SYMBOL TABLE for test.lsed

Total Tokenized Lexemes: 172
Elapsed Time: 0.0

LINE	LEXEME	TOKEN
1	import	KEYWORD_IMPORT
3	public	KEYWORD_PUBLIC
3	class	KEYWORD_CLASS
3	testClass	IDENTIFIER
3	:	CODEBLK_INDICATOR
4	{	OPEN_CURLY_BRACKET
5	init	KEYWORD_INIT
5	(	OPEN_PARENTHESIS
5	this	KEYWORD_THIS
5	,	SEPARATOR
5	int	KEYWORD_INT
5	num	IDENTIFIER
5	)	CLOSE_PARENTHESIS
5	:	CODEBLK_INDICATOR
6	{	OPEN_CURLY_BRACKET
7	this	KEYWORD_THIS
7	.	OBJECT_DELIMITER
7	property_2_1	IDENTIFIER
7	=	OP_ASSIGNMENT
7	5	INT_LITERAL
7	;	STMT_TERMINATOR
8	display	KEYWORD_DISPLAY
8	(	OPEN_PARENTHESIS
8	num	IDENTIFIER
8	)	CLOSE_PARENTHESIS
8	;	STMT_TERMINATOR
9	this	KEYWORD_THIS

LINE	LEXEME	TOKEN
9	.	OBJECT_DELIMITER
9	property_2_1	IDENTIFIER
9	;	STMT_TERMINATOR
10	}	CLOSE_CURLY_BRACKET
12	public	KEYWORD_PUBLIC
12	func	KEYWORD_FUNC
12	TestMethod	IDENTIFIER
12	(	OPEN_PARENTHESIS
12	)	CLOSE_PARENTHESIS
12	:	CODEBLK_INDICATOR
13	{	OPEN_CURLY_BRACKET
14	display	KEYWORD_DISPLAY
14	(	OPEN_PARENTHESIS
14	"Hi Word"	STRING_LITERAL
14	)	CLOSE_PARENTHESIS
14	;	STMT_TERMINATOR
15	}	CLOSE_CURLY_BRACKET
17	public	KEYWORD_PUBLIC
17	func	KEYWORD_FUNC
17	TestMethod3	IDENTIFIER
17	(	OPEN_PARENTHESIS
17	str	KEYWORD_STR
17	val1	IDENTIFIER
17	)	CLOSE_PARENTHESIS
17	:	CODEBLK_INDICATOR
18	{	OPEN_CURLY_BRACKET
19	display	KEYWORD_DISPLAY
19	(	OPEN_PARENTHESIS
19	val1	IDENTIFIER
19	)	CLOSE_PARENTHESIS

LINE	LEXEME	TOKEN
19	;	STMT_TERMINATOR
20	}	CLOSE_CURLY_BRACKET
21	}	CLOSE_CURLY_BRACKET
23	main	KEYWORD_MAIN
23	(	OPEN_PARENTHESIS
23	)	CLOSE_PARENTHESIS
23	{	OPEN_CURLY_BRACKET
24	display	KEYWORD_DISPLAY
24	(	OPEN_PARENTHESIS
24	"Hello Word!"	STRING_LITERAL
24	)	CLOSE_PARENTHESIS
24	;	STMT_TERMINATOR
25	integer	KEYWORD_INT
25	intvar	IDENTIFIER
25	=	OP_ASSIGNMENT
25	3	INT_LITERAL
25	;	STMT_TERMINATOR
26	float	KEYWORD_FLOAT
26	FloatVar	IDENTIFIER
26	=	OP_ASSIGNMENT
26	4.0f	FLOAT_LITERAL
26	;	STMT_TERMINATOR
27	//	COMMENT_SINGLE
29	char	KEYWORD_CHAR
29	char_Va_r	IDENTIFIER
29	=	OP_ASSIGNMENT
29	'd'	CHAR_LITERAL
29	;	STMT_TERMINATOR
30	str	KEYWORD_STR
30	stringVariable	IDENTIFIER

LINE	LEXEME	TOKEN
30	=	OP_ASSIGNMENT
30	"Testds das asddddddddddddddddddddddddddddd adasdasdasdasdasdasdas"	STRING_LITERAL
30	;	STMT_TERMINATOR
32	if	KEYWORD_IF
32	(	OPEN_PARENTHESIS
32	intvar	IDENTIFIER
32	<=	LESS_OR_EQUAL_OP
32	3	INT_LITERAL
32	and	KEYWORD_AND
32	not	KEYWORD_NOT
32	5	INT_LITERAL
32	)	CLOSE_PARENTHESIS
32	then	KEYWORD_THEN
32	:	CODEBLK_INDICATOR
33	{	OPEN_CURLY_BRACKET
34	intvar	IDENTIFIER
34	=	OP_ASSIGNMENT
34	5	INT_LITERAL
34	+	OP_ADDITION
34	3	INT_LITERAL
34	+	OP_ADDITION
34	9	INT_LITERAL
34	+	OP_ADDITION
34	7	INT_LITERAL
34	/	OP_DIVISION
34	2	INT_LITERAL
34	;	STMT_TERMINATOR
35	}	CLOSE_CURLY_BRACKET

LINE	LEXEME	TOKEN
36	elif	KEYWORD_ELIF
36	(	OPEN_PARENTHESIS
36	intvar	IDENTIFIER
36	!=	INEQUALITY_OP
36	3	INT_LITERAL
36	)	CLOSE_PARENTHESIS
36	:	CODEBLK_INDICATOR
37	{	OPEN_CURLY_BRACKET
38	intvar	IDENTIFIER
38	=	OP_ASSIGNMENT
38	10	INT_LITERAL
38	;	STMT_TERMINATOR
39	}	CLOSE_CURLY_BRACKET
40	else	KEYWORD_ELSE
40	:	CODEBLK_INDICATOR
41	{	OPEN_CURLY_BRACKET
42	pass	KEYWORD_PASS
42	;	STMT_TERMINATOR
43	}	CLOSE_CURLY_BRACKET
45	for	KEYWORD_FOR
45	(	OPEN_PARENTHESIS
45	int	KEYWORD_INT
45	i	IDENTIFIER
45	=	OP_ASSIGNMENT
45	0	INT_LITERAL
45	;	STMT_TERMINATOR
45	i	IDENTIFIER
45	<	LESS_THAN_OP
45	5	INT_LITERAL
45	;	STMT_TERMINATOR

LINE	LEXEME	TOKEN
45	i	IDENTIFIER
45	++	OP_INCREMENT
45	)	CLOSE_PARENTHESIS
45	:	CODEBLK_INDICATOR
46	{	OPEN_CURLY_BRACKET
47	display	KEYWORD_DISPLAY
47	(	OPEN_PARENTHESIS
47	testClass	IDENTIFIER
47	.	OBJECT_DELIMITER
47	property_2_1	IDENTIFIER
47	)	CLOSE_PARENTHESIS
47	;	STMT_TERMINATOR
48	}	CLOSE_CURLY_BRACKET
50	while	KEYWORD_WHILE
50	(	OPEN_PARENTHESIS
50	int	KEYWORD_INT
50	j	IDENTIFIER
50	<	LESS_THAN_OP
50	5	INT_LITERAL
50	)	CLOSE_PARENTHESIS
50	:	CODEBLK_INDICATOR
51	{	OPEN_CURLY_BRACKET
52	j	IDENTIFIER
52	++	OP_INCREMENT
52	;	STMT_TERMINATOR
53	}	CLOSE_CURLY_BRACKET
54	}	CLOSE_CURLY_BRACKET