CC ASSIGNMENT 2

Name	**Areeba Farooq**
Registration Number	**200901058**
Batch & Section	**BSCS 01 (SECTION A)**
Instructor's Name	**Mam Tayyaba**

Date:30-12-2022

Module 1

Implementation of lexical analyzer

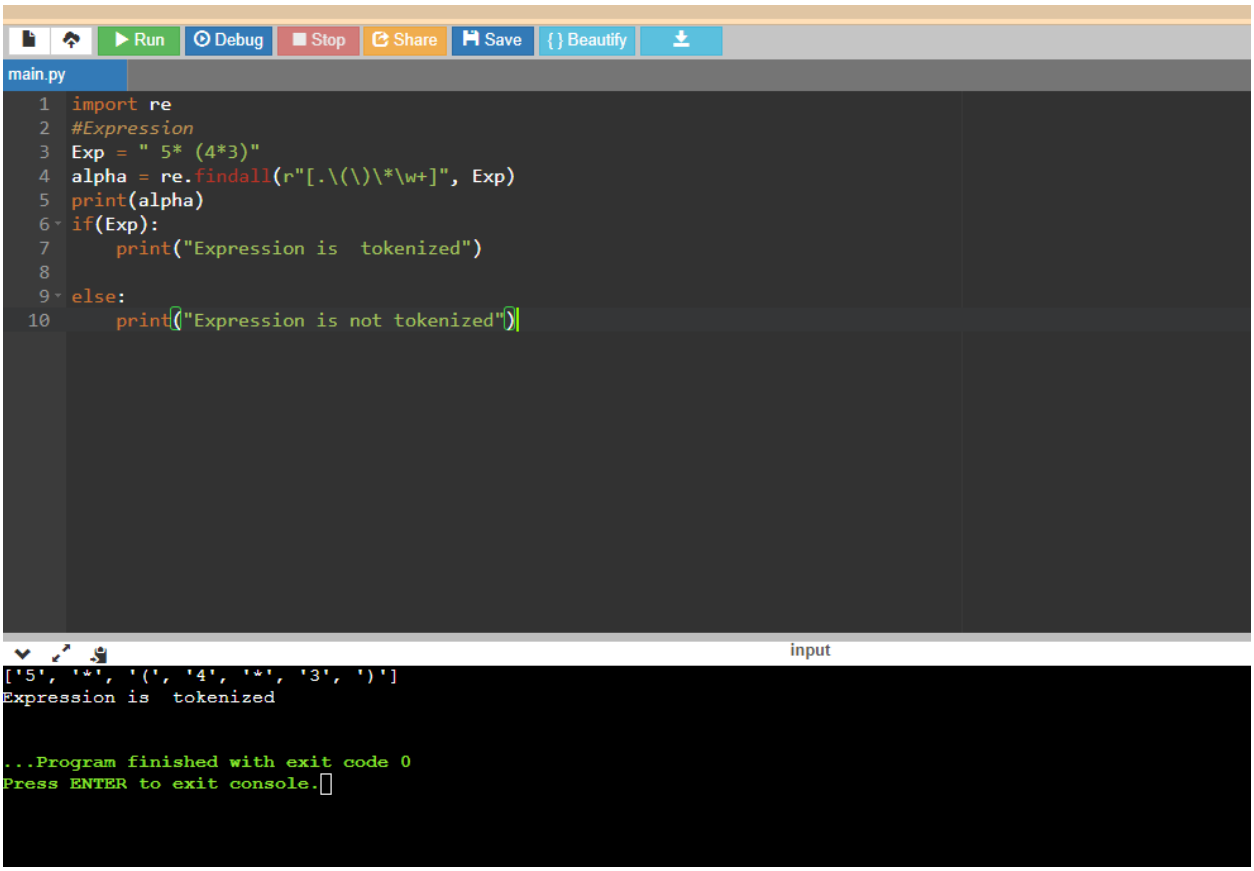
Tokenization of expression (expression can be i.e a + (b*c) or 3+ (5*2) digits, alphabets, characters)

- **Building regex for the expression**
- **Output tags/ tokens of the expression (i.e. ['a', '+', '(', 'b', '*', 'c', ')']**

We will use re library for this module

Code:

```
import re

#Expression

Exp = " 5* (4*3)"

alpha = re.findall(r"[.\(\)\*\w+]", Exp)

print(alpha)

if(Exp):

    print("Expression is  tokenized")

else:

    print("Expression is not tokenized")
```



Expression:

5+ (9/7+8+Areeba)

Code:

```
import re

#Expression

Exp = " 5+ (9/7+8+Areeba)"

x = re.findall(r"[.\(\)\*\w+]", Exp)

print(x)
```

```
main.py
import re
#Expression
Exp = " 5+ (9/7+8+Areeba)"
x = re.findall(r"[.\(\)\*\w+]", Exp)
print(x)
```

```
input
['5', '+', '(', '9', '7', '+', '8', '+', 'A', 'r', 'e', 'e', 'b', 'a', ')']

...Program finished with exit code 0
Press ENTER to exit console.
```

Module 2

Implementation of syntax tree using AST library of python

For implementation of this we will use AST library of python. AST is also known as Abstract Syntax Tree, is a python tool that is used to directly interact with python code and modify them.

