NAME : UTKARSH AGRAWAL

REG NO : RA1911003010307  
SECTION : E1

**Compiler Design (18CSC304J )**

**Exp\_01 - Implementation of Lexical Analyser**

**Source Code:**

keywords = {"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","printf","scanf","%d","include","stdio.h","main"}

operators = {"+","-","\*","/","<",">","=","<=",">=","==","!=","++","--","%"}

delimiters = {'(',')','{','}','[',']','"',"'",';','#',',',''}

def detect\_keywords(text):

arr = []

for word in text:

if word in keywords:

arr.append(word)

return list(set(arr))

def detect\_operators(text):

arr = []

for word in text:

if word in operators:

arr.append(word)

return list(set(arr))

def detect\_delimiters(text):

arr = []

for word in text:

if word in delimiters:

arr.append(word)

return list(set(arr))

def detect\_num(text):

arr = []

for word in text:

try:

a = int(word)

arr.append(word)

except:

pass

return list(set(arr))

def detect\_identifiers(text):

k = detect\_keywords(text)

o = detect\_operators(text)

d = detect\_delimiters(text)

n = detect\_num(text)

not\_ident = k + o + d + n

arr = []

for word in text:

if word not in not\_ident:

arr.append(word)

return arr

with open('file.c') as t:

text = t.read().split()

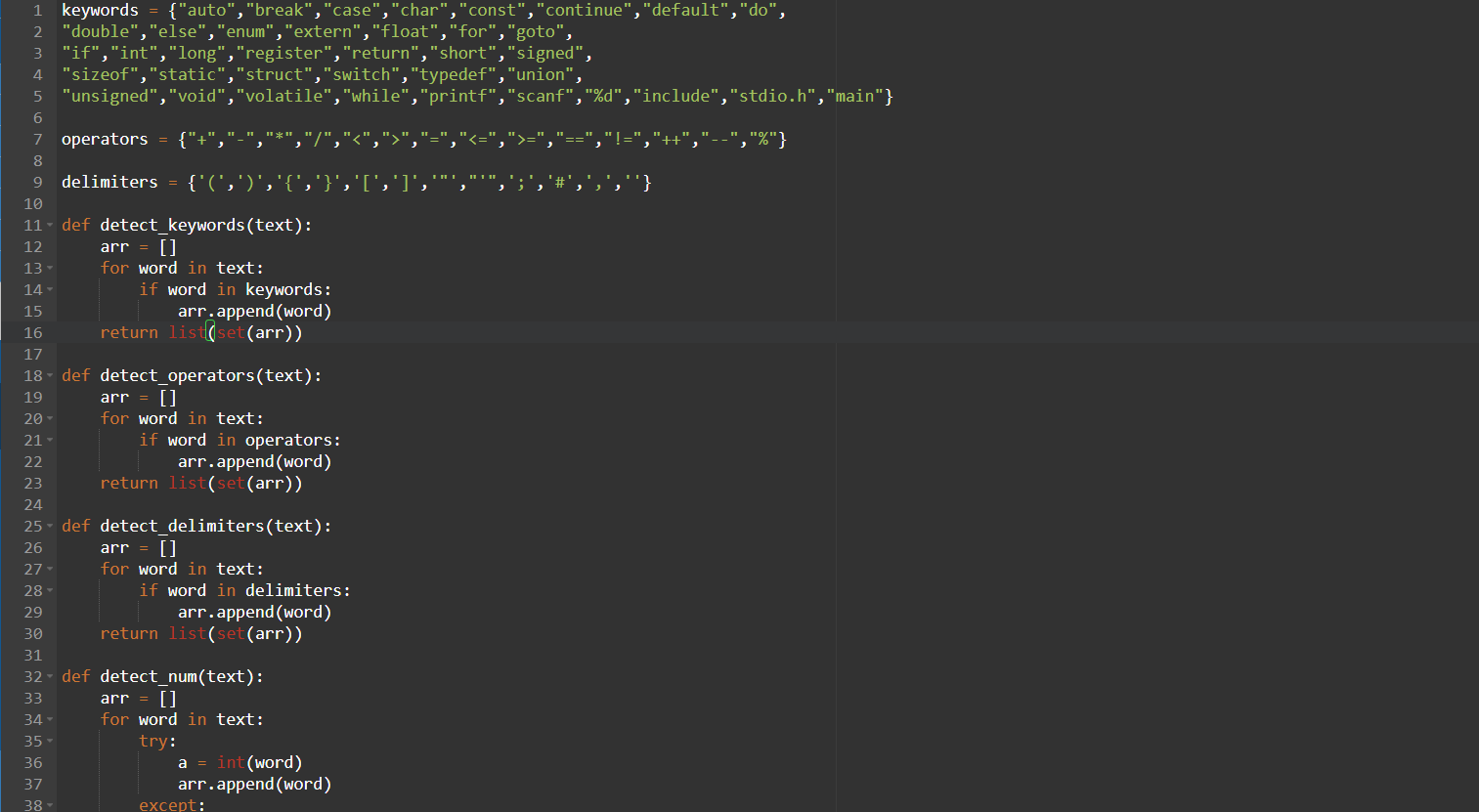
print("Keywords: ",detect\_keywords(text))

