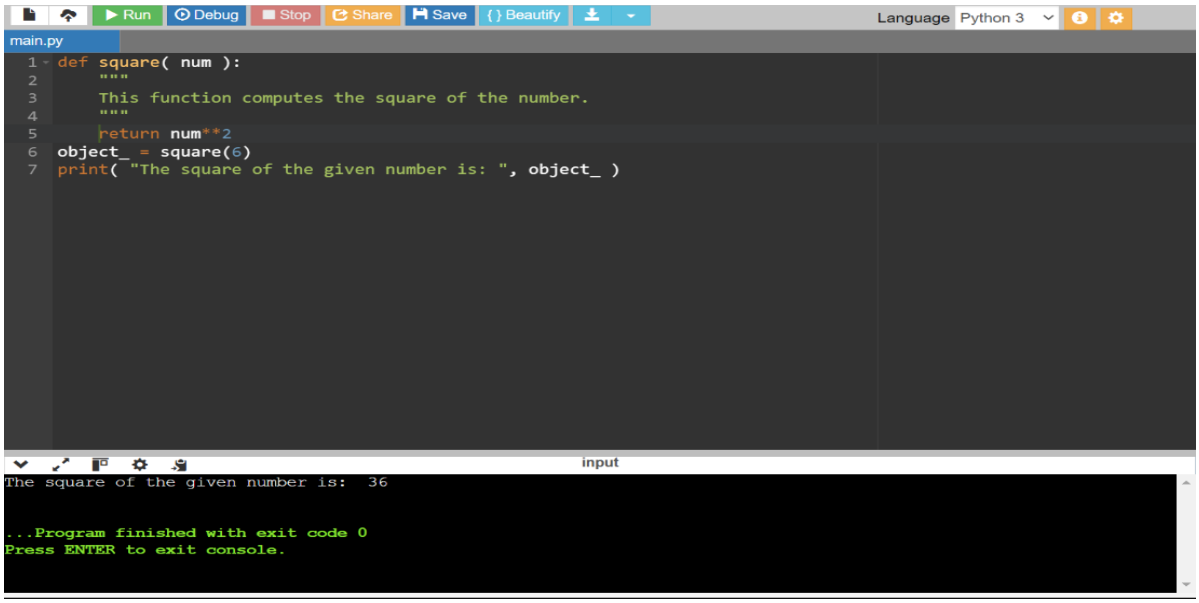
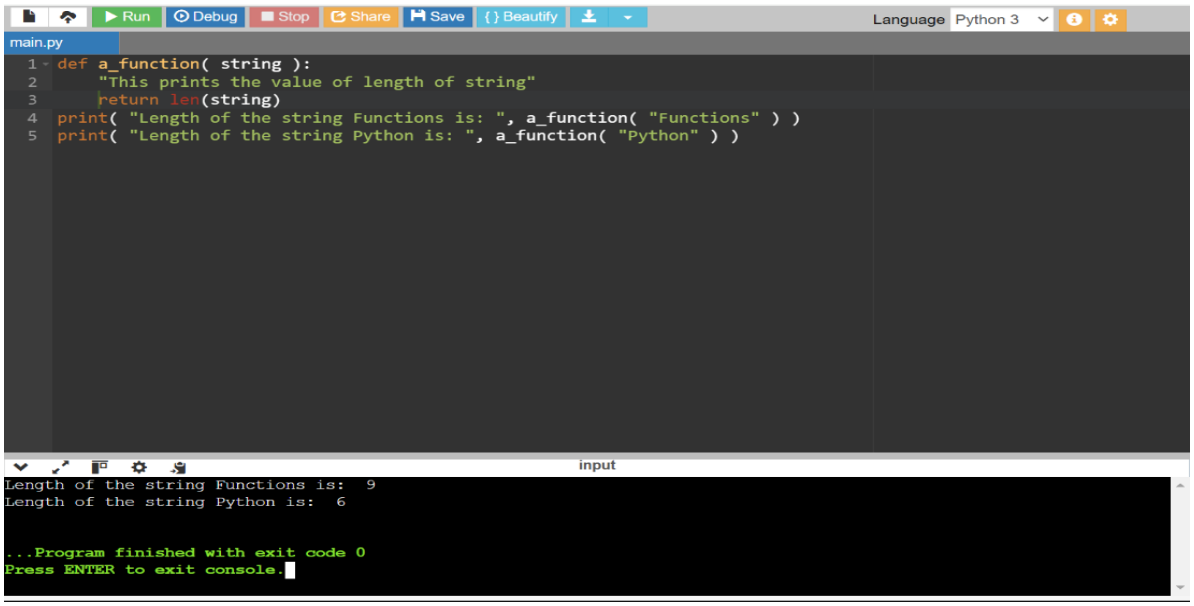
FUNCTION

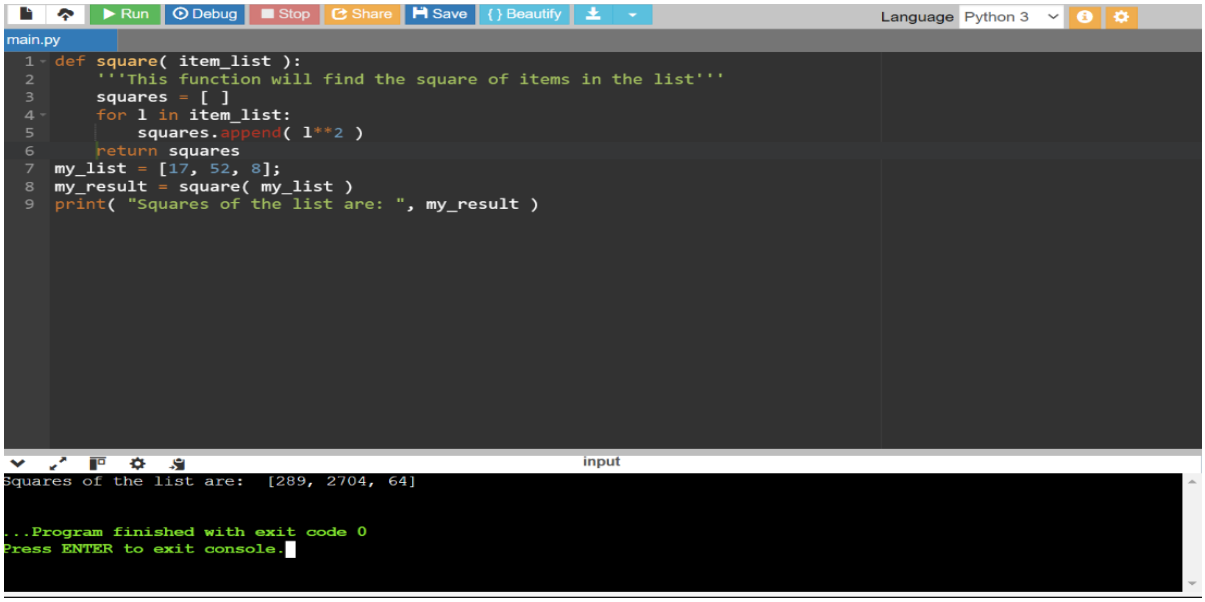
1. A function that returns the argument number's square when called.



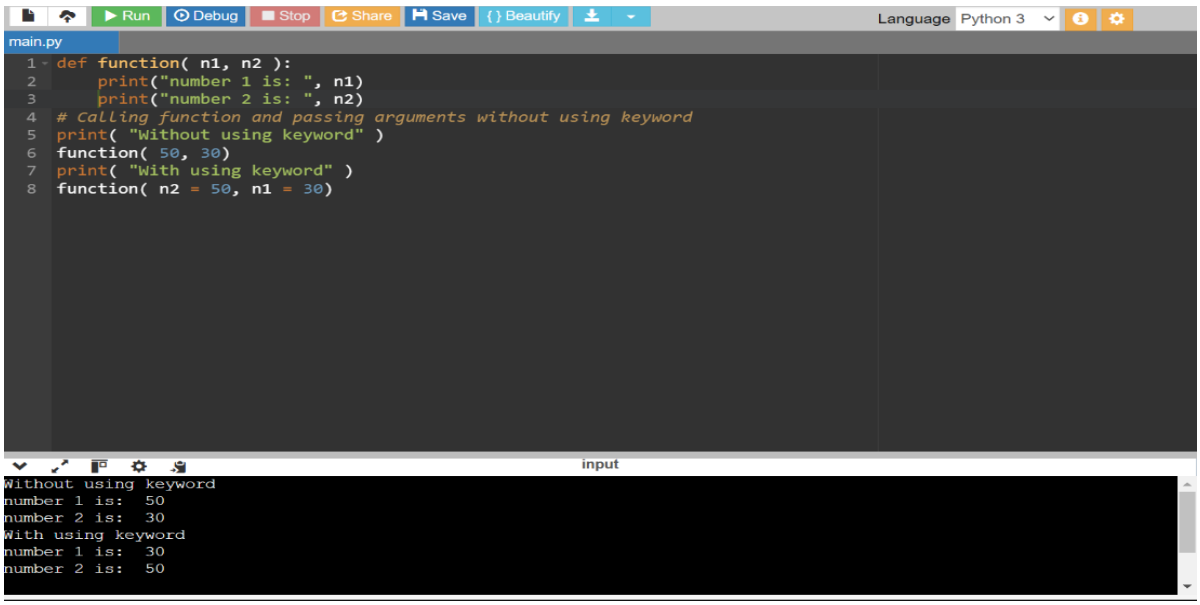
2. Example Python Code for calling a function.



