EX1 IMPLEMENTATION OF LEXICAL ANALYZER

Aim:

To write a c program to implement lexical analyzer and display the symbol table.

Code:

```
#include<stdio.h>
#include<fcntl.h>
#include<string.h>
#include<unistd.h>
#include<stdlib.h>

int isSeparator(char ch)
{
   if (ch == ',' || ch == '{' || ch == '}' ||
       ch == '(' || ch == ')' || ch==';')
       return 1;
   return 0;
}

int isOperator(char ch)
{
   if (ch == '+' || ch == '-' || ch == '*' ||
       ch == '/' || ch == '>' || ch == '<' ||
       ch == '=')
       return 1;
   return 0;
}


int isKey(char* test,int start,int end)
{
  char str[50]="\0";
  int i,len=end-start+1;
  for(i=0;i<len;i++)
                str[i]=test[i];
  //printf("test: %s",str);
  if (!strcmp(str, "if") || !strcmp(str, "else") || !strcmp(str, "while")
         || !strcmp(str, "do") || !strcmp(str, "break") ||  !strcmp(str, "for")
         || !strcmp(str, "int") || !strcmp(str, "double") || !strcmp(str, "float")
     || !strcmp(str, "return") || !strcmp(str, "char") || !strcmp(str, "case")
         || !strcmp(str, "short") || !strcmp(str, "switch") || !strcmp(str, "sizeof")
     || !strcmp(str, "long")  || !strcmp(str, "void") || !strcmp(str, "typedef")
     || !strcmp(str, "unsigned") || !strcmp(str, "goto") || !strcmp(str, "static")
     || !strcmp(str, "struct") || !strcmp(str, "continue") || !strcmp(str, "signed")
         || !strcmp(str, "unsigned"))
       return 1;
```

```
    return 0;
}

int isConstant(char ch)
{
  if ((ch>='0' && ch<='9')ll ch=='.')
    return 1;
  return 0;
}

int main()
{
        int i=0,j=0,k=0,l,flag,ptr,start,end;
//"/Users/cse03/Desktop/cd_lab/ex1/
        int fd = open("/Users/cse03/Desktop/cd_lab/ex1/sample.txt",O_RDONLY);
        if(fd==-1)
                printf("\nError opening file");
        char ch[2]="\0";
        char str[100][100];
   strcpy(str[0],"\0");
   char
test[50]="\0",type_str[50]="\0",check_str[50]="\0",var_str[50]="\0",value_str[50];

        int n=read(fd,ch,1);

        while(n!=0)
   {
     if((strcmp(ch,"\n")==0))
          {
                 strcat(str[j],ch);
        j++;
        strcpy(str[j],"\0");
     }
     else
     {
       strcat(str[j],ch);
     }
          n=read(fd,ch,1);
        }
        printf("\nThe code is:\n");  // j is the length
        for(l=0;l<j;l++)
                printf("Line%d: %s",l,str[l]);
        printf("\n");
   for(i=0;i<j;i++)  //for every line i
   {
         ptr=k;
         printf("Line: %d ",i);
                while(ptr!=strlen(str[i]))
                {  flag=0;
                        if( (str[i][ptr]>='a' && str[i][ptr]<='z') ll (str[i][ptr]>='A' && str[i]
```

```
[ptr]<='Z')  )
                    {
                      while( (str[i][ptr]>='a' && str[i][ptr]<='z') || (str[i][ptr]>='A' && str[i]
[ptr]<='Z') )
                      {
                            ptr++;
                      }
                      if(isKey(str[i],k,ptr-1)==1)
                      {
                            printf("Keyword  ");
                      }
                      else
                      {
                                  while( (str[i][ptr]>='a' && str[i][ptr]<='z') || (str[i][ptr]>='A' &&
str[i][ptr]<='Z') || (str[i][ptr]>='0'  && str[i][ptr]<='9') )
                                  {
                                        ptr++;
                                  }
                                  if(str[i][ptr]=='(')
                                  {       ptr=strlen(str[i]);
                                     printf("Function Call  ");
                                  }
                                  else
                                         printf("Variable  ");
                      } //end else
                    } //end if
                    else
                    {
                      if(isSeparator(str[i][ptr])==1)
                      {
                             ptr++;
                             flag=1;
                             printf("Separator  ");
                      }
                      else if(isOperator(str[i][ptr])==1)
          {
                                  flag=1;
                                  if(str[i][ptr]=='=')
                                        printf("Assignment Operator  ");
                                  else if(str[i][ptr]=='+' || str[i][ptr]=='-' || str[i][ptr]=='*' || str[i]
[ptr]=='/')
                                        printf("Arithmetic Operator  ");
                                  else if(str[i][ptr]=='>' || str[i][ptr]=='<')
                                        printf("Relational Operator  ");
                                  ptr++;
                      }
                      else if(isConstant(str[i][ptr])==1)
                      {
                     while((isConstant(str[i][ptr])==1))
                                        ptr++;
```

```
            printf("Constant  ");
                        flag=1;
                    }
                if(flag==0)
                            ptr++;
                  }  //end else
                            //check if test string is a keyword
                            //if it is not a keyword  concat prev k and continue
concatenating while including nos and alpha and underscore
                            //now if condition fails identify it as id
                            //when condition fails reinitialise test string and read again
and check for operators <,<=,+....
                }  //end while strlen
                    printf("\n");
   } //end for

printf("\n\tSYMBOL TABLE\n");
printf("\nTYPE     NAME     VALUE         NO.OF BYTES \n");
       for(i=0;i<j;i++)
       {              ptr=k=0;
                     if( (str[i][ptr]>='a' && str[i][ptr]<='z') || (str[i][ptr]>='A' && str[i]
[ptr]<='Z')  )
                     {
                       while( (str[i][ptr]>='a' && str[i][ptr]<='z') || (str[i][ptr]>='A' && str[i]
[ptr]<='Z') )
                       {
                            ptr++;
                       }
                       if(isKey(str[i],k,ptr-1)==1)
                       {

                              int x,len=ptr-k;
                              strcpy(type_str,"\0");
                              for(x=0;x<len;x++)
                                          type_str[x]=str[i][k+x];    type_str[x]='\0';
                              if(!strcmp(type_str,"int"))
                              {
                                          ptr++;
                                          while(str[i][ptr]!=';')
                                     {
                                                 while(str[i][ptr]==' ') ptr++;
                                                 if(str[i][ptr]==',')  ptr++;
                                                 start=ptr;
                                                while((str[i][ptr]>='a' && str[i][ptr]<='z') ||(str[i]
[ptr]>='A' && str[i][ptr]<='Z') || (str[i][ptr]>='0' && str[i][ptr]<='9'))
                                                            ptr++;

                                                 end=ptr-1;
                                                 int x,len=end-start+1;
                                                 strcpy(var_str,"\0");
```

```
                                        for(x=0;x<len;x++)
                                                var_str[x]=str[i][start+x];
        var_str[x]='\0';//printf("\ntyp: %s",var_str); //exit(0);
                                        while (str[i][ptr]==' ') ptr++;
                                        if(str[i][ptr]=='=')
                                        {                       ptr++;//printf("\ntyp:
%s",type_str); exit(0);
                                                while (str[i][ptr]==' ') ptr++;
