

Virtual Lab Experiment

Design Lexical Analyzer in C/C++ -

```
#include<bits/stdc++.h>

using namespace std;

string keywords[]={"int","float","if","else","while","for"};
string operators[]={"<",">","<=",">=","==","=", "+", "-", "++", "--",
};
char punctuation[]={'(',')','{','}',';','(',')','[',']'};
vector<string> k,o,c,i;
vector<string> p;

/*
This function checks whether the given word is a keyword in
the list of keywords present.
If it is a keyword, it is added into the vector 'k'.
*/

void search_for_keywords(string a ,int flag)
{
if(a.size()==0)return;
int size=sizeof(keywords)/sizeof(keywords[0]);
for(int i=0;i<size;i++)
{
if(a==keywords[i])
{
k.push_back(a);
return;
}
}
if(!flag) i.push_back(a);
}
```

```
/*  
This function checks if the given character is present in  
the given array of punctuation marks.  
If it is present, it is added into the vector 'p'.  
*/
```

```
bool search_for_punctuation(char a )  
{  
    //cout<<a;  
    int size=sizeof(punctuation)/sizeof(punctuation[0]);  
    for(int i=0;i<size;i++)  
    {  
        //cout<<punctuation[i];  
        if(a==punctuation[i])  
        {  
            //cout<<a;  
            string temp=" ";  
            temp[0]=a;  
            2  
            //cout<<temp;  
            p.push_back(temp);  
            return true;  
        }  
    }  
    return false;  
}  
/*
```

```
This function is to create the tokens of integers and  
floatingpoint integers.  
*/
```

```
void search_for_constants(string a )  
{
```

```
if(a.size()>0) c.push_back(a);
```

```
}
```

```
/*
```

This function prints the list/vector of strings and the number of strings which is provided as the input.

```
*/
```

```
void print(vector<string>a )
```

```
{
```

```
cout<<"\n";//cout<<"-----
```

```
- \n";
```

```
for(int i=0;i<a.size();i++)
```

```
{
```

```
cout<<a[i]<<" ";
```

```
}
```

```
cout<<"\nTotal="<<a.size()<<"\n";
```

```
cout<<"-----\n";
```

```
}
```

```
/*
```

This function checks if the given input is a part of the list of operators defined.

If it is a part of the list, it is added into the vector of operators 'o'.

```
*/
```

```
void search_for_operators(string line,int& i)
```

```
{
```

```
// This is to check the operators which are composed of two characters like '++', '+='.
```

```
string temp=line.substr(i,2);
```

```
//cout<<temp<<endl;
```

```
int size=sizeof(operators)/sizeof(operators[0]);
```

```
for(int j=0;j<size;j++)
```

```

{
//cout<<punctuation[i];
if(temp==operators[j])
{
o.push_back(temp);i=i+1;
3
return;
}
}
// This is to check the operators which are composed of
only one character like '+', '-'.
temp=line.substr(i,1);
//cout<<temp<<endl;
for(int j=0;j<size;j++)
{
//cout<<punctuation[i];
if(temp==operators[j])
{
o.push_back(temp);
return;
}
}
}
int main()
{
cout<<"Enter number of lines of input:";
int n;
cin>>n;n++;
while(n--)
{
char arr[100];

```

```

cin.getline(arr,100,'\n');
string line=arr;
//cout<<line<<line.length();
int i;
string cur="";
string cur_num="";
int flag=0,flag2=0;
int no_of_dots=0;
for(i=0;i<line.length();i++)
{
char now=line[i];
if((now>='a'&&now<='z')||(now>='A'&&now<='Z'))
{ // Check for keywords and identifiers starts.
if(now=='e'&&flag==0&&no_of_dots>0){cur_num+=now;continue;}
if(cur_num.size()>0){flag2=1;}
flag=1;
cur+=line[i]; }
else if(now==' '){
// Found a delimiter, hence checking the stored
input till now for keywords and constants.
if(flag)search_for_keywords(cur,flag2);
4
else if(!flag2) search_for_constants(cur_num);
cur="";flag=0;cur_num="";no_of_dots=0; }
else if(now>='0'&&now<='9' || now=='.')
{ //Check for number starts. Keeping count of
number of '.'s.
if(now=='.'&&no_of_dots>0)cur_num="";
else if(now=='.')no_of_dots++;
if(flag)cur+=line[i];
else cur_num+=line[i];

```

```

}
else{
//cout<<now;
// If none of the above conditions pass, this
block of code checks for all of the keywords, numbers, operators
and punctuations.
if((now=='+'||now=='-
')&&flag==0&&cur_num.size()>0){cur_num+=now;continue;}
if(flag)search_for_keywords(cur,flag2);
else if(!flag2) search_for_constants(cur_num);
cur="";flag=0;cur_num="";no_of_dots=0;
if(!search_for_punctuation(now))
search_for_operators(line,i);
; //cout<<now;}}
//If still some are not matched, search for keywords and
numbers.
if(flag)search_for_keywords(cur,flag2);
else if(!flag2) search_for_constants(cur_num);
cur="";flag=0;cur_num="";}
cout<<"\n\nKeywords:";
print(k);cout<<"Operators:";
print(o);cout<<"Constants:";
print(c);cout<<"Punctuation:";
print(p);cout<<"Identifiers:";
print(::i);
cout<<"Total tokens
are:"<<k.size()+o.size()+c.size()+p.size()+::i.size()<<"\n";
return 0;
}

```

The screenshot displays a C++ IDE with the following components:

- Top Bar:** Shows the file name "D:\NTW_CD_Lab\CompilerDesignPrograms\Set_A_Prog... [Executing] - Dev-C++ 5.11".
- Left Panel:** Contains a project tree with "D:\NTW_CD_Lab\CompilerDesignPrograms\Set_A_Prog..." and a list of files including "A2.cpp".
- Main Editor:** Displays the source code of "A2.cpp". The code defines a function `void search_for_keywords` that searches for keywords in a string. It also includes a `main` function that processes a sample code snippet and prints the results.
- Output Window:** Shows the execution results, including the number of lines of input (12), the input string, and the results of the search for keywords, constants, and operators. The output indicates that the program found 12 keywords, 1 constant, and 1 operator.
- Compiler Messages:** A warning message is displayed at the bottom: "Warning: matches this 'Y' under old rules". This message is repeated three times, corresponding to the three occurrences of the 'Y' character in the sample code.

The screenshot shows the Visual Studio IDE with the following components:

- Title Bar:** D:\NTW_CD_Lab\CompilerDesignPrograms\Set_A_Program\A2.cpp - [Executing] - Dev-C++ 5.11
- Menu Bar:** File, Edit, Search, View, Project, Execute, Tools, A2.exe, Window, Help.
- Toolbar:** Standard Visual Studio toolbar icons.
- Project Explorer:** Shows the project structure with files like glibale.h and A2-app.
- Code Editor:** Contains the C++ source code for Set_A_Program.A2.cpp, which includes functions for searching keywords, constants, punctuation, and operators in a string.
- Output Console:** Displays the program's execution results:


```
Enter number of lines of Input:3
int int tmp size:
float before 23.5
is 28 c3;

Keywords:
int int float
Total:3

Operators:
Total:0

Constants:
23;
Total:1

Punctuation:
!
Total:2

Identifiers:
sin qst hello
Total:3

Total tokens are:19

Process exited after 68.22 seconds with return value 0
Press any key to continue . . .
```
- Status Bar:** Line: 44, Col: 2, Sel: 0, Lines: 140, Length: 5201, Insert mode, Done parsing in 0 seconds.