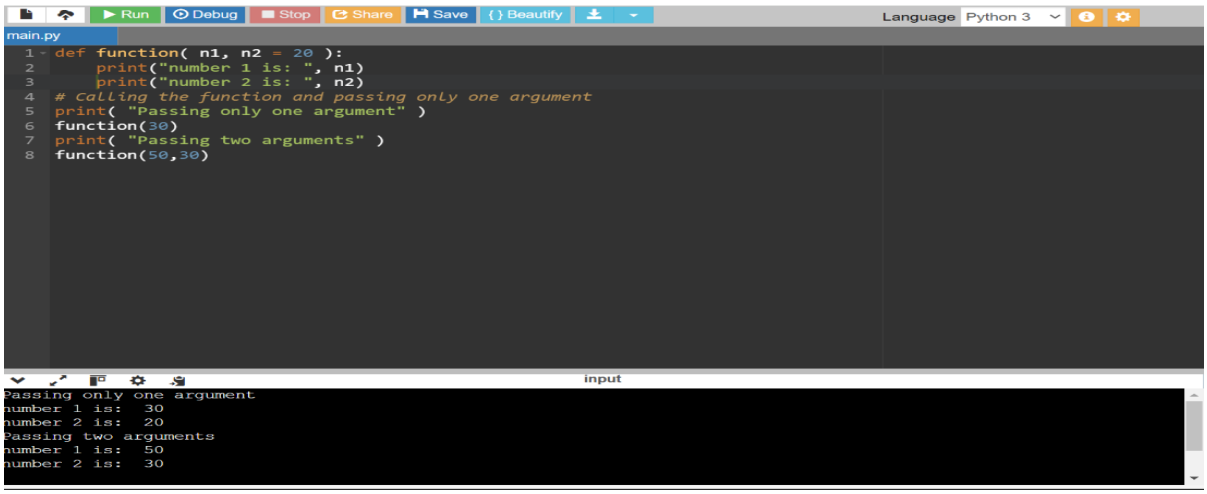
3. Python Code for Pass by Reference vs. Value



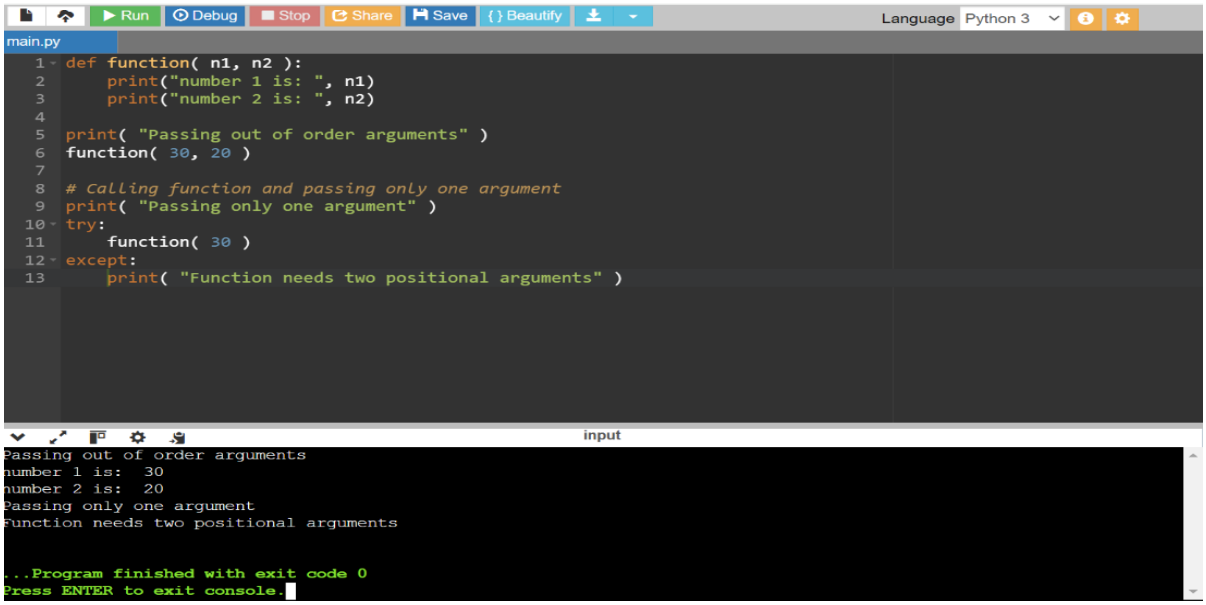
4. Python code to demonstrate the use of default arguments



5. Python code to demonstrate the use of keyword arguments



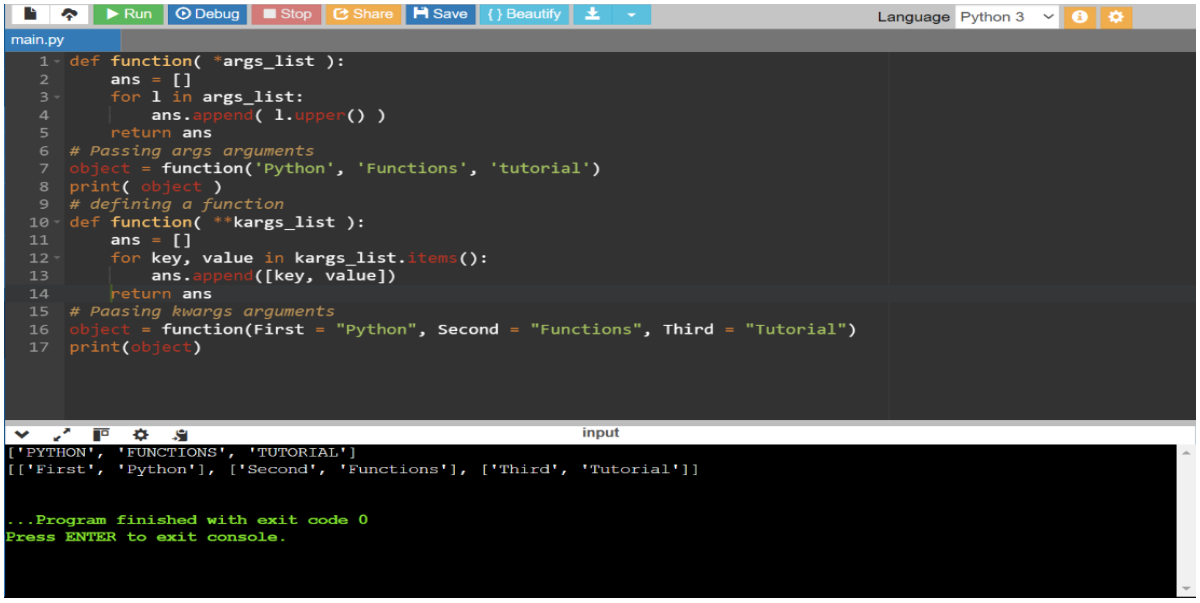
6. Python code to demonstrate the use of default arguments



```
def function( n1, n2 ):
    print("number 1 is: ", n1)
    print("number 2 is: ", n2)

print( "Passing out of order arguments" )
function( 30, 20 )

# Calling function and passing only one argument
print( "Passing only one argument" )
try:
    function( 30 )
except:
    print( "Function needs two positional arguments" )
```

```
Passing out of order arguments
number 1 is:  30
number 2 is:  20
Passing only one argument
Function needs two positional arguments


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

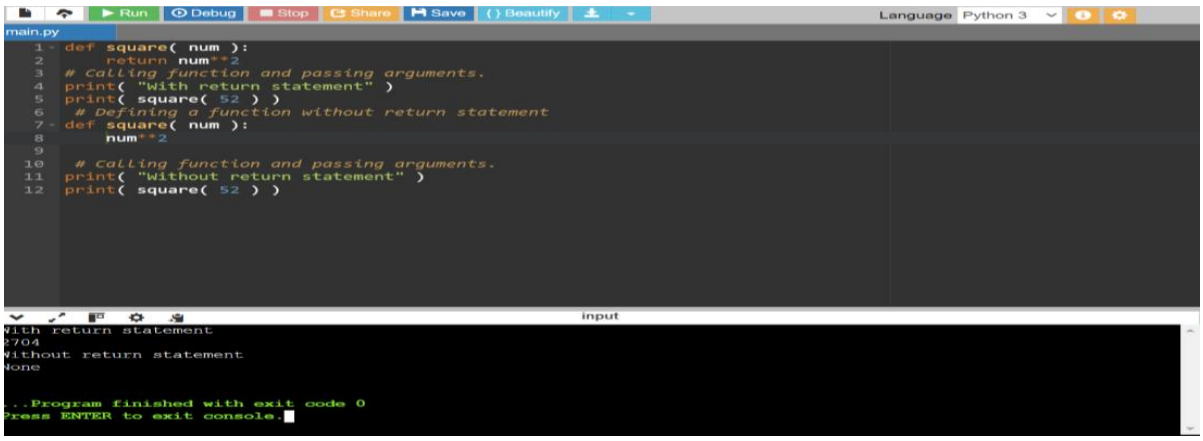
7. Python code to demonstrate the use of variable-length arguments



```
def function( *args_list ):
    ans = []
    for l in args_list:
        ans.append( l.upper() )
    return ans
# Passing args arguments
object = function('Python', 'Functions', 'tutorial')
print( object )
# defining a function
def function( **kargs_list ):
    ans = []
    for key, value in kargs_list.items():
        ans.append([key, value])
    return ans
# Paasing kwargs arguments
object = function(First = "Python", Second = "Functions", Third = "Tutorial")
print(object)
```

```
['PYTHON', 'FUNCTIONS', 'TUTORIAL']
[['First', 'Python'], ['Second', 'Functions'], ['Third', 'Tutorial']]


