorithm – internal analyzer

1. **While** the current character is a space or tab **do**
2. Advance to the next character.
3. **If** the current character is an end-of-string **then**,
4. **Return** an **End-of-expression** token.
5. **If** current character is a digit **then**,
6. **Set** accumulator to numerical value of digit.
7. Advance to the next character.
8. **While** current character is a digit **do**
9. Multiply accumulator by 10 and add numerical value of current digit.
10. Advance to the next character.
11. **Return** an **Integer** token.
12. **If** current character is an equal sign **then**,
13. Advance to the next character.
14. **Return** an **Assignment** token.
15. **If** current character is a open-parenthesis **then**,
16. Advance to the next character.
17. **Return** an **Opening-Parenthesis** token.
18. **If** current character is a closed-parenthesis **then**,
19. Advance to the next character.
20. **Return** a **Closing-Parenthesis** token.
21. **If** current character is an asterisk **then**,
22. Advance to the next character.
23. **Return** a **Multiplication** token.
24. **If** current character is a virgule[[1]](#footnote-2) **then**,
25. Advance to the next character.
26. **Return** a **Division** token.
27. **If** current character is an exclamation mark **then**,
28. Advance to the next character.
29. **Return** a **Factorial** token.
30. **If** current character is a plus-sign **then**,
31. Advance to the next character.
32. **If** the previous token is a **Closing-Parenthesis**, **Operand** or **Postfix-Operator** **then**,
33. **Return** an **Addition** token.
34. **Else**
35. **Return** an **Identity** token.
36. **If** current character is a dash **then**,
37. Advance to the next character.
38. **If** the previous token is a **Closing-Parenthesis**, **Operand** or **Postfix-Operator** **then**,
39. **Return** a **Subtraction** token.
40. **Else**
41. **Return** a **Negation** token.
42. **If** current character is an alphabetic character **then**,
43. **Set** an identifier string to nil.
44. **Do**
45. Append the current character to the identifier string
46. Advance to the next character
47. **While** the current character is alphabetic **do**
48. **If** the identifier is in the identifier map **then**,
49. **Return** the associated token from the identifier map
50. **Else**
51. Add the identifier to the variable map.
52. **Return** a **Variable** token.

1. A virgule is the forward slash ‘/’ character. [↑](#footnote-ref-2)