Write a program to implement Lexical Analyzer(Lexer).

SAMPLE CPP FILE:

```
#include <iostream>
using namespace std;
int main()
{
cout << "Hello world";
return 0;
}
```

SOURCE CODE:

```
#include <bits/stdc++.h>
using namespace std;

int isKeyword(char buffer[])
{
char keywords[49][10] = {"include", "asm", "double", "new", "switch", "auto", "else",
"operator", "template", "break", "enum", "private", "this", " case", "extern", "protected",
"throw", "catch", "float", "public", "try", "char", "for", "register", "typedef", "class", "friend",
"return", "union", "const", "goto", "short", "unsigned", "continue", "if", "signed", "virtual",
"default", "inline", "sizedof", "void", "delete", "int", "static", "volatile", "do", "long", "struct",
"while"};
int i, flag = 0;
for (i = 0; i < 48; ++i)
{
if (strcmp(keywords[i], buffer) == 0)
{
flag = 1;
break;
}
}
return flag;
}

int main()
{
char ch, buffer[15], operators[] = "+-*/%=><|&";

ifstream fin("temp.cpp");
int i, j = 0;
if (!fin.is_open())
{
cout << "error while opening the file\n";
```

```
exit(0);
}
while (!fin.eof())
{
ch = fin.get();

for (i = 0; i < 10; ++i)
{
if (ch == operators[i])
cout << ch << " is operator\n";
}

if (isalnum(ch))
{
buffer[j++] = ch;
}
else if ((ch == ' ' || ch == '\n') && (j != 0))
{
buffer[j] = '\0';
j = 0;

if (isKeyword(buffer) == 1)
cout << buffer << " is keyword\n";
else
cout << buffer << " is indentifier\n";
}
}
fin.close();
return 0;
}
```

```
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab6$ g++ q.cpp
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab6$ ./a.out
include is keyword
< is operator
> is operator
iostream is indentifier
using is indentifier
namespace is indentifier
std is indentifier
int is keyword
main is indentifier
cout is indentifier
< is operator
< is operator
Hello is indentifier
world is indentifier
return is keyword
0 is indentifier
```