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

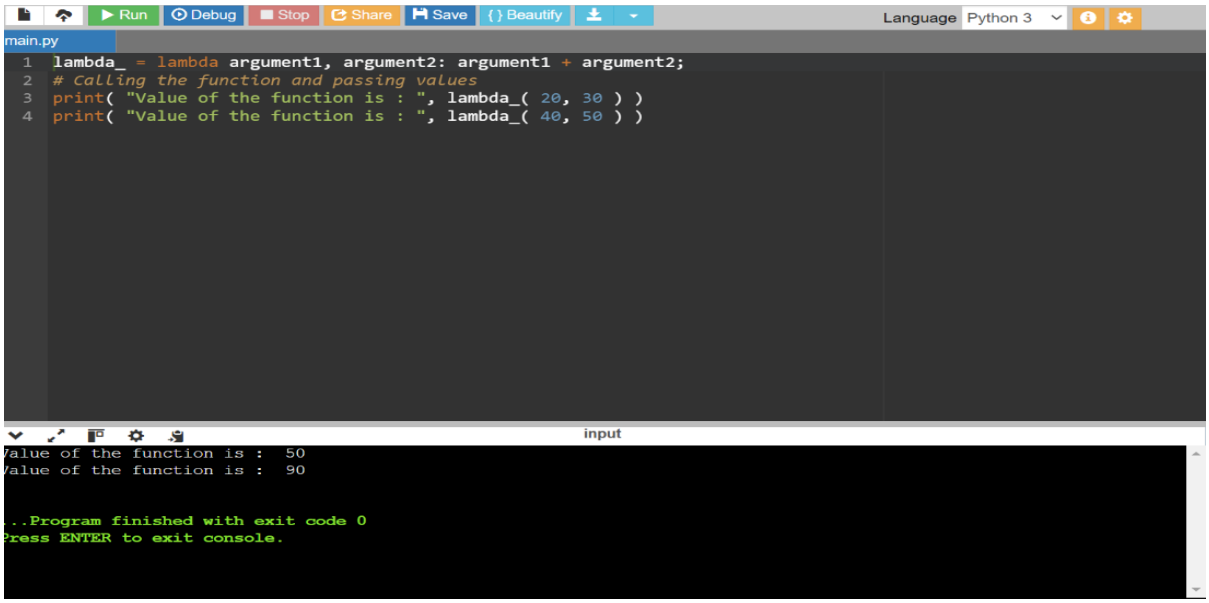
8. Python code to demonstrate the use of return statements



