

Compiler Design

Exp_01 - Implementation of Lexical Analyser

Ayush Jindal

RA1911003010308

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))

```

```

1 keywords = {"auto","break","case","char","const","continue","default","do",
2 "double","else","enum","extern","float","for","goto",
3 "if","int","long","register","return","short","signed",
4 "sizeof","static","struct","switch","typedef","union",
5 "unsigned","void","volatile","while","printf","scanf","%d","include","stdio.h","main"}
6
7 operators = {"+", "-", "*", "/", "<", ">", "=", "<=", ">=", "==", "!=", "++", "--", "%"}
8
9 delimiters = {'(', ')', '{', '}', '[', ']', '"', "'", ';', '#', ',', '\n'}
10
11 def detect_keywords(text):
12     arr = []
13     for word in text:
14         if word in keywords:
15             arr.append(word)
16     return list(set(arr))
17
18 def detect_operators(text):
19     arr = []
20     for word in text:
21         if word in operators:
22             arr.append(word)
23     return list(set(arr))
24
25 def detect_delimiters(text):
26     arr = []
27     for word in text:
28         if word in delimiters:
29             arr.append(word)
30     return list(set(arr))
31
32 def detect_num(text):
33     arr = []
34     for word in text:
35         try:
36             a = int(word)
37             arr.append(word)
38         except:
39             pass
40     return list(set(arr))
41
42 def detect_identifiers(text):
43     k = detect_keywords(text)
44     o = detect_operators(text)
45     d = detect_delimiters(text)
46     n = detect_num(text)
47     not_ident = k + o + d + n
48     arr = []
49     for word in text:
50         if word not in not_ident:
51             arr.append(word)
52     return arr
53
54 with open('file.c') as t:
55     text = t.read().split()
56
57 print("Keywords: ",detect_keywords(text))
58 print("Operators: ",detect_operators(text))
59 print("Delimiters: ",detect_delimiters(text))
60 print("Identifiers: ",detect_identifiers(text))
61 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 ) ;
}

```

```

1 # include < stdio.h >
2 void main ( ) {
3     int a , b , c ;
4     scanf ( " %d %d " , & a , & b , & c ) ;
5     int mul ;
6     mul = a * b * c ;
7     printf ( " %d + 5 " , mul ) ;
8 }

```

Output:

```

Keywords: ['scanf', 'stdio.h', 'include', 'printf', '%d', 'main', 'int', 'void']
Operators: ['>', '=', '+', '<', '*']
Delimiters: ['(', ')', ';', '#', '"', ',', '(', ')']
Identifiers: ['a', 'b', 'c', '&', 'a', '&', 'b', '&', 'c', 'mul', 'mul', 'a', 'b', 'c', 'mul']
Numbers: ['5']

```