                                                start=ptr;
                                                while(str[i][ptr]>='0' && str[i][ptr]<='9') ptr+
+;
                                                        end=ptr-1;//printf("qwe: %c\n",str[i]
[end]);
                                                len=end-start+1;
                                              strcpy(value_str,"\0");     //printf("\nlen:
%d",len);
                                                for(x=0;x<len;x++)
                                                  value_str[x]=str[i][start+x];
                                                  value_str[x]='\0';      //printf("\ntyp:
%s",value_str);
                                                //value=atoi(value_str);
                                                printf("\n%s      %s       %s
2\n",type_str,var_str,value_str);
                                        }
                                        else if(str[i][ptr]==',')
                                        {
                                                printf("\n%s        %s       Not initialised
2\n",type_str,var_str);
                                                ptr++;
                                        }
                                        else if(str[i][ptr]==';')
                                        {
                                                printf("\n%s       %s       Not initialised
2\n",type_str,var_str);
                                        }
                                } //end while(str[i][ptr]!=';')

                        }  //end if(!strcmp(type_str,"int"))
                        else if(!strcmp(type_str,"float"))
                        {
                                ptr++;
                                while(str[i][ptr]!=';')
                            {
                                        while(str[i][ptr]==' ') ptr++;
                                        if(str[i][ptr]==',')  ptr++;
                                        start=ptr;
                                  while((str[i][ptr]>='a' && str[i][ptr]<='z') ||(str[i]
[ptr]>='A' && str[i][ptr]<='Z') || (str[i][ptr]>='0' && str[i][ptr]<='9'))
                                                ptr++;
```

```
                                        end=ptr-1;
                                        int x,len=end-start+1;
                                        strcpy(var_str,"\0");
                                        for(x=0;x<len;x++)
                                                var_str[x]=str[i][start+x];
var_str[x]='\0';   //printf("\ntyp: %s",var_str); //exit(0);
                                        while (str[i][ptr]==' ') ptr++;
                                        if(str[i][ptr]=='=')
                                        {                       ptr++;//printf("\ntyp:
%s",type_str); exit(0);
                                                while (str[i][ptr]==' ') ptr++;
                                                start=ptr;
                                                while((str[i][ptr]>='0' && str[i][ptr]<='9') ||
str[i][ptr]=='.' ) ptr++;
                                                        end=ptr-1;//printf("qwe: %c\n",str[i]
[end]);
                                                len=end-start+1;
                                              strcpy(value_str,"\0");     //printf("\nlen:
%d",len);
                                                for(x=0;x<len;x++)
                                                  value_str[x]=str[i][start+x];
                                                  value_str[x]='\0';      //printf("\ntyp:
%s",value_str);
                                                //value=atoi(value_str);
                                                printf("\n%s     %s       %s
2\n",type_str,var_str,value_str);
                                        }
                                        else if(str[i][ptr]==',')
                                        {
                                                printf("\n%s     %s       Not initialised
2\n",type_str,var_str);
                                                ptr++;
                                        }
                                        else if(str[i][ptr]==';')
                                        {
                                                printf("\n%s     %s       Not initialised
2\n",type_str,var_str);
                                        }
                                } //end while(str[i][ptr]!=';')

                        }       //end if float
                else if(!strcmp(type_str,"char"))
                {
                        ptr++;                  //printf("ji: %s",type_str);
                        while(str[i][ptr]!=';')
                  {
                                while(str[i][ptr]==' ') ptr++;
                                if(str[i][ptr]==',')  ptr++;
                                start=ptr;
                              while((str[i][ptr]>='a' && str[i][ptr]<='z') ||(str[i]
```

```
[ptr]>='A' && str[i][ptr]<='Z') || (str[i][ptr]>='0' && str[i][ptr]<='9'))
                                                ptr++;

                                        end=ptr-1;
                                        int x,len=end-start+1;
                                        strcpy(var_str,"\0");
                                        for(x=0;x<len;x++)
                                                var_str[x]=str[i][start+x];
var_str[x]='\0';   //printf("\ntyp: %s",var_str); exit(0);
                                        while (str[i][ptr]==' ') ptr++;
                                        if(str[i][ptr]=='=')
                                        {                       ptr++;//printf("\ntyp:
%s",type_str); exit(0);
                                                while (str[i][ptr]==' ' || str[i][ptr=='\'']) ptr++;
                                                start=ptr;
                                                while((str[i][ptr]>='a' && str[i][ptr]<='z') ||
(str[i][ptr]>='A' && str[i][ptr]<='Z') || (str[i][ptr]>='0' && str[i][ptr]<='9')) ptr++;   //change
this line
                                                        end=ptr-1;printf("qwe: %c\n",str[i]
[end]);  exit(0);
                                                len=end-start+1;
                                              strcpy(value_str,"\0");     //printf("\nlen:
%d",len);
                                                for(x=0;x<len;x++)
                                                  value_str[x]=str[i][start+x];
                                                  value_str[x]='\0';      //printf("\ntyp:
%s",value_str);
                                                //value=atoi(value_str);
                                                        printf("\n%s     %s     %s
2\n",type_str,var_str,value_str);
                                        }
                                        else if(str[i][ptr]==',')
                                        {
                                                printf("\n%s     %s     Not initialised
2\n",type_str,var_str);
                                                ptr++;
                                        }
                                        else if(str[i][ptr]==';')
                                        {
                                                printf("\nT%s     %s     Not initialised
2\n",type_str,var_str);
                                        }
                                } //end while(str[i][ptr]!=';')

                        }//end if char
                }  //end if(isKey(str[i],k,ptr-1)==1)
              }//end if
        } //end for

return 0;
```