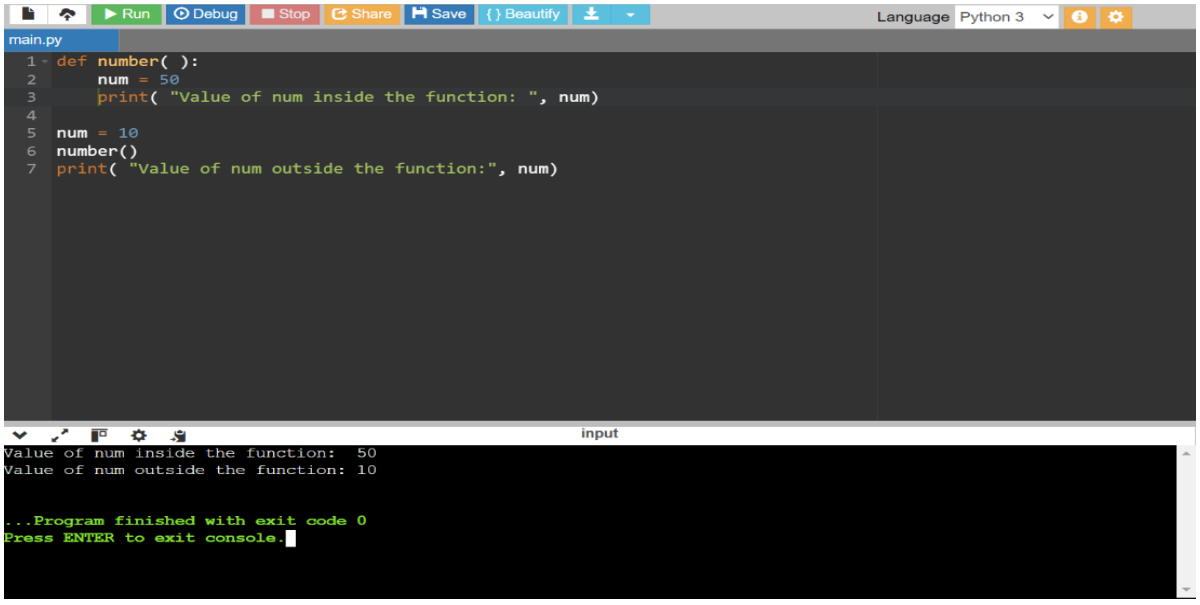
```
def square( num ):
    return num**2
# Calling function and passing arguments.
print( "With return statement" )
print( square( 52 ) )
 # Defining a function without return statement
def square( num ):
    num**2

 # Calling function and passing arguments.
print( "Without return statement" )
print( square( 52 ) )
```

```
With return statement
2704
Without return statement
None


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

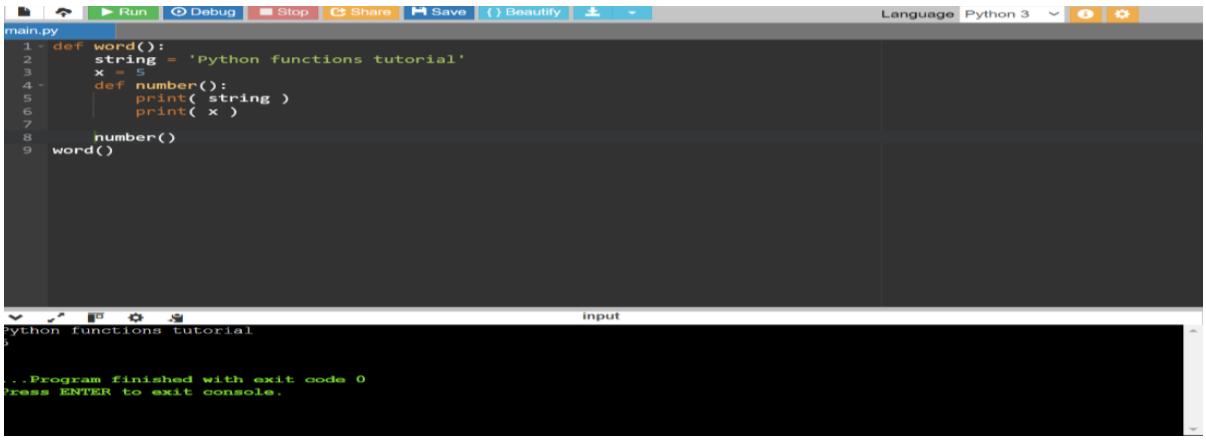
9. Python code to demonstrate ananymous functions



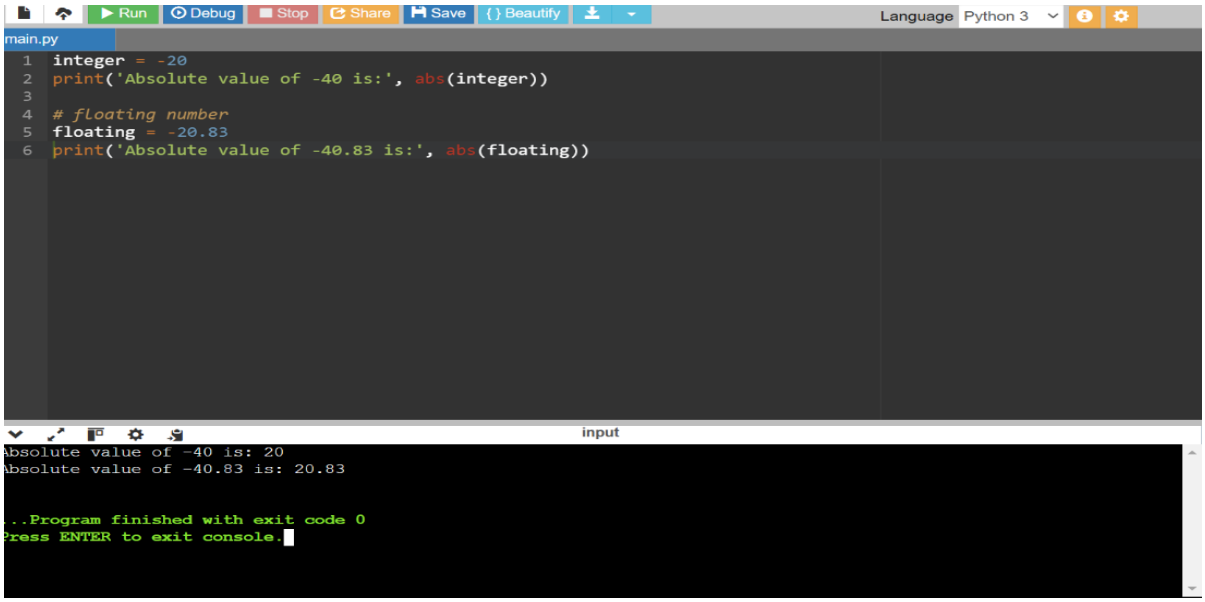
10. Python code to demonstrate scope and lifetime of variable



