Practical-3

Write a program to implement the lexical analyzer.

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<ctype.h>
#include<stdbool.h>
int isKeyword(char buffer[])
{
	char keywords[34][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","printf","main"};
	int i, flag = 0;
	for(i = 0; i < 34; ++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;
char c;
bool flagk;
fp = fopen("program.txt","r");
if(fp == NULL)
{
        printf("error while opening the file\n");
        exit(0);
}
flagk=0;
while((ch = fgetc(fp)) != EOF)
{
        if(ch=='"')
        {
                flagk=!flagk;
                continue;
        }
        if(flagk==1)
        continue;
        for(i = 0; i < 6; ++i)
        {
                if(ch == operators[i])
                printf("%c is operator\n", ch);
        }
        if(isalnum(ch) && !isdigit(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 identifier\n", buffer);
                }
        }
        return 0;
}
```

OUTPUT:

```
void is keyword
main is keyword
int is keyword
ab is identifier
= is operator
a is identifier
= is operator
b is identifier
int is keyword
= is operator
+ is operator
cab is identifier
printf is identifier
```