}

Output:

```
ex1 — -bash — 79×44
[gml03:cd_lab cse03$ cd ex1
[gml03:ex1 cse03$ gcc lexan.c
[gml03:ex1 cse03$ ./a.out

The code is:
Line0: main()
Line1: {
Line2: int var=10,b=20,c;
Line3: float x=10.0,y=20,z;
Line4: if(a>b)
Line5:  printf("a is greater");
Line6: else
Line7:  printf("b is greater");
Line8: }

Line: 0 Function Call
Line: 1 Separator
Line: 2 Keyword  Variable  Assignment Operator  Constant  Separator  Variable
Assignment Operator  Constant  Separator  Variable  Separator
Line: 3 Keyword  Variable  Assignment Operator  Constant  Separator  Variable
Assignment Operator  Constant  Separator  Variable  Separator
Line: 4 Keyword  Separator  Variable  Relational Operator  Variable  Separator

Line: 5 Function Call
Line: 6 Keyword
Line: 7 Function Call
Line: 8 Separator

        SYMBOL TABLE

TYPE     NAME     VALUE          NO.OF BYTES

int      var       10                     2

int      b        20                     2

int       c       Not initialised        2

float     x       10.0                   2

float     y       20                  2

float     z       Not initialised        2
gml03:ex1 cse03$
```