27-Jan-22

Experiment 1 - Lexical Analyzer

Dhawal Patil
RA1911003010575
CSE A2

Aim:

Write A Program to implement a Lexical Analyzer and perform Tokenization.

Algorithm:

1.) Initialize variables and create arrays containing operators, keywords, numbers and other special characters.
2.) Read Input file and run it till the end of file.
3.) Get each character of the file and store it in variable ch.
4.) Check if character belongs as a number, operator, or special character and add it into the respective list and mark it as already added to list.
5.) If it does not belong to any it is added to a buffer array that stores characters till a space or a delimiter is found and is then sent to isKeyword function to check if its valid keyword or not.
6.) Store count of all keywords, operators, special, and use it to list them out and get the desired output.

Code:

```
#include<bits/stdc++.h>
#include<stdlib.h>
#include<string.h>
#include<ctype.h>
using namespace std;
int isKeyword(char buffer[]){
   char keywords[36][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", "include" , "cout", "main", "iostream"};
   int i, flag = 0;
   for(i = 0; i < 36; ++i){
      if(strcmp(keywords[i], buffer) == 0){
         flag = 1;
         break;
      }
   }
   return flag;
```

```
}
int main(){
   char ch, buffer[15],b[30], logical_op[] = "><",math_op[]="+-
*/=",numer[]=".0123456789",other[]=",;\(){}[]'':";
   ifstream fin("inputfile.txt");
   int mark[1000]={0};
   int i,j=0,kc=0,ic=0,lc=0,mc=0,nc=0,oc=0,aaa=0;
   vector < string > k;
   vector<char >id;
   vector<char>lo;
   vector<char>ma;
   vector<string>nu;
    vector<char>ot;
   if(!fin.is_open()){
      cout<<"error while opening the file\n";
      exit(0);
   }
   while(!fin.eof()){
         ch = fin.get();
       for(i = 0; i < 12; ++i){
           if(ch == other[i]){
               int aa=ch;
            if(mark[aa]!=1){
               ot.push_back(ch);
               mark[aa]=1;
               ++oc;
            }
           }
      }
      for(i = 0; i < 5; ++i){
           if(ch == math_op[i]){
               int aa=ch;
            if(mark[aa]!=1){
               ma.push_back(ch);
               mark[aa]=1;
               ++mc;
            }
           }
      }
      for(i = 0; i < 2; ++i){
           if(ch == logical_op[i]){
               int aa=ch;
            if(mark[aa]!=1){
```

```
                lo.push_back(ch);
                mark[aa]=1;
                ++lc;
             }
            }
      }
      if(ch=='0' || ch=='1' || ch=='2' || ch=='3' || ch=='4' || ch=='5' || ch=='6' || ch=='7' ||
ch=='8' || ch=='9' || ch=='.' ||ch == ' ' || ch == '\n' || ch == ';'){
         if(ch=='0' || ch=='1' || ch=='2' || ch=='3' || ch=='4' || ch=='5' || ch=='6' || ch=='7' ||
ch=='8' || ch=='9' || ch=='.')b[aaa++]=ch;
          if((ch == ' ' || ch == '\n' || ch == ';') && (aaa != 0)){
               b[aaa] = '\0';
               aaa = 0;
               char arr[30];
                strcpy(arr,b);
                   nu.push_back(arr);
                ++nc;
             }
      }
        if(isalnum(ch)){
           buffer[j++] = ch;
        }
        else if((ch == ' ' || ch == '\n') && (j != 0)){
               buffer[j] = '\0';
               j = 0;
               if(isKeyword(buffer) == 1){
                  k.push_back(buffer);
                  ++kc;
               }
               else{
                if(buffer[0]>=97 && buffer[0]<=122) {
                   if(mark[buffer[0]-'a']!=1){
                   id.push_back(buffer[0]);
                   ++ic;
                   mark[buffer[0]-'a']=1;
                  }
                 }
               }
        }
   }
   fin.close();
   printf("Keywords: ");
    for(int f=0;f<kc;++f){
```

```
        if(f==kc-1){
           cout<<k[f]<<"\n";
        }
        else {
           cout<<k[f]<<", ";
        }
}
printf("Identifiers: ");
 for(int f=0;f<ic;++f){
    if(f==ic-1){
           cout<<id[f]<<"\n";
        }
        else {
           cout<<id[f]<<", ";
        }
}
printf("Math Operators: ");
for(int f=0;f<mc;++f){
        if(f==mc-1){
           cout<<ma[f]<<"\n";
        }
        else {
           cout<<ma[f]<<", ";
        }
}
printf("Logical Operators: ");
for(int f=0;f<lc;++f){
        if(f==lc-1){
           cout<<lo[f]<<"\n";
        }
        else {
           cout<<lo[f]<<", ";
        }
}
printf("Numerical Values: ");
for(int f=0;f<nc;++f){
        if(f==nc-1){
           cout<<nu[f]<<"\n";
        }
        else {
           cout<<nu[f]<<", ";
        }
}
```

