|  |  |
| --- | --- |
| **Ex No:1** | **Token Separation** |
| **Date of Exercise** | **13/01/2022** |

**Questions**

1. Write a program to perform token separation for a given subset of a language.

**Aim**

To write a program to perform token separation for a given subset of language.

**Description**

**Lexical Analysis**

Scanning is the first phase of a compiler in which the source program is read character by character and then grouped in to various tokens. Token is defined as sequence of characters with collective meaning. The various tokens could be identifiers, keywords, operators, punctuations, constants, etc. The input is a program written in any high level language and the output is stream of tokens. Regular expressions can be used for implementing this token separation

During the Lexical analysis phase of the compiler, the source program will be read from left to right and get grouped into TOKENS. Tokens can be any one of the following

* Keywords (if, while, do etc)
* identifiers (num, a, b, c etc)
* Punctuation Symbols (:,”,’, .,; etc)
* operators (+, -, /, \*, &gt;, &lt;, = etc)

The Keywords, Operators and Punctuation symbols should be declared and defined in the character array. When the input expression is given each character should be checked and Keywords, Operators, Punctuation symbols, Identifiers should be displayed separately in a table format.

**Program**

import re

keywords = ['if','else','while','int','float','double','for'];

ic = 0

tc = 0

while True:

F = False

sentence = re.split('([^a-zA-Z0-9\_])',input())

for i in sentence:

if i=='' or i==' ':

continue;

tc+=1

if(i=="end"):

F = True

break

elif(i in keywords):

print('< '+i+' , Keyword >')

elif(re.match(r"[;,'\".]+$",i)):

print('< '+i+' , Punctuation >')

elif(re.match(r"[<>=+-/\*]+",i)):

print('< '+i+' , Operator >')

elif(re.match(r"[a-z][\_0-9a-z]\*$",i)):

ic += 1

print('< '+i+' , Identifier id-'+str(ic)+' >')

elif(re.match(r"[0-9]+$",i)):

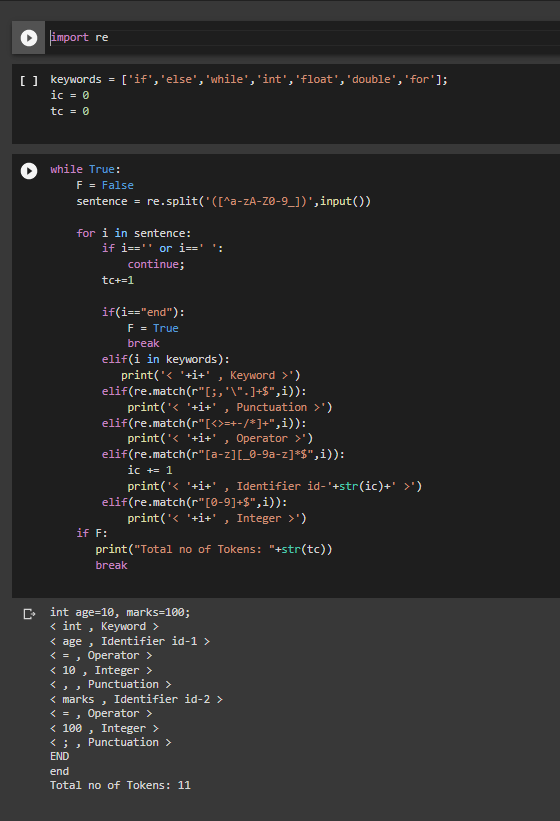
print('< '+i+' , Integer >')

if F:

print("Total no of Tokens: "+str(tc))

break

**Output Screenshot**



**Result**

Thus the program ran successfully and the output was verified.