Ahsanullah University of Science and Technology

Department of Computer Science and Engineering

CSE 4130

Formal languages and Compilers lab

Assignment No: 04

Submitted By:

Name: Anika Tanzim

ID: 16.02.04.072

Group: B1

Date of Submission:**6 September 2020**

Code:

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

FILE *p1, *p2;
char d,e,f,g;
char c;
char lna[10];
int ln = 1, x;

void LineNumbers(){

   if(!p1)
      printf("\nFile can't be opened!");
   else
   {
         while((c = fgetc(p1)) != EOF)
         {

         itoa(ln, lna, 10);
         for(x=0; x < strlen(lna); x++)
            fputc(lna[x], p2);

         fputc(' ', p2);
         while(c!='\n')
         {
            fputc(c, p2);
            c = fgetc(p1);
         }
         fputc('\n', p2);
         ln++;
      }
   }

}

void Lexemes()
{
```

```
    char c;
    fputc(' ', p2);
    while((c=fgetc(p1))!=EOF)
    {


        if(!isalnum(c) && c!='.' && c!=' ' && c!='_' ){
            fputc(' ', p2);
        }

        fputc(c, p2);
        if(c=='=' || c=='!' || c=='>' || c=='<' )
        {
            char d;
            if((d=fgetc(p1))=='=')
            {
                fputc(d, p2);
                fputc(' ', p2);
            }
            else
            {
                fputc(' ', p2);
                fputc(d, p2);
                if(!isalnum(d) && d!='.' && d!=' ' && d!='_')
                    fputc(' ', p2);
            }
        }
        else if(!isalnum(c) && c!='.' && c!=' ' && c!='_')
            fputc(' ', p2);

    }

}

void RemoveComments(){

    while((c = fgetc(p1)) != EOF)
    {
```

```
if( c == ' ')
{

   //one space
   fputc(c, p2);
   while((c = fgetc(p1))==' ')
   {
     continue;
   }
}


if (c=='/') // removing comment
{
   d=fgetc(p1);
   if(d=='/')
   {
      //single line comment
      while((e=fgetc(p1))!=EOF)
      {
         if(e=='\n'){
           break;
         }
      }

   }
   else if( d == ' ' && (d=fgetc(p1)) ==' ' && (d=fgetc(p1)) =='*')
   {
      //multiple line comment
      while((f=fgetc(p1))!=EOF)
      {
         if(f=='*')
         {
            g=fgetc(p1);
            g=fgetc(p1);
            g=fgetc(p1);
            if(g=='/'){
              break;
```

```
                    }
                }
            }
        }
        else
        {
            fputc(c,p2);
            fputc(d,p2);
        }
    }
    else{
        fputc(c,p2);
    }
  }
}


int Identifier(char *str)
{
   int i, s=0, l;
   l=strlen(str);
   if((isalpha(str[0])) || (str[0]=='_'))
      s=1;
   if(s==1)
   {
      for(i=1; i<l; i++)
      {
         if(!isalnum(str[i]) && str[i]!='_'&& !isalpha(str[i]))
         {
            s=0;
            break;
         }
      }
   }
   return s;
}

int Keyword(char *str) {
   int s=0;
```

```
    if( (!strcmp(str, "while")) || (!strcmp(str, "static")) || (!strcmp(str, "if")) ||
    (!strcmp(str, "volatile")) || (!strcmp(str, "do")) || (!strcmp(str, "goto")) ||
    (!strcmp(str, "sizeof")) || (!strcmp(str, "else")) || (!strcmp(str, "default")) ||
    (!strcmp(str, "void")) || (!strcmp(str, "for")) || (!strcmp(str, "signed")) ||
    (!strcmp(str, "continue")) || (!strcmp(str, "unsigned")) || (!strcmp(str, "short")) ||
    (!strcmp(str, "char")) || (!strcmp(str, "float")) || (!strcmp(str, "double")) ||
    (!strcmp(str, "int")) || (!strcmp(str, "char")) || (!strcmp(str, "const")) ||
    (!strcmp(str, "union")) || (!strcmp(str, "return")) || (!strcmp(str, "extern")) ||
    (!strcmp(str, "enum")) || (!strcmp(str, "register")) || (!strcmp(str, "typedef")) ||
    (!strcmp(str, "switch")) || (!strcmp(str, "long")) || (!strcmp(str, "break")) ||
    (!strcmp(str, "auto")) || (!strcmp(str, "struct")) )
    {
       s=1;
    }
    return s;
}


int main(void)
{


    int arr1[100];
    int arr2[100];

    p1 = fopen("input.c", "r");
    p2 = fopen("output1.txt", "w");