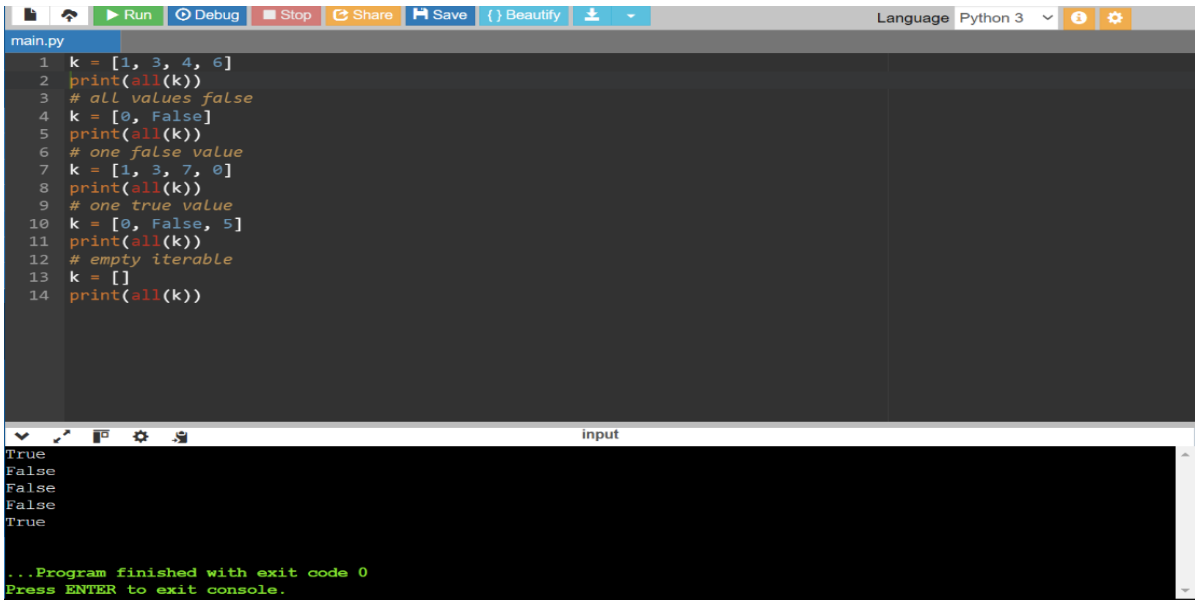
11. Python code to show how to access variables of a nested functions



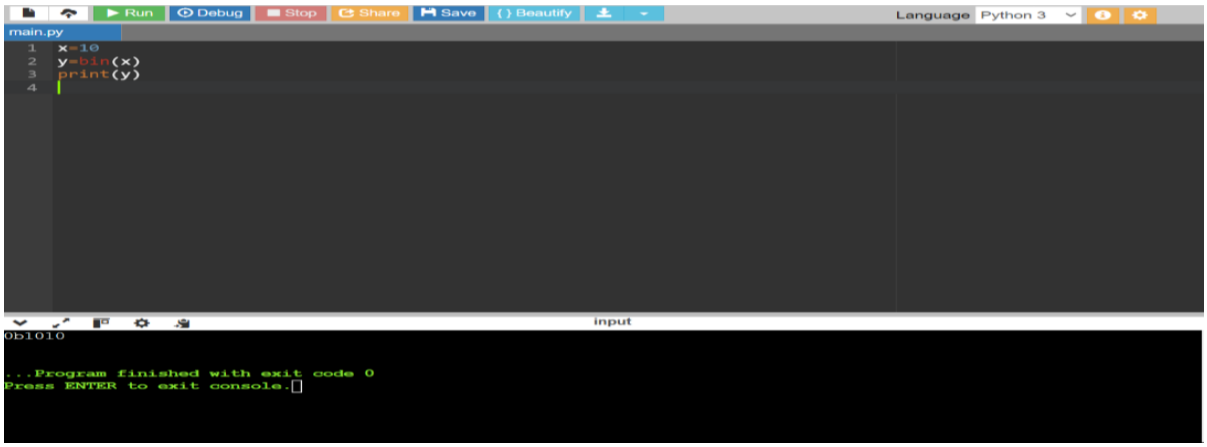
12. Python abs() Function Example



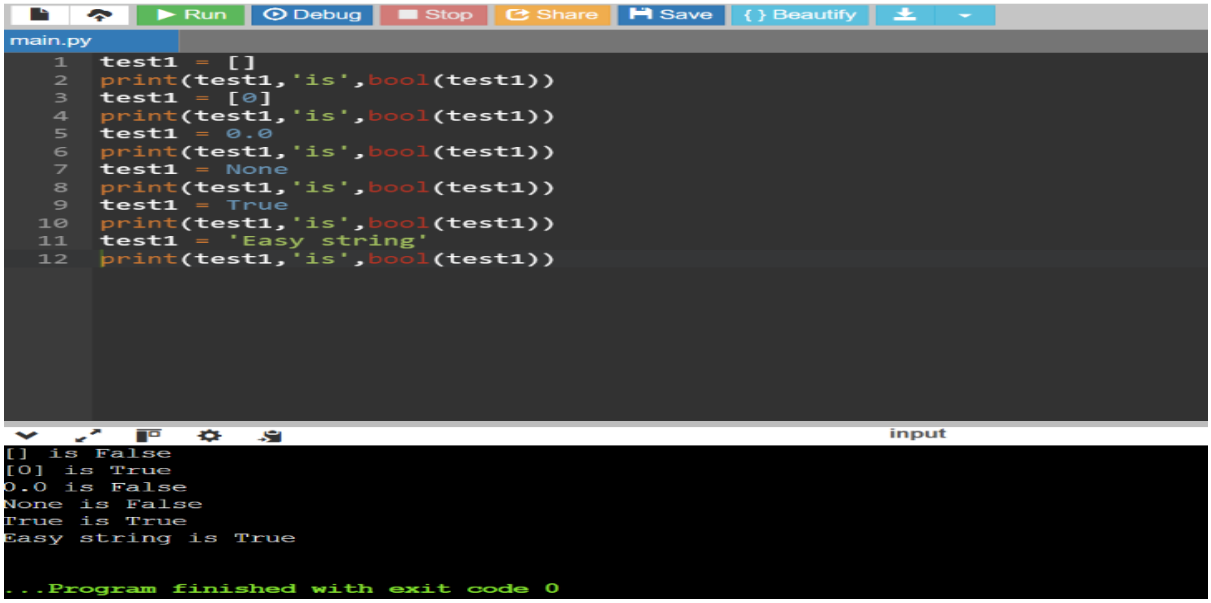
13. Python all() Function Example



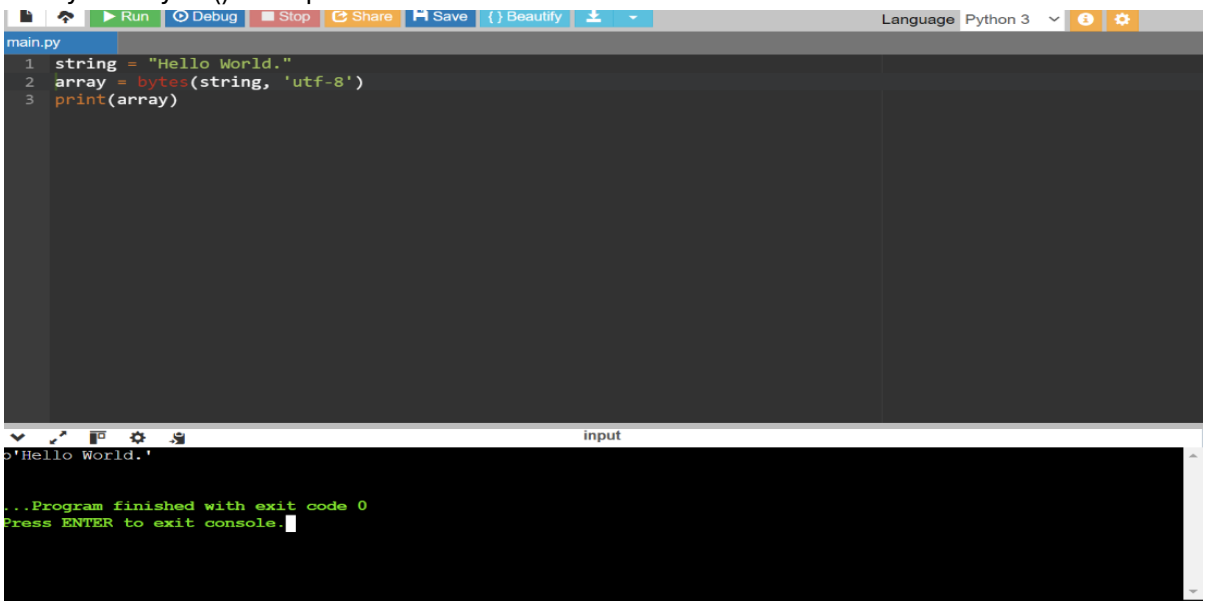
14. Python bin() Function Example



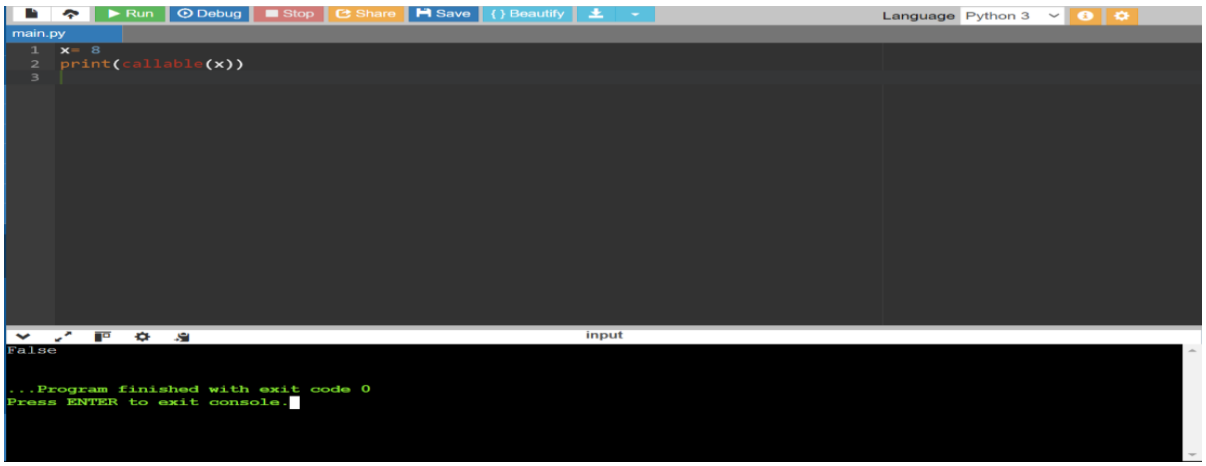
15. Python bool() Example



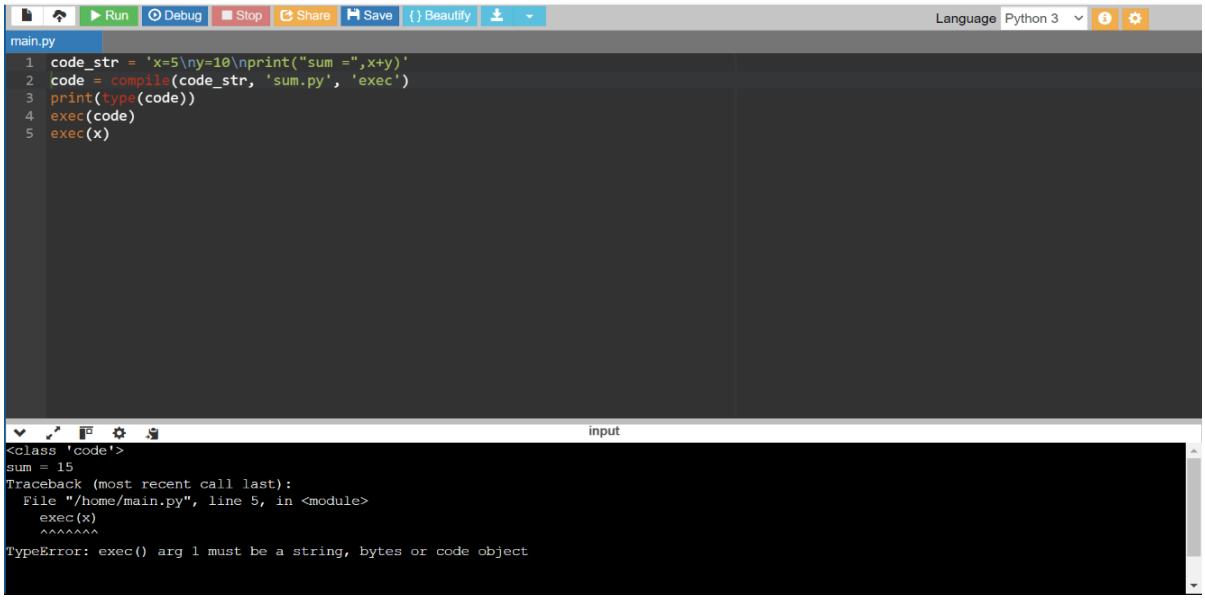
16. Python bytes() Example



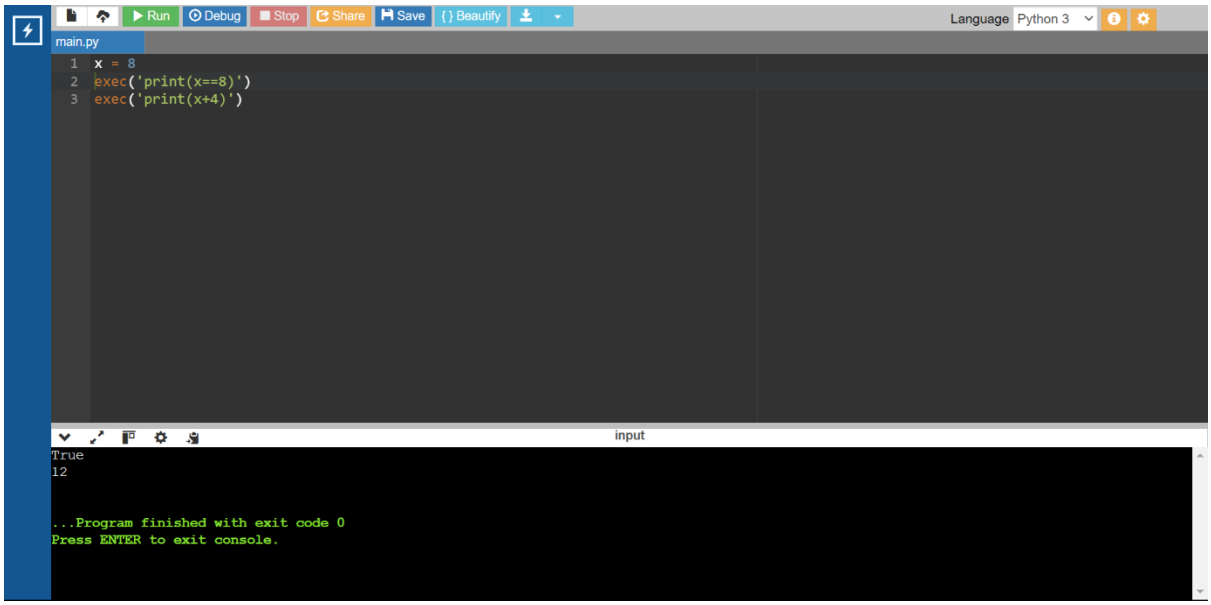
17. Python callable() Function Example



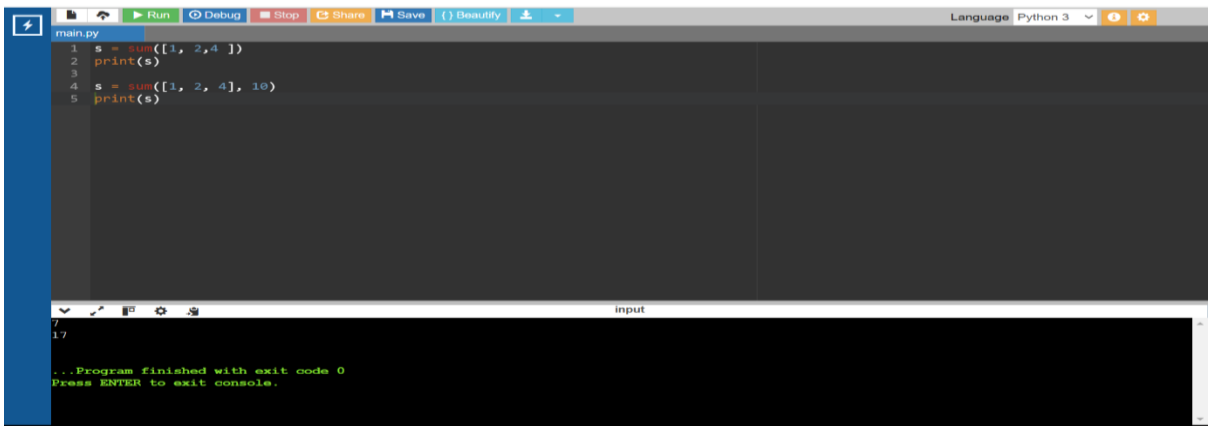