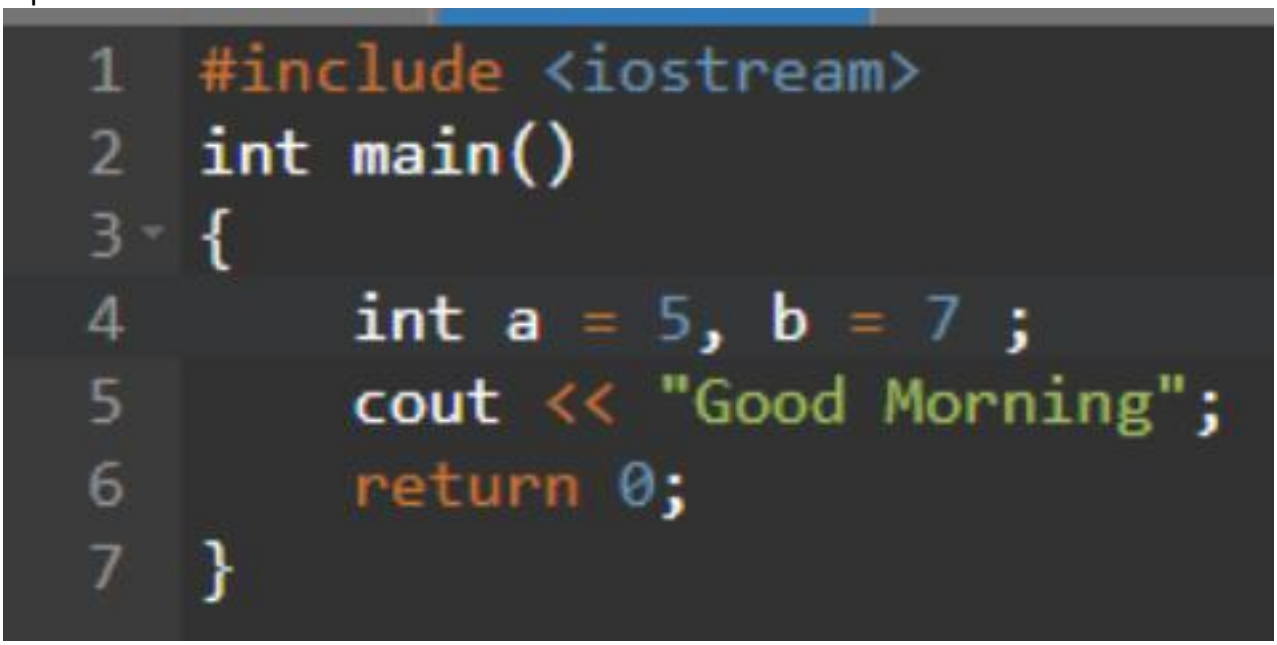
```
    printf("Others: ");
    for(int f=0;f<oc;++f){
        if(f==oc-1){
            cout<<ot[f]<<"\n";
        }
        else {
            cout<<ot[f]<<" ";
        }
    }
    return 0;
}
```

Screenshots:

Output:

```
Keywords: include, iostream, int, main, int, cout, return
Identifiers: a, b
Math Operators: =
Logical Operators: <, >
Numerical Values: 5, 7, 0
Others: ( ) { , ; }
```

Input File:



```
1  #include <iostream>
2  int main()
3  {
4      int a = 5, b = 7 ;
5      cout << "Good Morning";
6      return 0;
7  }
```

Result:

Hence Lexical Analyzer was successfully implemented and the desired result was obtained.