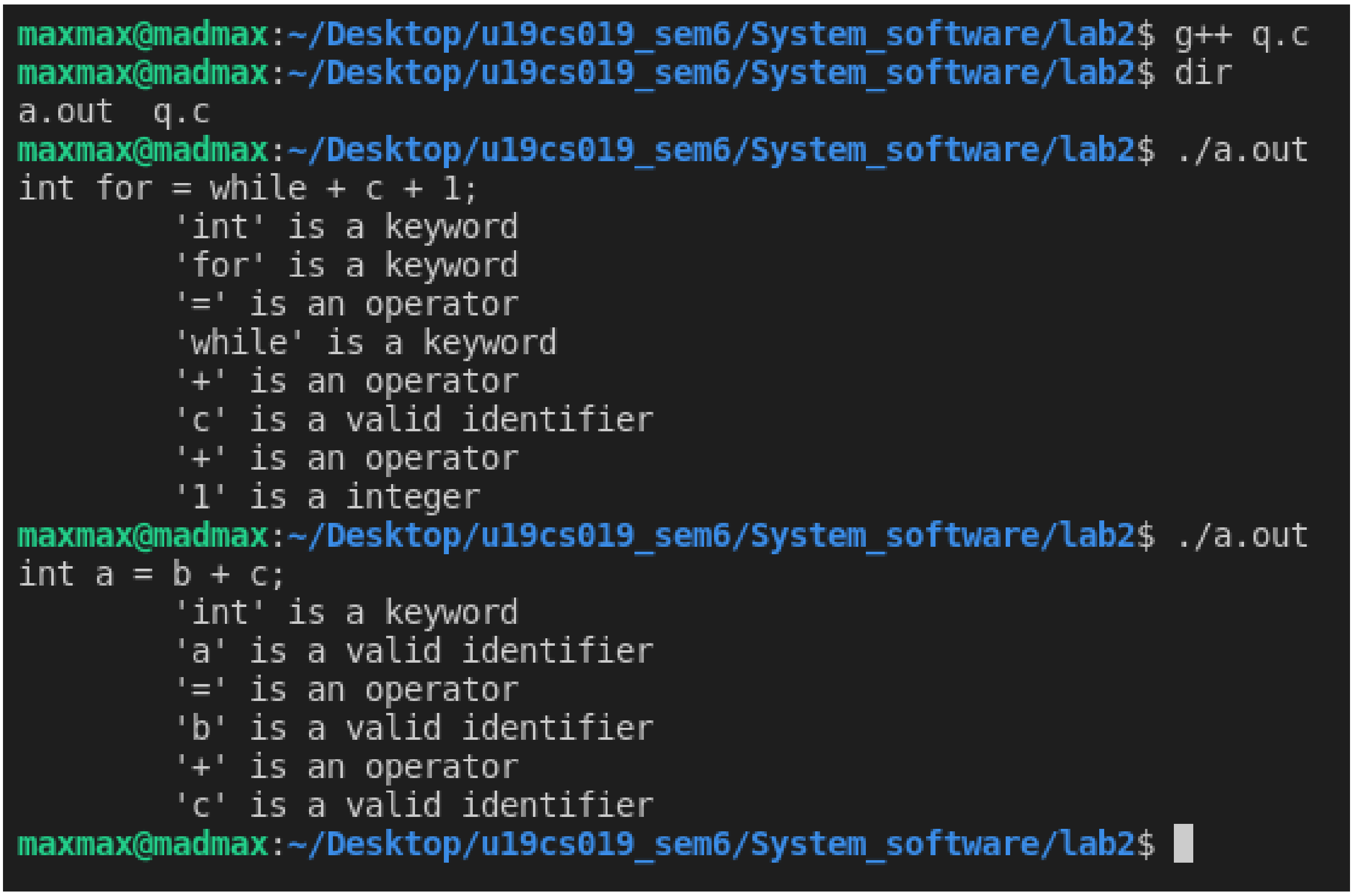
Write a program to detect tokens in c program.

