**PROGRAM – 2**

**AIM :** WAP to identify keywords , operators and Identifiers . Give error when identifier is invalid . Take input from source file .

**PROGRAM :**

#include <bits/stdc++.h>

using namespace std ;

int main() {

ifstream file1("test2.cpp") ;

ifstream keyw("keywords.txt") ;

ifstream ops("operators.txt") ;

vector<string> keywords ;

vector<string> operators ;

char arr[] = {' ' , ',' , '(' , ')' , ';' , '\t'} ;

vector<char> delimiters (arr, arr + sizeof(arr) / sizeof(arr[0]));

string key\_word ;

while(getline(keyw , key\_word)) {

keywords . push\_back(key\_word) ;

}

string op ;

while(getline(ops , op)) {

operators . push\_back(op) ;

}

if(file1 . is\_open()) {

string str ;

int i = 1 ;

bool main\_started = false ;

while(getline(file1 , str)) {

int x = 0 ;

while(str[x] == ' ')

x ++ ;

if(str[x] == '#' && ! main\_started) {

cout << "Macros : " << str << " , Line : " << i << "\n" ;

} else {

string token = "" ;

for(int k = 0 ; k < str . length() ; k ++) {

string temp = string(1, str[k]) ;

if(find(delimiters . begin() , delimiters . end() , str[k]) != delimiters . end() || find(operators . begin() , operators . end() , temp) != operators . end()) {

if(token == "main")

main\_started = true ;

if(find(operators . begin() , operators . end() , temp) != operators . end()) {

string temp1 = string(1 , str[k + 1]) ;

temp1 = temp + temp1 ;

if(!(find(operators . begin() , operators . end() , temp1) != operators . end())) {

cout << "Operator : " << temp << " , Line : " << i << " , Position : " << k << "\n" ;

} else {

string temp2 = string(1 , str[k + 2]) ;

temp2 = temp1 + temp2 ;

if(!(find(operators . begin() , operators . end() , temp2) != operators . end())) {

cout << "Operator : " << temp1 << " , Line : " << i << " , Position : " << k << "\n" ;

k ++ ;

} else {

cout << "Operator : " << temp2 << " , Line : " << i << " , Position : " << k << "\n" ;

k += 2 ;

}

}

}

if(find(keywords . begin() , keywords . end() , token) != keywords . end()) {

cout << "Keyword : " << token << " , Line : " << i << " , Position : " << k - token . length() << "\n" ;

} else {

bool isNum = true ;

for(int x = 0 ; x < token . length() ; x ++) {

int asci = (int)token[x] ;

if(asci < 48 || asci > 57) {

isNum = false ;

break ;

}

}

if(token != "" && !isNum && !(find(operators . begin() , operators . end() , token) != operators . end())) {

bool is\_id = true ;

int asc = (int)token[0] ;

if(!((asc >= 65 && asc <= 90) || (asc >= 97 && asc <= 122) || (asc == 95))) {

if(!((token[0] == '"' && token[token . length() - 1] == '"') || (token[0] == '\'' && token[token . length() - 1] == '\''))) {

cout << "Invalid Identifier : " << token << " , Error at line : " << i << " pos : " << k - token . length() << " , Identifier must start with an alphabet or an underscore ...\n" ;

}

is\_id = false ;

} else {

for(int z = 1 ; z < token . length() ; z ++) {

asc = (int)token[z] ;

if(!((asc >= 65 && asc <= 90) || (asc >= 97 && asc <= 122) || (asc >= 48 && asc <= 57) || (asc == 95))) {

cout << "Invalid Identifier : " << token << " , Error at line : " << i << " pos : " << k - token . length() + z << " , Identifier can not contain " << token[z] << " ...\n" ;

is\_id = false ;

}

}

}

if(is\_id) {

cout << "Identifier : " << token << " , Line : " << i << " , Position : " << k - token . length() << "\n" ;

}

}

}

token = "" ;

} else {

token += str[k] ;

}

}

}

i ++ ;

cout << "\n" ;

}

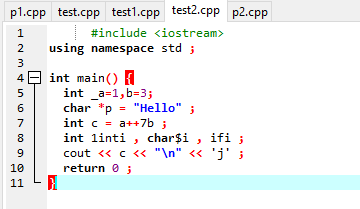
file1 . close() ;

}

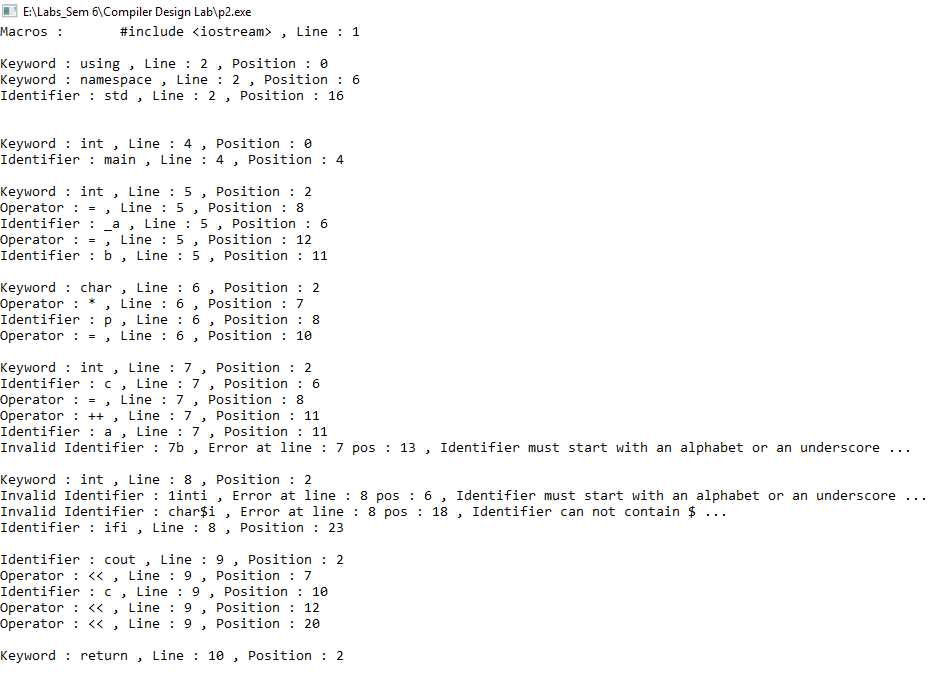
return 0 ;

}

**Input File :**

****

**Output :**

****