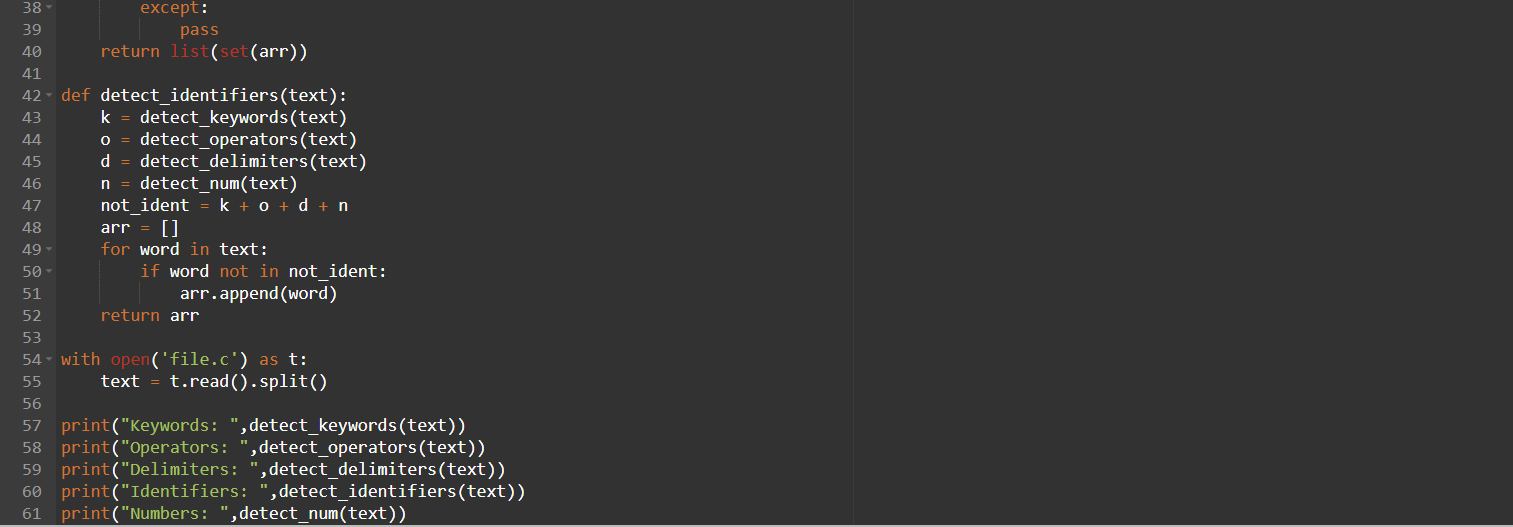
print("Operators: ",detect\_operators(text))

print("Delimiters: ",detect\_delimiters(text))

print("Identifiers: ",detect\_identifiers(text))

print("Numbers: ",detect\_num(text))





**Input Code:**

# include < stdio.h >

void main ( ) {

int a , b , c ;

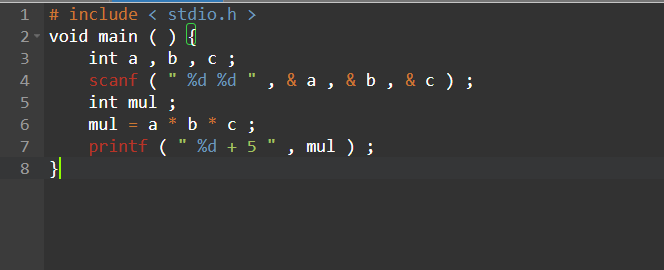
scanf ( " %d %d " , & a , & b , & c ) ;

int mul ;

mul = a \* b \* c ;

printf ( " %d + 5 " , mul ) ;

}



**Output:**

