**GAURAV AMARNANI**

## SPCC LAB 02

### LEXICAL ANALYSER

**Code:**

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

int main()
{
    char keys[32][10]={"auto","double","int","struct","break","else",
                       "long","switch","register","typedef",
                       "return","union","short","unsigned","signed",
                       "void","sizeof","volatile","static","while",
                       "case","enum","char","extern","const","float",
                       "continue","for","default","goto","do","if"};
    char arith[7][3]={"+","-","*","/","%","++","--"};
    char logical[3][2]={"&&","!","||"};
    char relational[6][2]={"<",">","==","<=",">=","!="};
    char assignop[6][2]={"=","+=","-=","*=","/=","%="};
    char bitwise[6][2]={"&","|","^","~","<<",">>"};
    char punct[8][3]={";",",","{","}","[","]","(",")"};

    FILE *f1;
    f1=fopen("program.txt","r");
    char w[100];
    int i, found=0;
    int op=0, ky=0, id=0, sp=0;
    if(f1==NULL)
    {
        printf("Enter a input in .txt file\n");
    }
    else
    {
        while(fscanf(f1,"%s",w)==1)
        {
            for(i=0;i<32;i++)
            {
                if(strcmp(w,keys[i])==0)
                {
                    printf("%s is a keyword\n",w);
                    found=1;
                    ky++;
                }
            }
            for(i=0;i<7;i++)
            {
                if(strcmp(w,arith[i])==0)
                {
                    printf("%s is a Arithmetic operator\n",w);
                    found=1;
                    op++;
                }
            }
```

```
            for(i=0;i<3;i++)
            {
                if(strcmp(w,logical[i])==0)
                {
                    printf("%s is a Logical operator\n",w);
                    found=1;
                    op++;
                }
            }
            for(i=0;i<6;i++)
            {
                if(strcmp(w,relational[i])==0)
                {
                    printf("%s is a Relational operator\n",w);
                    found=1;
                    op++;
                }
                else if(strcmp(w,assignop[i])==0)
                {
                    printf("%s is a Assignment operator\n",w);
                    found=1;
                    op++;
                }
                else if(strcmp(w,bitwise[i])==0)
                {
                    printf("%s is a Bitwise operator\n",w);
                    found=1;
                    op++;
                }
            }
            for(i=0;i<8;i++)
            {
                if(strcmp(w,punct[i])==0)
                {
                    printf("%s is a Special character\n",w);
                    found=1;
                    sp++;
                }
            }
            if(found  == 0)
            {
                if (isdigit(w[0]))
                {
                    if (isdigit(w[0]) && isalpha(w[1]))
                    {
                        printf("%s is an Invalid Identifier\n", w);
                    }
                    else
                    {
                        printf("%s is constant\n", w);
                    }
                }
                else if (isalnum(w[0]))
                {
                    if (isalnum(w[strlen(w)-1]) && isalpha(w[0]))
                    {
                        printf("%s is Identifier\n", w);
                    }
                    else if ((w[0] >= '0') && (w[0] <= '9'))
                    {
                        printf("%s is an Invalid Identifier\n", w);
                    }
```

```
                }
            }
            found=0;
        }

    }
    return 0;
}
```

program.txt

```
program.txt
1    void main()
2    {
3        int j = 3 , b = 4 , c ;
4        c = a + b ;
5        printf( "%d" , c ) ;
6    }
```

**Output:**