Pascallite Language Definition - Stage 0	
1.	The features of Stage 0 are the program statement, the declaration of integer and boolean variables and constants, and a single begin-end statement which is the main body of the program.
2.	Keywords (all keywords are reserved): **program, const, var, integer, boolean, begin, end, true, false, not**
3.	All statements except one terminate with a semicolon. The exception is the `begin-end` statement, which ends with a period.

4.	Blanks are token delimiters; i.e., no token can have embedded blanks. The general rule is that if two tokens are adjacent, then one of them must be composed solely of special characters:
	`: , ; = + - .`
	Compare this to languages such as FORTRAN, which are essentially insensitive to the insertion of blanks in code (except within character literals); for example, the two FORTRAN statements below have exactly the same meaning, but in Pascallite this is not true:
	`A = B C D + 5 * * T`
	`A=BCD+5**T`

5.	Comments begin with `{` and end with `}`. They may appear anywhere a blank is allowed without changing the meaning of the program. A comment may be any text not containing an embedded `}`.
6.	Stage 0 tokens: a keyword, a non-keyword identifier, a special symbol, or an unsigned integer
7.	Each identifier may be declared at most once.
8.	Keywords are reserved.
9.	The program name is not declared.
10.	An identifier which occurs to the right of `=` in a constant declaration must be a previously declared constant.

Pascallite Language Implementation Features	
1.	A statement may freely extend over any number of lines without any special markers to denote the continuation. A *newline* is equivalent to a blank as a delimiter, so that no token may extend over more than one line.
2.	The symbol table will hold up to 256 entries. This number is arbitrary but is large enough to accommodate most "student" programs.
3.	Normally there is a maximum and a minimum valued *integer*. We won't check that with our compiler.
4.	All consts and vars are limited to a length of 15 characters. This is also an arbitrary choice.

Pascallite Language Definition - Stage 0

Pascallite Lexicon

1.	TOKEN	→	KEYWORD | SPEC_SYM | NON_KEY_ID | INTEGER
2.	KEYWORD	→	'program' | 'begin' | 'end' | 'var' | 'const' | 'integer' | 'boolean' | 'true' | 'false' | 'not'
3.	SPEC_SYM	→	'=' | ':' | ',' | ';' | '.' | '+' | '-'
4.	NON_KEY_ID	→	ALPHA ALPHANUMS
5.	ALPHANUMS	→	ALPHANUM ALPHANUMS
		→	ε
6.	ALPHANUM	→	('_' | ε) (ALPHA | NUM)
7.	INTEGER	→	NUM NUMS
8.	NUMS	→	NUM NUMS
		→	ε
9.	ALPHA	→	'a' | 'b' | 'c' | 'd' | 'e' | 'f' | 'g' | 'h' | 'i' | 'j' | 'k' | 'l' | 'm' | 'n' | 'o' | 'p' | 'q' | 'r' | 's' | 't' | 'u' | 'v' | 'w' | 'x' | 'y' | 'z'
10.	NUM	→	'0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'

Pascallite Grammar Stage 0

1.	PROG	→ PROG_STMT CONSTS VARS BEGIN_END_STMT	{'program'}
2.	PROG_STMT	→ 'program' NON_KEY_ID ';'	{'program'}
3.	CONSTS	→ 'const' CONST_STMTS	{'const'}
		→ ε	{'var','begin'}
4.	VARS	→ 'var' VAR_STMTS	{'var'}
		→ ε	{'begin'}
5.	BEGIN_END_STMT	→ 'begin' 'end' '.'	{'begin'}
6.	CONST_STMTS	→ NON_KEY_ID '='	{NON_KEY_ID}
		(	
		NON_KEY_ID	{NON_KEY_ID}
		| LIT	{'true','false',INTEGER,'+','-','not'}
		)	
		';'	
		(	
		CONST_STMTS	{NON_KEY_ID}
		| ε	{'var','begin'}
		)	
7.	VAR_STMTS	→ IDS ':' TYPE ';'	{NON_KEY_ID}
		(	
		VAR_STMTS	{NON_KEY_ID}
		| ε	{'begin'}
		)	
8.	IDS	→ NON_KEY_ID	{NON_KEY_ID}
		(	
		',' IDS	{','}
		| ε	{':'}
		)	
9.	TYPE	→ 'integer'	{'integer'}
		→ 'boolean'	{'boolean'}
10.	LIT	→ INTEGER	{INTEGER}
		→ BOOLEAN	{'false','true'}
		→ 'not' BOOLEAN	{'not'}
		→ '+' INTEGER	{'+'}
		→ '-' INTEGER	{'-'}
11.	BOOLEAN	→ 'true'	{'true'}
		→ 'false'	{'false'}