Ahsanullah University of Science and Technology (AUST)
Department of Computer Science and Engineering

**Course No : ** CSE4130

Course Title: Formal Languages and Compilers Lab

Session : Spring 2020

Assignment No : 04

Submitted By :

**Lab Group : ** A1

Name : Umme Habiba

ID :170104004

Question: Suppose, a given C source program has been scanned, filtered, lexically analyzed and tokenized as that were done in earlier sessions. In addition, line numbers have been assigned to the source code lines for generating proper error messages. As the first step to Syntax Analysis, we now perform detection of simple syntax errors like duplication of tokens except parentheses or braces, unbalanced braces or parentheses problem, unmatched ‘else’ problem, etc. Duplicate identifier declarations must also be detected with the help of the Symbol Table.

Answer:

```
#include<stdio.h>

#include<string.h>

#include<stdlib.h>

FILE *f1,*f2;

char c,lex[10];

int ln,line,countIf=0,opening_brace=0,closing_brace=0,countIfElse=0,i,j;

char str[10];

char temp[10];

int k;

void comment_rmv(){

  f1=fopen("input.txt","r");

  f2=fopen("output1.txt","w");

  if(!f1)

    printf("File can't be opened");

  else{

    while((c=fgetc(f1))!=EOF) {

      if(c=='/') {

        c=fgetc(f1);

        if(c=='/') {
            while(c!='\n') {
                c=fgetc(f1);
            }
        }
        else if(c=='*') {
            while((c=fgetc(f1))!=EOF) {
                if(c=='*') {
                    c=fgetc(f1);
                    if(c=='/')
                    {
                        break;
                    }
                }
            }
        }
        else{
            fputc(c,f2);
        }
    }
    else if(c=='\t') {
    }
    else{
        fputc(c,f2);
    }
}
```

```
    }
    fclose(f1);
    fclose(f2);
}
void space_rmv(){
    f1=fopen("output1.txt","r");
    f2=fopen("output2.txt","w");
    if(!f1)
        printf("File can't be opened");
    else{
        while((c=fgetc(f1))!=EOF) {
            if(c==' ') {
                while(c==' ') {
                    c=fgetc(f1);
                }
                fputc(' ',f2);
                fputc(c,f2);
            }
            else{
                fputc(c,f2);
            }
        }
    }
    fclose(f1);
    fclose(f2);
```

```
}

void closing_barce_rmv(){
   ln=1;
   f1=fopen("output2.txt","r");
   f2=fopen("output3.txt","w");
   fprintf(f2,"%d :",ln);
   if(!f1)
      printf("File can't be opened");
   else{
      while((c=fgetc(f1))!=EOF) {
         if(c=='\n') {
            ln++;
            fprintf(f2,"\n%d :",ln);
         }
         else{
            if(c=='{') {
               opening_brace++;
            }
            if(c=='}') {
               closing_brace++;
               if(( opening_brace< closing_brace)||(opening_brace> closing_brace)) {
                  printf("Misplaced '}' at line %d \n",ln);
               }
            }
            fprintf(f2,"%c",c);


          }
        }
      }
      fclose(f1);
      fclose(f2);
}
void lexim_sep(){
      f1 = fopen("output3.txt","r");
      f2 = fopen("output4.txt","w");
      fputc(' ',f2);
      if(!f1)
        printf("\nFile can't be opened!");
      else{
        while((c=fgetc(f1)) != EOF) {
          if(!isalnum(c) && c!='_'&& c!='.'&& c!=' ') {
            fputc(' ',f2);
          }
          fputc(c,f2);
          if(c=='='||c=='!'||c=='<'||c=='>') {
            c=fgetc(f1);
            if(c=='=') {
              fputc(c,f2);
              if(!isalnum(c) && c!='_'&& c!='.'&& c!=' ')
```

```
                  fputc(' ',f2);
               }
               else{
                  fputc(' ',f2);
                  fputc(c,f2);
                  if(!isalnum(c) && c!='_'&& c!='.'&& c!=' ')
                     fputc(' ',f2);
               }
            }
            else if(!isalnum(c) && c!='_'&& c!='.'&& c!=' ')
               fputc(' ',f2);