18. Python compile() Function Example



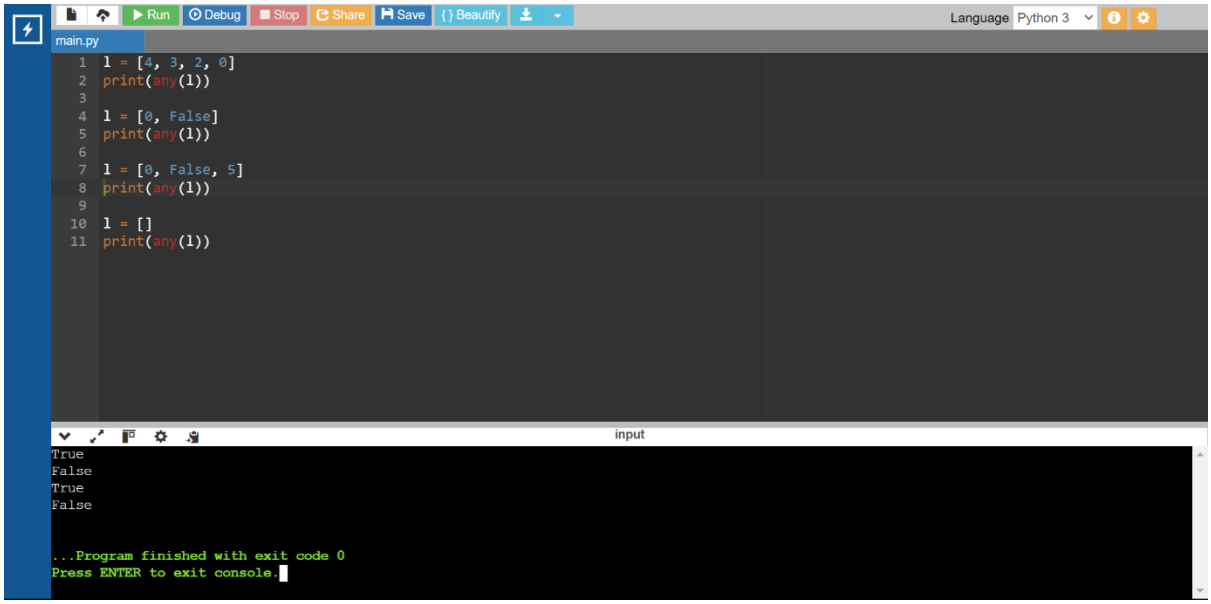
19. Python exec() Function Example



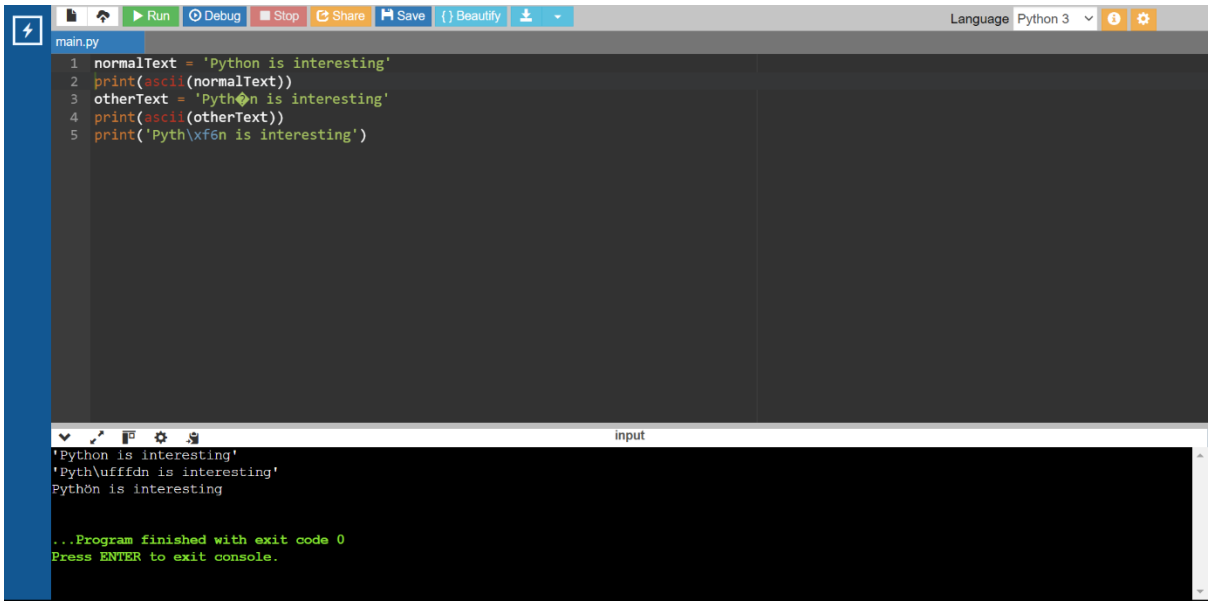
20. Python sum() Function Example



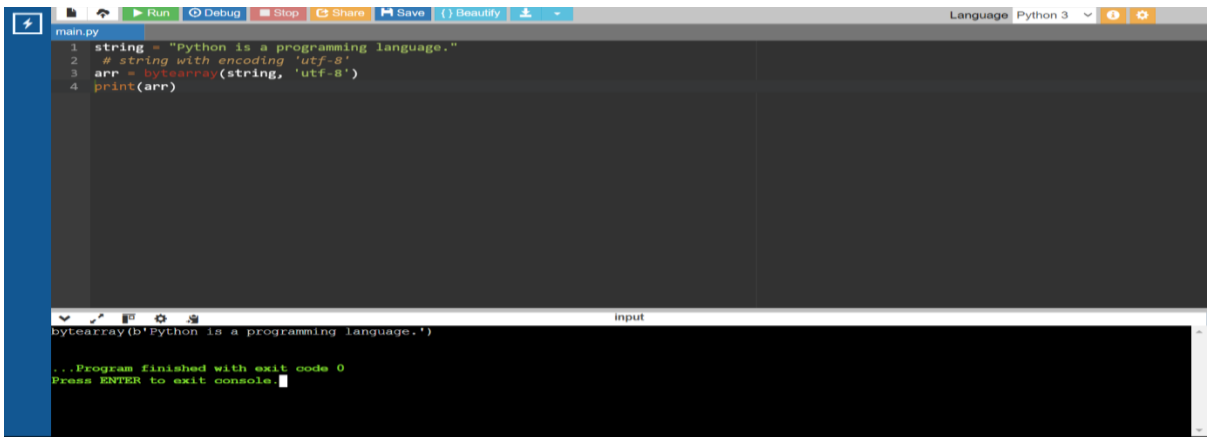
21. Python any() Function Example



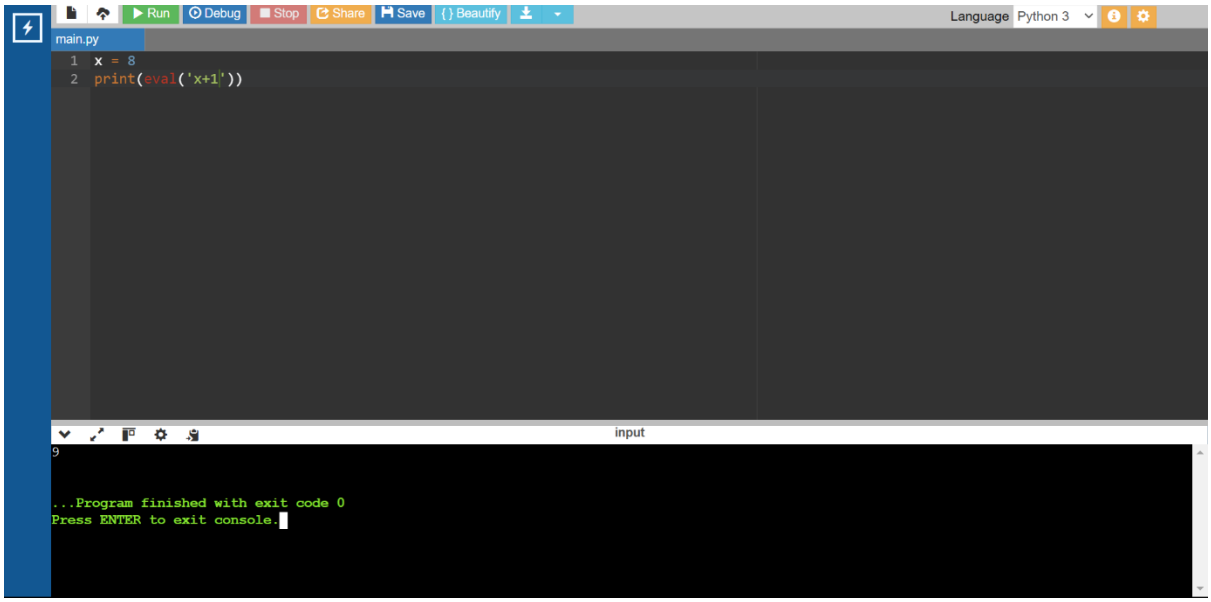
22. Python ascii() Function Example



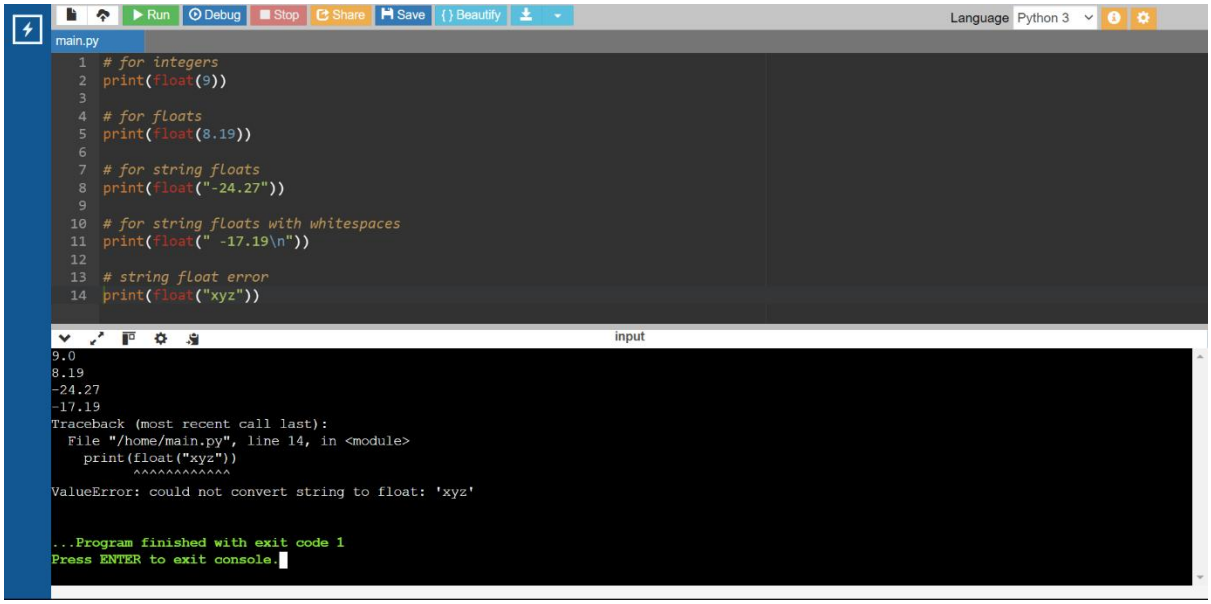
23. Python bytearray() Example



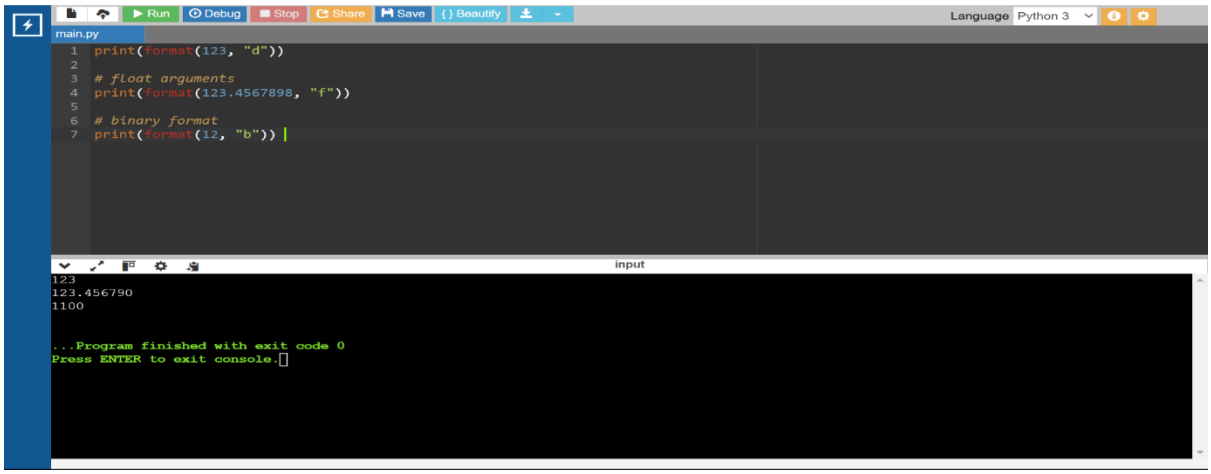
24. Python eval() Function Example



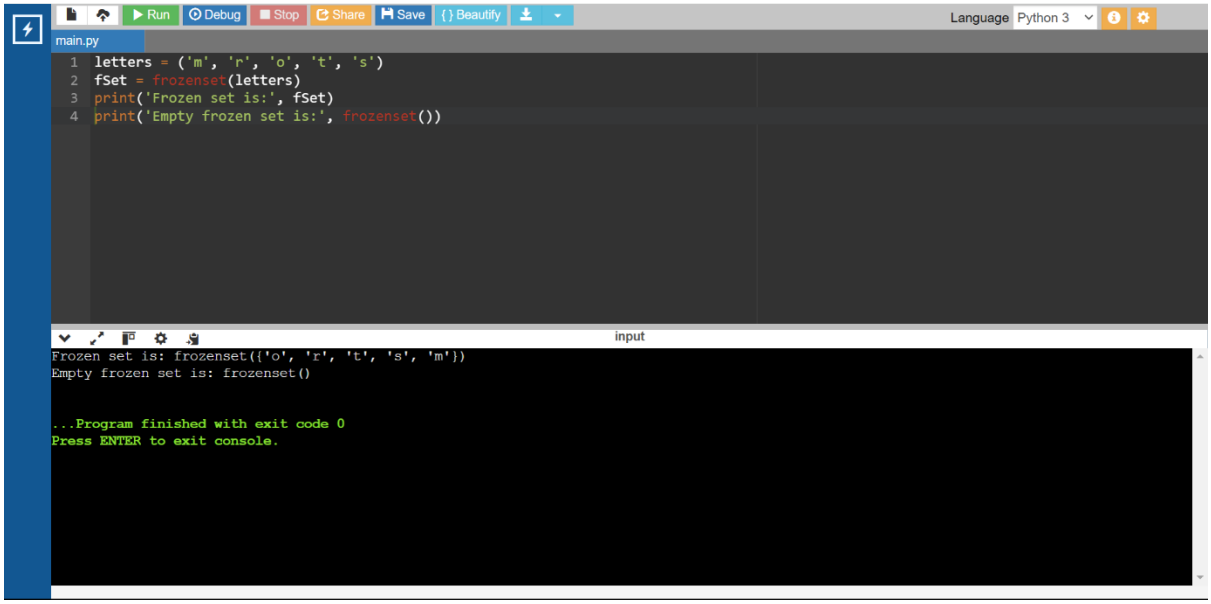
25. Python float() Example



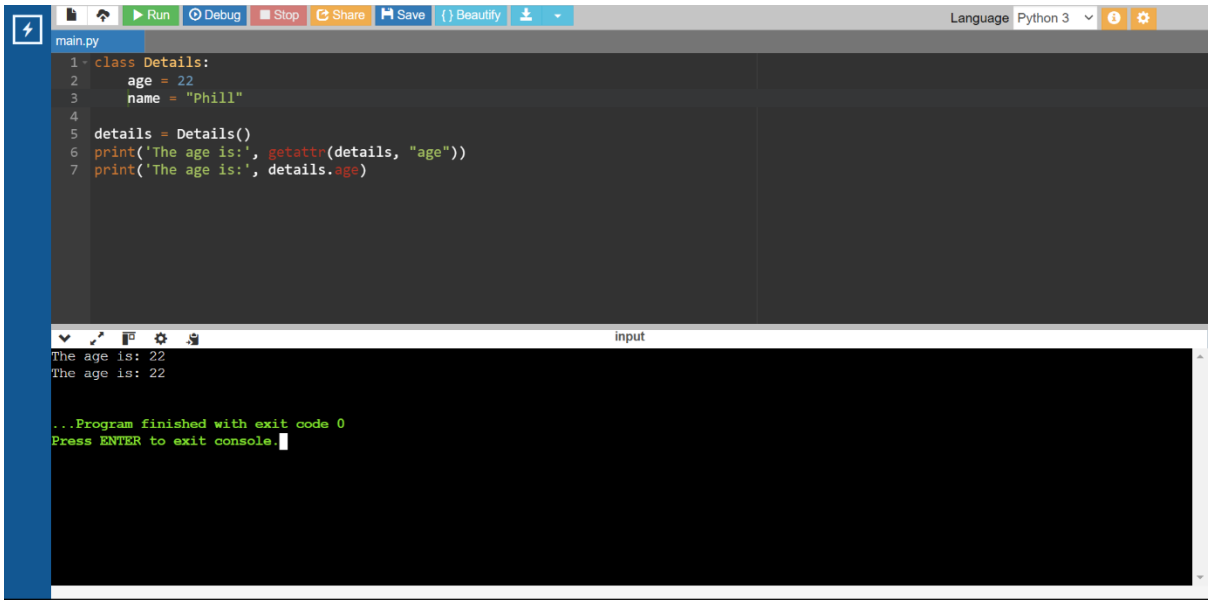
26. Python format() Function



