# Lexical Analyzer in C

This document contains the C program for a Lexical Analyzer, which identifies tokens such as keywords, identifiers, numbers, operators, and special symbols.

#include <stdio.h>  
#include <ctype.h>  
#include <string.h>  
  
// List of keywords  
char \*keywords[] = {"int", "float", "if", "else", "while", "return", "void", "char"};  
int numKeywords = 8;  
  
// Function to check if a word is a keyword  
int isKeyword(char \*word) {  
 for (int i = 0; i < numKeywords; i++) {  
 if (strcmp(word, keywords[i]) == 0) {  
 return 1;  
 }  
 }  
 return 0;  
}  
  
// Function to check if a character is an operator  
int isOperator(char ch) {  
 char operators[] = "+-\*/=<>!&|";  
 for (int i = 0; operators[i] != '\0'; i++) {  
 if (ch == operators[i]) {  
 return 1;  
 }  
 }  
 return 0;  
}  
  
// Lexical Analyzer function  
void lexicalAnalyzer(char \*code) {  
 int i = 0;  
 while (code[i] != '\0') {  
 // Skip spaces  
 if (isspace(code[i])) {  
 i++;  
 continue;  
 }  
  
 // Identify keywords and identifiers  
 if (isalpha(code[i])) {  
 char word[50];  
 int j = 0;  
 while (isalnum(code[i])) {  
 word[j++] = code[i++];  
 }  
 word[j] = '\0';  
   
 if (isKeyword(word)) {  
 printf("Keyword: %s\n", word);  
 } else {  
 printf("Identifier: %s\n", word);  
 }  
 }  
 // Identify numbers  
 else if (isdigit(code[i])) {  
 char num[20];  
 int j = 0;  
 while (isdigit(code[i])) {  
 num[j++] = code[i++];  
 }  
 num[j] = '\0';  
 printf("Number: %s\n", num);  
 }  
 // Identify operators  
 else if (isOperator(code[i])) {  
 printf("Operator: %c\n", code[i]);  
 i++;  
 }  
 // Identify special characters  
 else {  
 printf("Special Symbol: %c\n", code[i]);  
 i++;  
 }  
 }  
}  
  
int main() {  
 char code[1000];  
 printf("Enter the source code:\n");  
 fgets(code, sizeof(code), stdin);  
 printf("\nTokens:\n");  
 lexicalAnalyzer(code);  
 return 0;  
}