Practical - 2

AIM: Write a C program to develop a lexical analyzer to recognize a few tokens in C.(Note: Read the small C program from file and recognize a tokens like Identifiers, Operators, Comments, Constants, Special Symbols etc.)

Example program:

<u>Input.c</u>

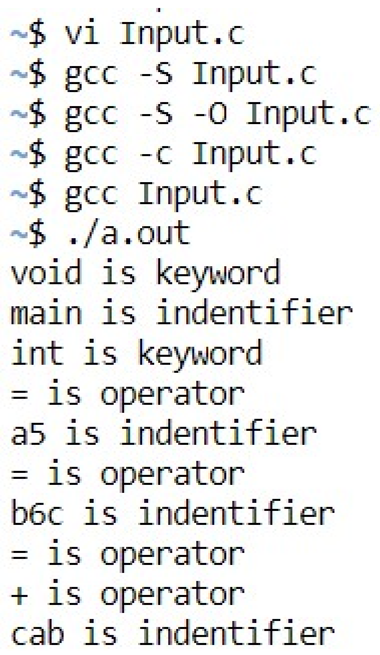
```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<ctype.h>

int isKeyword(char buffer[]){
char keywords[32][10] = {"auto","break","case","char","const","continue","default",
"do","double","else","enum","extern","float","for","goto",
"if","int","long","register","return","short","signed",
"sizeof","static","struct","switch","typedef","union",
"unsigned","void","volatile","while"};
int i, flag = 0;
for(i = 0; i < 32; ++i){
if(strcmp(keywords[i], buffer) == 0){
flag = 1;
break;
}
}
return flag;
}

int main(){
char ch, buffer[15], operators[] = "+-*/%=";
FILE *fp;
int i,j=0;
fp = fopen("program.txt","r");
if(fp == NULL){
printf("error while opening the file\n");
exit(0);
}
while((ch = fgetc(fp)) != EOF){
  for(i = 0; i < 6; ++i){
  if(ch == operators[i])
  printf("%c is operator\n", ch);
  }

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

```
  buffer[j] = '\0';
  j = 0;

  if(isKeyword(buffer) == 1)
  printf("%s is keyword\n", buffer);
  else
  printf("%s is indentifier\n", buffer);
  }

}
fclose(fp);
return 0;
}
```

<u>Program.txt</u>

```
void main()
{
int a=5, b=6,c;
c=a+b;
}
```

Output:



```
~$ vi Input.c
~$ gcc -S Input.c
~$ gcc -S -O Input.c
~$ gcc -c Input.c
~$ gcc Input.c
~$ ./a.out
void is keyword
main is indentifier
int is keyword
= is operator
a5 is indentifier
= is operator
b6c is indentifier
= is operator
+ is operator
cab is indentifier
```