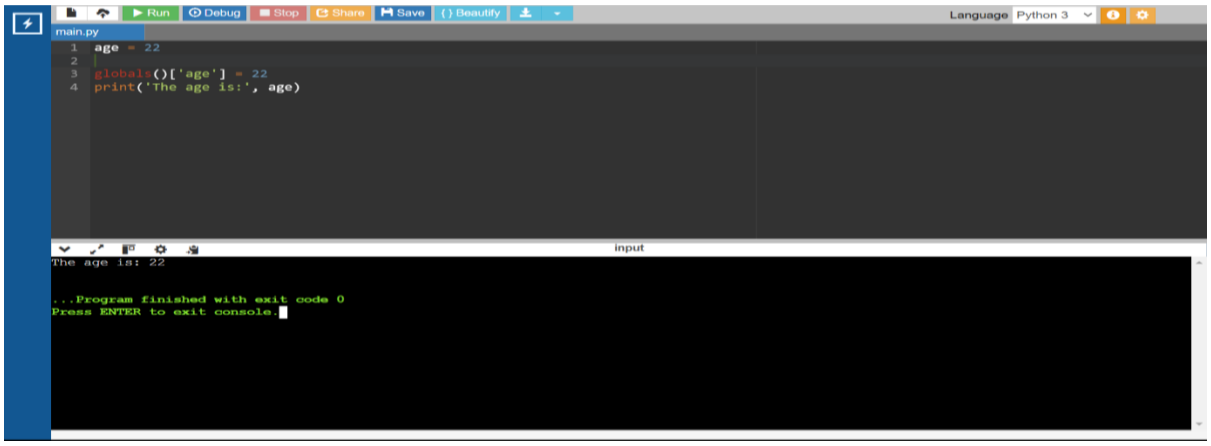
27. Python frozenset() Example



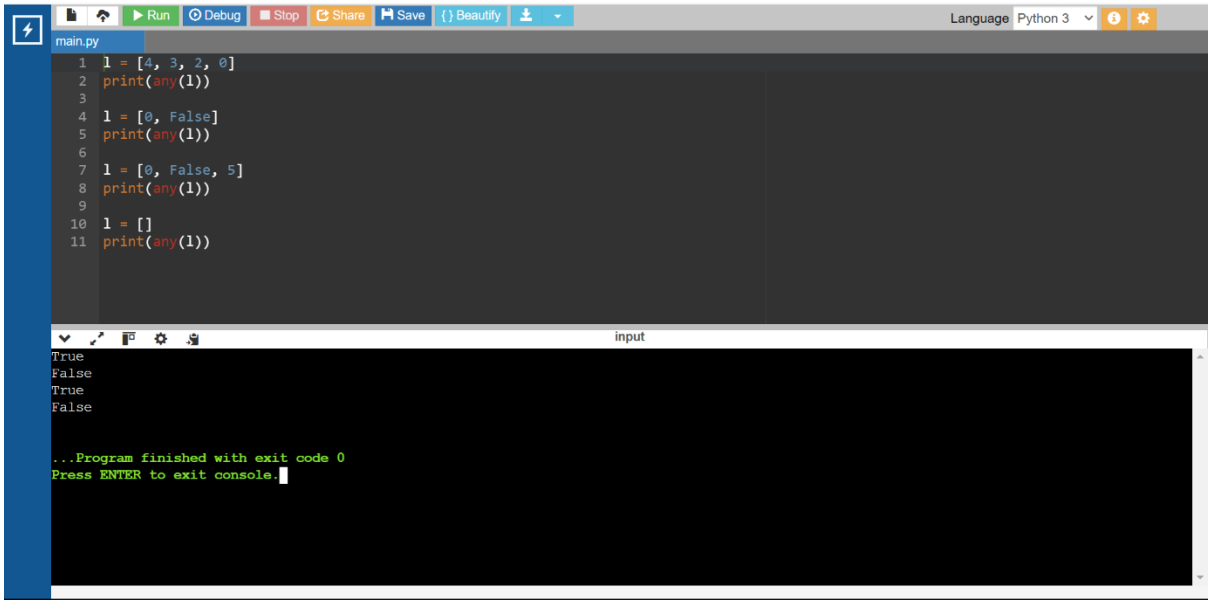
28. Python getattr() Function Example



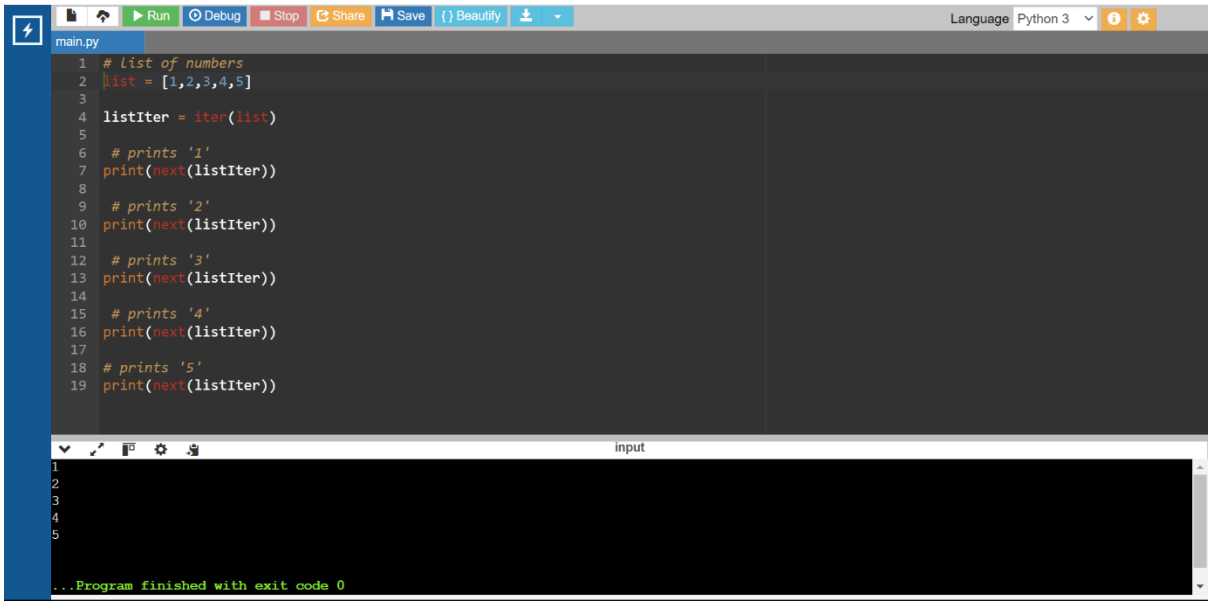
29. Python globals() Function Example



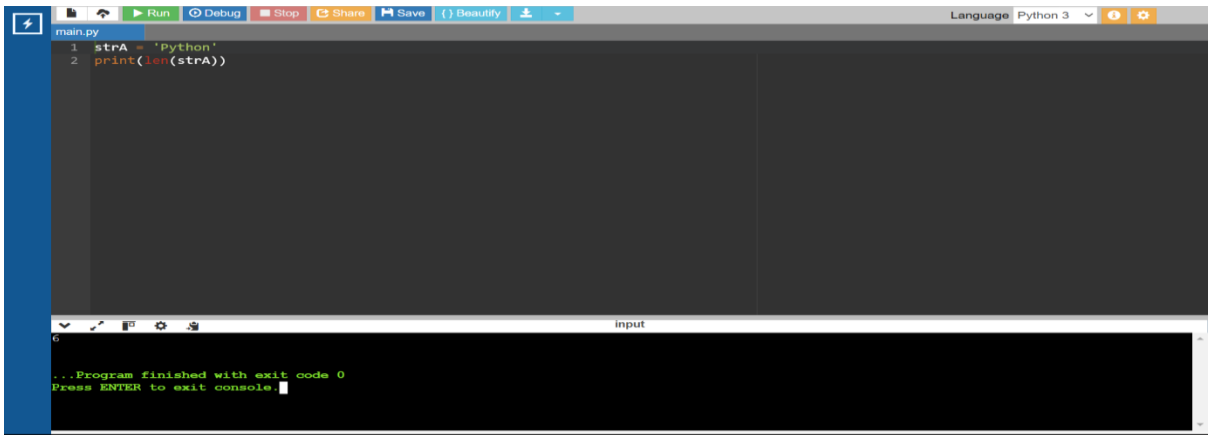
30. Python hasattr() Function Example



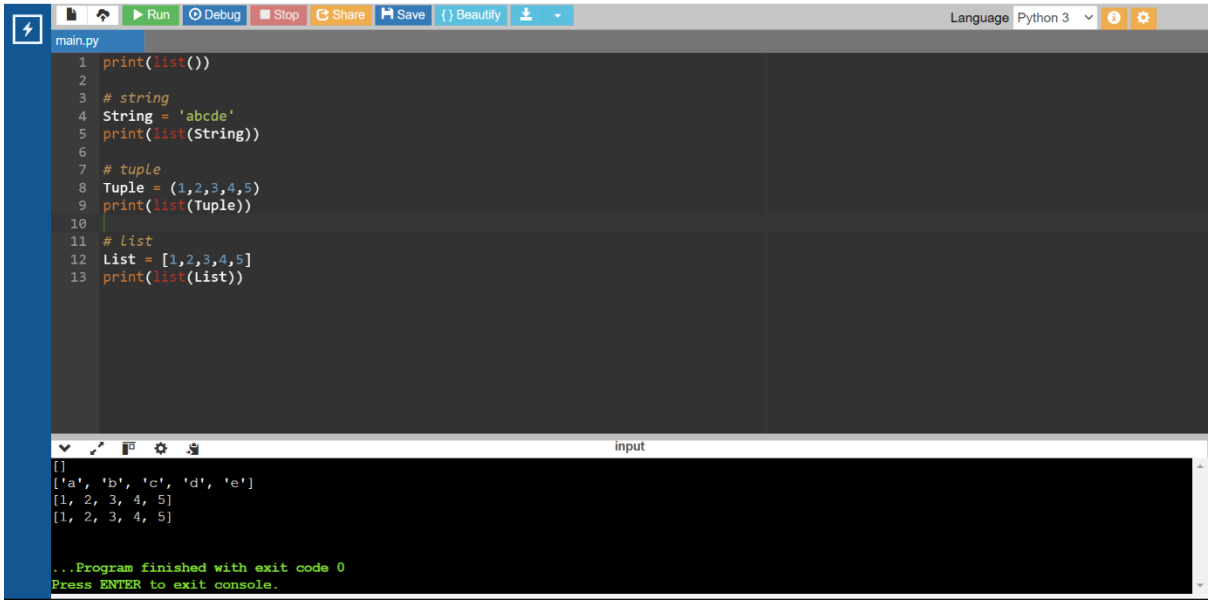
31. Python iter() Function Example



32. Python len() Function Example

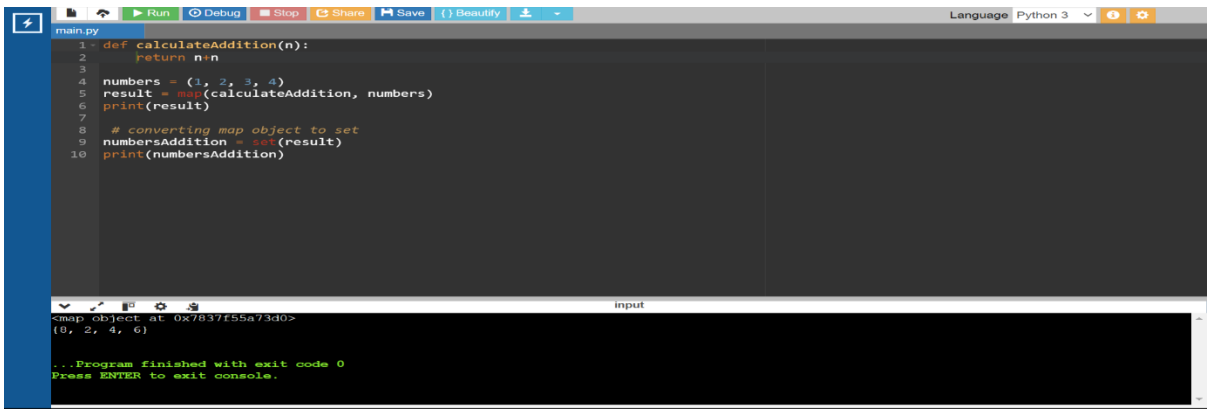


33. Python list() Example

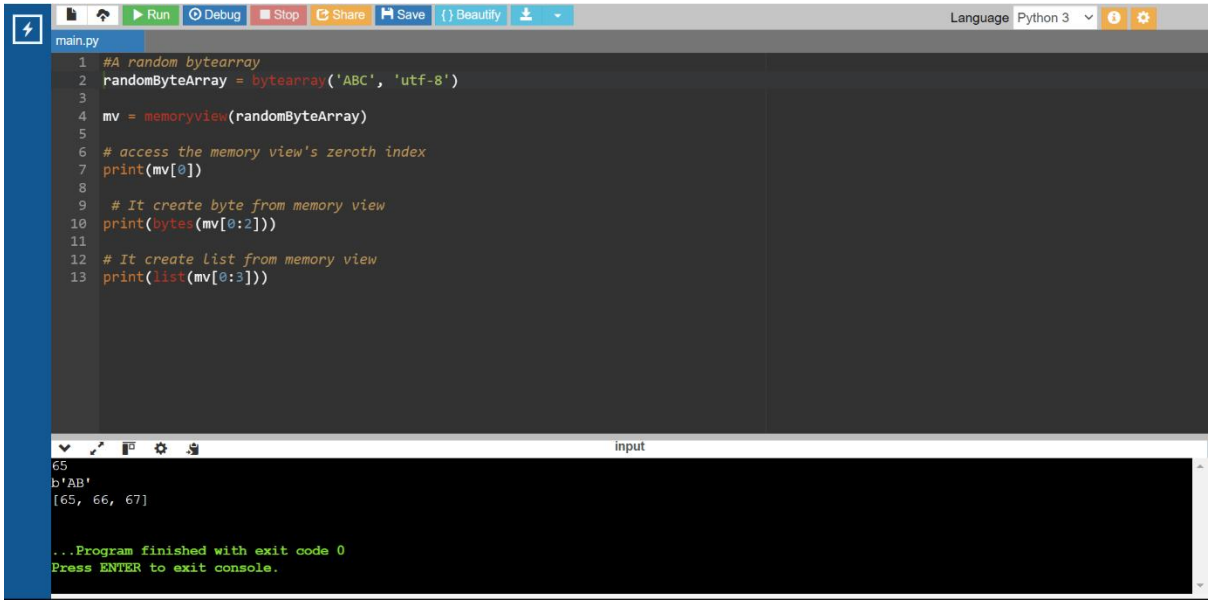


34. Python locals() Function Example

35. Python map() Function Example



36. Python memoryview () Function Example



