Program Input

The input to your program will be C++ code. You will first pass this code through lexical analyser. If lexical analyser will generate tokens in a file then your Parser program should read tokens from input file.

Program Output

If there are no syntax errors in the program then Parser should print the following: “The program is parsed successfully" If there is any syntax error in input program then your program should terminate on first syntax error and print the line number of the first syntax error with error message: “Syntax Error: Line # 3"

Token Type	**Lexical Specification**
keyword	One of the strings **while, if, else, return, break, continue, int , float, void**
identifier	Token Id for identifiers matches a letter followed by letters or digits or underscore: **letter** $\rightarrow$ [**A-Z** | **a-z**] **digit** $\rightarrow$ [**0-9**] **id** $\rightarrow$ **letter** (**letter** | **digit** | _)$^{+}$
num	Token num matches unsigned numbers: **digits**$\rightarrow$ **digit digit**$^{+}$ **optional-fraction** $\rightarrow$ (**. digits**) | ε **optional-exponent** $\rightarrow$ (E(+ | $-$ | ε) **digits**) | ε **num**$\rightarrow$ **digits optional-fraction optional-exponent**
addop	+ , -
mulop	*, /
relop	<, >, <=, >=, ==, !=
assignop	=
and	&&
or	||
not	!
)	)
(	(
{	{
}	}
[	[
]	]
;	;

Table 1: Table of lexical specifications for tokens