         }
      }
   fclose(f1);
   fclose(f2);
}
void else_rmv(){
   ln=1;
   f1=fopen("output4.txt","r");
   if(!f1)
      printf("File can't be opened");
   else{
      while((c=fgetc(f1))!=EOF) {
         if(c=='\n') {
            ++ln;
            }
            else{
                i =0;
                while(c!=' ' && c!=EOF) {
                    lex[i]=c;
                    ++i;
                    c=fgetc(f1);
                }
                lex[i]='\0';
                if(strcmp(lex,"if")==0) {
                    ++countIfElse;
                }
                else if(strcmp(lex,"else")==0) {
                    --countIfElse;
                    if(countIfElse<0)
                        printf("\nUnmatched else detected at line %d \n",ln);
                }
            }
        }
    }
    fclose(f1);
}
int Check_keyword(char lex[10]) {
    int i,s;
```

```
if(!strcmp(lex,"int")||!strcmp(lex,"char")||!strcmp(lex,"float")||!strcmp(lex,"double")||!strcmp(l
ex,"void")||!strcmp(lex,"if")||!strcmp(lex,"else")||!strcmp(lex,"return")){

        s=1;

        return s;

    }

    else{

        s=0;

        return s;

    }

}

int Check_separator(char lex[10]) {

    if(!strcmp(lex,"'")||!strcmp(lex,",")||!strcmp(lex,";"

                                                  )) {

        return 1;

    }

    else

        return 0;

}

int checkId(char lex[10]) {

    int i, s=0, l;

    l=strlen(lex);

    if((isalpha(lex[0])) || (lex[0]=='_'))

        s=1;

    if(s==1) {

        for(i=1; i<l; i++){
            if(!isalnum(lex[i]) && lex[i]!='_'&&
                !isalpha(lex[i])) {
              s=0;
              break;
            }
          }
        }
        return s;
      }
      int isItLineNo(char st[10]) {
        int l,s;
        l=strlen(st);
        for( k=0; k<l; k++){
          if(isdigit(st[k]))
            s=1;
          else
          {
            s=0;
            break;
          }
        }
        if(s==1) {
          ln = atoi(st);
          fscanf(f1, "%s", &str);
          if(!strcmp(str,":")) {
            return 1;
        }
    }
    return 0;
}
void token_rmv(){
    ln=1;
    char lex1[10],lex2[10];
    f1=fopen("output4.txt","r");
    if(!f1)
        printf("File can't be opened");
    else{
        strcpy(temp, "");

        while(fscanf(f1, "%s", &str)!=EOF) {
            if(isItLineNo(str))
                line=ln;
            if (Check_keyword(str) ) {
                  if(strcmp(str,temp)==0) {
                  printf("Duplicate token %s at line %d\n",str,line);
                   strcpy(temp, str);
                }
                else
                   strcpy(temp, str);
            }
            else if (Check_separator(str) )
            {
                if(strcmp(str,temp)==0) {
                    printf("Duplicate token %s at line %d\n",str,line);
                    strcpy(temp, str);
                }
                else
                    strcpy(temp, str);
            }
            else
                strcpy(temp, str);
        }
    }
}
void output(){
    ln=1;
    char lex1[10],lex2[10];
    char i1[]="id";
    f1=fopen("output4.txt","r");
    f2=fopen("output5.txt","w");
    if(!f1)
        printf("File can't be opened");
    else{
        while((c=fgetc(f1))!=EOF) {
            if(c=='\n') {
```

```
    ln++;
}
i=0;
while(c!=' ' && c!=EOF)  {
    lex1[i]=c;
    i++;
    c=fgetc(f1);
}
lex1[i]='\0';
if (checkId(lex1)) {
    if (!Check_keyword(lex1)) {
        fputs(i1,f2);
        fputc(' ',f2);
        fputs(lex1,f2);
        fputc(' ',f2);
    }
    else{
        fputs(lex1,f2);
        fputc(' ',f2);
    }
}
else{
    fputs(lex1,f2);
    fputc(' ',f2);
}
```

```
        }
        fclose(f1);
        fclose(f2);
    }
}
int main()
{
   comment_rmv();
    space_rmv();
    closing_barce_rmv();
    lexim_sep();
    else_rmv();
    token_rmv();
    output();
}
```