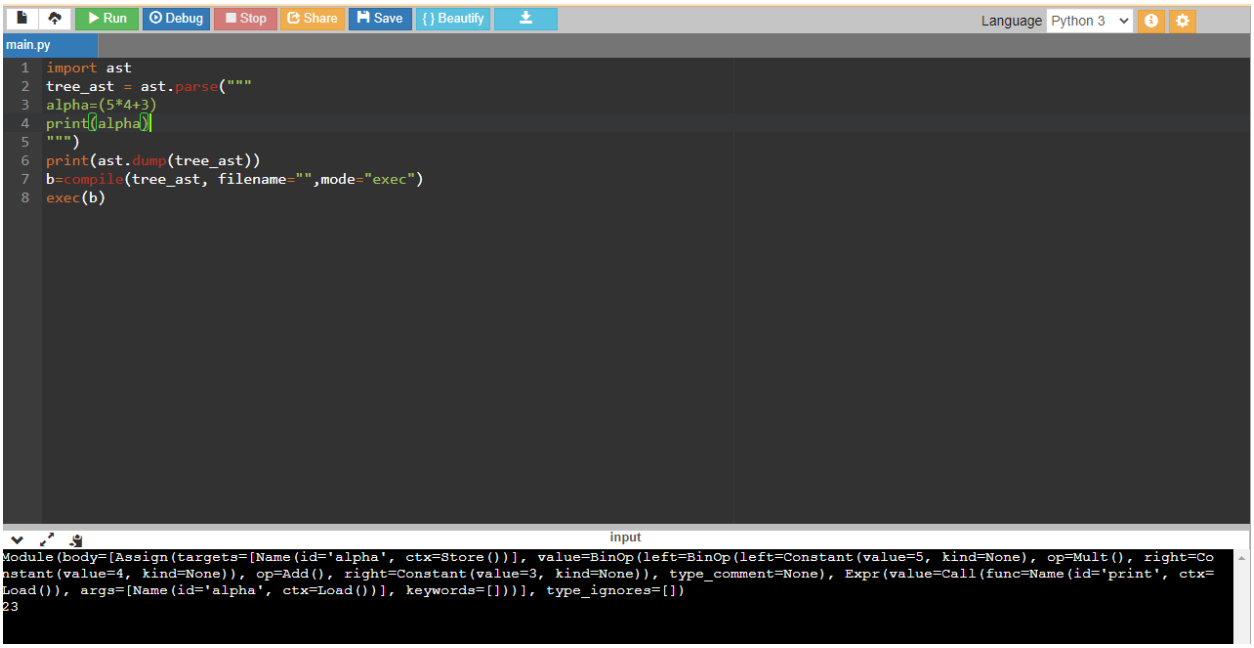
alpha=(5*4+3)

Implementation

```
import ast

tree_ast = ast.parse("""

alpha=(5*4+3)

print(alpha)

""")

print(ast.dump(tree_ast))
```

b=compile(tree_ast, filename="",mode="exec")

exec(b)



x=(5+7/9)

<u>Code</u>

import ast

tree_ast = ast.parse("""

x=(5+7/9)

print(x)

""")

print(ast.dump(tree_ast))

b=compile(tree_ast, filename="",mode="exec")

exec(b)

main.py

```python
import ast
tree_ast = ast.parse("""
x=(5+7/9)
print(x)
""")
print(ast.dump(tree_ast))
b=compile(tree_ast, filename="",mode="exec")
exec(b)
```

input

```
Module(body=[Assign(targets=[Name(id='x', ctx=Store())], value=BinOp(left=Constant(value=5, kind=None), op=Add(), right=BinOp(left=Constan
t(value=7, kind=None), op=Div(), right=Constant(value=9, kind=None))), type_comment=None), Expr(value=Call(func=Name(id='print', ctx=Load(
)), args=[Name(id='x', ctx=Load())], keywords=[]))], type_ignores=[])
5.777777777777778


...Program finished with exit code 0
Press ENTER to exit console.
```

GitHub link:

https://github.com/AREEBA-FAROOQ2001/IST