[Skip to content](https://github.com/jrsinha2/lexical-analyser/blob/master/lexi.py" \l "start-of-content)

* Why GitHub?
* [Enterprise](https://github.com/enterprise)
* Explore
* [Marketplace](https://github.com/marketplace)
* Pricing

Top of Form



Bottom of Form

[**lexical-analyser**](https://github.com/jrsinha2/lexical-analyser)**/lexi.py**

|  |  |  |
| --- | --- | --- |
|  | import re | |
|  | keyword = ['Begin','End','or', 'exe', 'test', 'int', 'repeat', 'alpha', 'times', 'bool'] | |
|  | oper = ['+', '-', '\*', '/', '=', '<', '>', '<=', '>=', '==', '!='] | |
|  | delim = ['\t','\n','(',')','{','}','[',']', ' ',] | |
|  | num = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9'] | |
|  | boolean=['True','False'] | |
|  | alpha =['a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','z','Q','W','E','R','T','Y','U','I','O','P','A','S','D','F','G','H','J','K','L','Z','X','C','V','B','N','M'] | |
|  | def is\_key(kw): | |
|  | if kw in keyword: | |
|  | return True | |
|  | return False | |
|  |  | |
|  | def is\_bool(key): | |
|  | if key in boolean: | |
|  | return True | |
|  | return False | |
|  |  | |
|  | def is\_delim(char): | |
|  | if char in delim: | |
|  | return True | |
|  | return False | |
|  |  | |
|  | def is\_digit(char): | |
|  | if char in num: | |
|  | return True | |
|  | elif char.isdigit()==True: | |
|  | return True | |
|  | return False | |
|  |  | |
|  | def is\_alpha(char): | |
|  | if char in alpha: | |
|  | return True | |
|  | return False | |
|  |  | |
|  | def is\_assignope(char): | |
|  | if char =='$': | |
|  | return True | |
|  | return False | |
|  |  | |
|  | def is\_seperator(char): | |
|  | if char ==',': | |
|  | return True | |
|  | return False | |
|  | def is\_oper(char): | |
|  | if char in oper: | |
|  | return True | |
|  | return False | |
|  |  | |
|  | fo=open("input",'r') | |
|  | temp=fo.read() | |
|  | lexims=re.findall("\s\*(?:(\d+)|(\w+)|(.))",temp) | |
|  | for i in range(len(lexims)): | |
|  | for j in range(len(lexims[i])): | |
|  | if(lexims[i][j]!=''): | |
|  | print(lexims[i][j], end=' ') | |
|  | if(is\_key(lexims[i][j])==True): | |
|  | print("Keyword") | |
|  | elif(is\_bool(lexims[i][j])==True): | |
|  | print("Boolean Values") | |
|  | elif(is\_delim(lexims[i][j])==True): | |
|  | print("Braces") | |
|  | elif(is\_digit(lexims[i][j])==True): | |
|  | print("Integer") | |
|  | elif(is\_alpha(lexims[i][j])==True): | |
|  | print("Alphabet(character)") | |
|  | elif(is\_oper(lexims[i][j])==True): | |
|  | print("Operator") | |
|  | elif(is\_assignope(lexims[i][j])==True): | |
|  | print("Assignment Operator") | |
|  | elif(is\_seperator(lexims[i][j])==True): | |
|  | print("Seperator Operator") | |
|  |  | |
|  | else: | |
|  | print("Variable")  o/p | |
|  | | Begin Keyword |
|  | | int Keyword |
|  | | a1 Variable |
|  | | = Operator |
|  | | 3 Integer |
|  | | $ AssignmentOperator |
|  | | int Keyword |
|  | | c1 Variable |
|  | | = Operator |
|  | | 2 Integer |
|  | | $ AssignmentOperator |
|  | | alpha Keyword |
|  | | b1 Variable |
|  | | = Operator |
|  | | Aditya Variable |
|  | | $ AssignmentOperator |
|  | | test Keyword |
|  | | ( Braces |
|  | | a1 Variable |
|  | | > Operator |
|  | | a2 Variable |
|  | | ) Braces |
|  | | End Keyword |