```
// importing libraries
#include <stdbool.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

bool DelimiterCheck(char ch) {
if (ch == ' ' || ch == '+' || ch == '-' || ch == '*' || ch == '/' || ch == ',' || ch == ';' || ch == '>' || ch == '<' || ch == '=' || ch == '(' || ch == ')' || ch == '[' || ch == ']' || ch == '{' || ch == '}')
return (true);
return (false);
}

bool OperatorCheck(char ch) {
if (ch == '+' || ch == '-' || ch == '*' || ch == '/' || ch == '>' || ch == '<' || ch == '=')
return (true);
return (false);
}

bool IdentifierCheck(char* str) {
if (str[0] == '0' || str[0] == '1' || str[0] == '2' || str[0] == '3' || str[0] == '4' || str[0] == '5' || str[0] == '6' || str[0] == '7' || str[0] == '8' || str[0] == '9' || DelimiterCheck(str[0]) == true)
return (false);
return (true);
}

bool KeywordCheck(char* str) {
if (!strcmp(str, "if") || !strcmp(str, "for") || !strcmp(str, "else") || !strcmp(str, "while") || !strcmp(str, "do") || !strcmp(str, "break") || !strcmp(str, "continue") || !strcmp(str, "int") || !strcmp(str, "double") || !strcmp(str, "float") || !strcmp(str, "return") || !strcmp(str, "char") || !strcmp(str, "case") || !strcmp(str, "char") || !strcmp(str, "sizeof") || !strcmp(str, "long") || !strcmp(str, "short") || !strcmp(str, "typedef") || !strcmp(str, "switch") || !strcmp(str, "unsigned") || !strcmp(str, "void") || !strcmp(str, "static") || !strcmp(str, "struct") || !strcmp(str, "goto"))
return (true);
return (false);
}

bool IntegerCheck(char* str) {
int i, len = strlen(str);
```

```
if (len == 0)
return (false);
for (i = 0; i < len; i++) {
if (str[i] != '0' && str[i] != '1' && str[i] != '2' && str[i] != '3' && str[i] != '4' && str[i] != '5'
&& str[i] != '6' && str[i] != '7' && str[i] != '8' && str[i] != '9' || (str[i] == '-' && i > 0))
return (false);
}
return (true);
}

bool RealNumberCheck(char* str) {
int i, len = strlen(str);
bool hasDecimal = false;

if (len == 0)
return (false);
for (i = 0; i < len; i++) {
if (str[i] != '0' && str[i] != '1' && str[i] != '2' && str[i] != '3' && str[i] != '4' && str[i] != '5'
&& str[i] != '6' && str[i] != '7' && str[i] != '8' && str[i] != '9' && str[i] != '.' || (str[i] == '-'
&& i > 0)) return (false);
if (str[i] == '.')
hasDecimal = true;
}
return (hasDecimal);
}

char* extractSubstrign(char* str, int left, int right) {
int i;
char* subStr = (char*)malloc(sizeof(char) * (right - left + 2));

for (i = left; i <= right; i++)
subStr[i - left] = str[i];
subStr[right - left + 1] = '\0';
return (subStr);
}

void parse(char* str) {
int left = 0, right = 0;
int len = strlen(str);

while (right <= len && left <= right) {
if (DelimiterCheck(str[right]) == false)
right++;

if (DelimiterCheck(str[right]) == true && left == right) {
if (OperatorCheck(str[right]) == true)
printf("\t'%c' is an operator\n", str[right]);

right++;
left = right;
} else if (DelimiterCheck(str[right]) == true && left != right
|| (right == len && left != right)) {
char* subStr = extractSubstrign(str, left, right - 1);

if (KeywordCheck(subStr) == true)
printf("\t'%s' is a keyword\n", subStr);

else if (IntegerCheck(subStr) == true)
printf("\t'%s' is a integer\n", subStr);

else if (RealNumberCheck(subStr) == true)
printf("\t'%s' is a real number\n", subStr);

else if (IdentifierCheck(subStr) == true
&& DelimiterCheck(str[right - 1]) == false)
printf("\t'%s' is a valid identifier\n", subStr);

else if (IdentifierCheck(subStr) == false
&& DelimiterCheck(str[right - 1]) == false)
printf("\t'%s' is not a valid identifier\n", subStr);
left = right;
}
}
return;
}

int main() {
// maximum length of string is 100 here
char str[100];
scanf("%[^\n]", str);

parse(str);
return (0);
}
```



```
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab2$ g++ q.c
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab2$ dir
a.out  q.c
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab2$ ./a.out
int for = while + c + 1;
        'int' is a keyword
        'for' is a keyword
        '=' is an operator
        'while' is a keyword
        '+' is an operator
        'c' is a valid identifier
        '+' is an operator
        '1' is a integer
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab2$ ./a.out
int a = b + c;
        'int' is a keyword
        'a' is a valid identifier
        '=' is an operator
        'b' is a valid identifier
        '+' is an operator
        'c' is a valid identifier
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab2$ 
```