    LineNumbers();

    fclose(p1);
    fclose(p2);


    //print output file 1
    p2 = fopen("output1.txt", "r");
    while((c = fgetc(p2))!= EOF)
       printf("%c", c);
    fclose(p2);
```

```
//intermiditae
 p1 = fopen("output1.txt", "r");
 p2 = fopen("output2.txt","w");

 Lexemes();
 fclose(p1);
 fclose(p2);


 p1 = fopen("output2.txt", "r");
 p2 = fopen("output3.txt","w");


 RemoveComments();

 fclose(p1);
 fclose(p2);

// id print
 p1 = fopen("output3.txt", "r");
 p2 = fopen("output4.txt","w");
 char arr[100];
 int i;

while((c=fgetc(p1))!=EOF)
{

   if((isalpha(c)) || (c=='_'))
   {
      i=0;
      arr[i++]=c;
      while((c=fgetc(p1))!=' ')
      {
         arr[i++]=c;
      }
      arr[i]='\0';
      if(Identifier(arr) && (!Keyword(arr))){
```

```
            fprintf(p2, "id %s",arr);
        }
        else{
            fprintf(p2, "%s",arr);
        }

    }
    fputc(c,p2);
}

 printf("\n\n");
 fclose(p1);
 fclose(p2);
```

//print intermediate output

```
 p2 = fopen("output4.txt","r");
 while((c=fgetc(p2))!=EOF){
    printf("%c",c);
 }
 fclose(p2);

printf("\n\n");
```

//mismatched curly braces,duplicate token and mismatched else if

```
ln =1;
int m=0,n=0;
int cnt=0;
int dup=0;
p2 = fopen("output2.txt","r");


while((c=fgetc(p2))!=EOF){


    if(c=='f' && (c=fgetc(p2))=='o' && (c=fgetc(p2))=='r'){
        while((c=fgetc(p2))!=')'){
            if(c==';'){
```

```
            dup++;
        }
    }
    if(dup>2){
        printf("duplicate token at line %d",ln);
    }
}


if(c==';'){
    c=fgetc(p2);
    c=fgetc(p2);
    if((c=fgetc(p2)) == ';'){
        printf("duplicate token at line %d\n",ln);
    }
}

if(c=='\n'){
    ln++;
}
if(c=='{'){
    arr1[m]=ln;
    m++;

}
if(c=='}'){
    arr2[n]=ln;
    n++;

}
if(c=='i'){
    if((c=fgetc(p2))=='f'){
        cnt++;
    }
}
if(c=='e' && (c=fgetc(p2))=='l' && (c=fgetc(p2))=='s' && (c=fgetc(p2))=='e'){
    cnt--;
    if(cnt<0){
        printf("mismatched else at line %d\n",ln);
```

```
            }
        }


    }


    int x1=0,x2=0;

    if(m!=n){
        if(m<=n){
            for(x1=0;x1<m;x1++){
                if(arr1[x1]>arr2[x1]){
                    printf("misplaced '}' at line %d\n",arr2[x1]);
                }
            }
        }else{
            for(x1=0;x1<n;x1++){
                if(arr1[x1]>arr2[x1]){
                    printf("misplaced '}' at line %d\n",arr2[x1]);
                }
            }
            printf("unbalanced paranthesis at line %d\n",ln-1);

        }
    }

    fclose(p2);


    return 0;
}
```

Output:

```
1 /* A program fragment*/
2
3 float x1=3.125;;;
4 /*Definition of function f1*/
5 double      f1(float a,int int x)
6 {if(x<x1)
7 double z;;
8 else z=     0.01+x*5.5}}
9 else return z;
10 }
11 /*Beginning of 'main'   */
12 int main(void)
13 {{{{
14 int n1; double z;
15 n1=25; z=f1(n1);}
16
```

```
 1
 2
 3 float id x1 = 3.125 ; ; ;
 4
 5 double id f1 ( float id a , int int id x )
 6 { if ( id x < id x1 )
 7 double id z ; ;
 8 else id z = 0.01 + id x * 5.5 } }
 9 else return id z ;
 10 }
 11
 12 int id main ( void )
 13 { { { {
 14 int id n1 ; double id z ;
 15 id n1 = 25 ; id z = id f1 ( id n1 ) ; }
 16
```

```
duplicate token at line 3
duplicate token at line 7
```

mismatched else at line 9
misplaced '}' at line 8
misplaced '}' at line 10
unbalanced parenthesis at line 16