```
#A random bytearray
randomByteArray = bytearray('ABC', 'utf-8')

mv = memoryview(randomByteArray)

# access the memory view's zeroth index
print(mv[0])

 # It create byte from memory view
print(bytes(mv[0:2]))

# It create list from memory view
print(list(mv[0:3]))
```

```
65
b'AB'
[65, 66, 67]


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

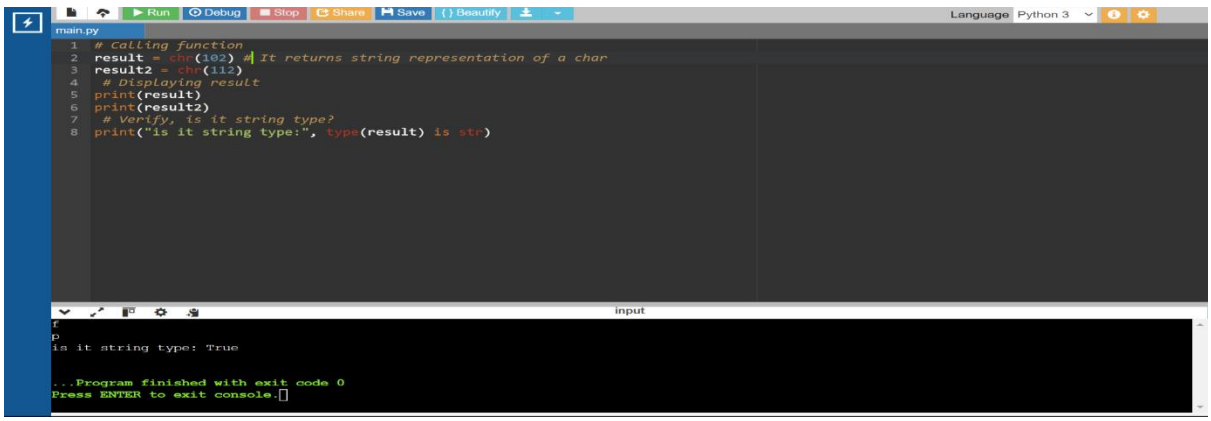
37. Python object() Example

```
python = object()

print(type(python))
print(dir(python))
```

```
<class 'object'>
['__class__', '__delattr__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getstate__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__le__', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__sizeof__', '__str__', '__subclasshook__']


...Program finished with exit code 0
```

https://www.onlinegdb.com/online_python_compiler#tab-stdin

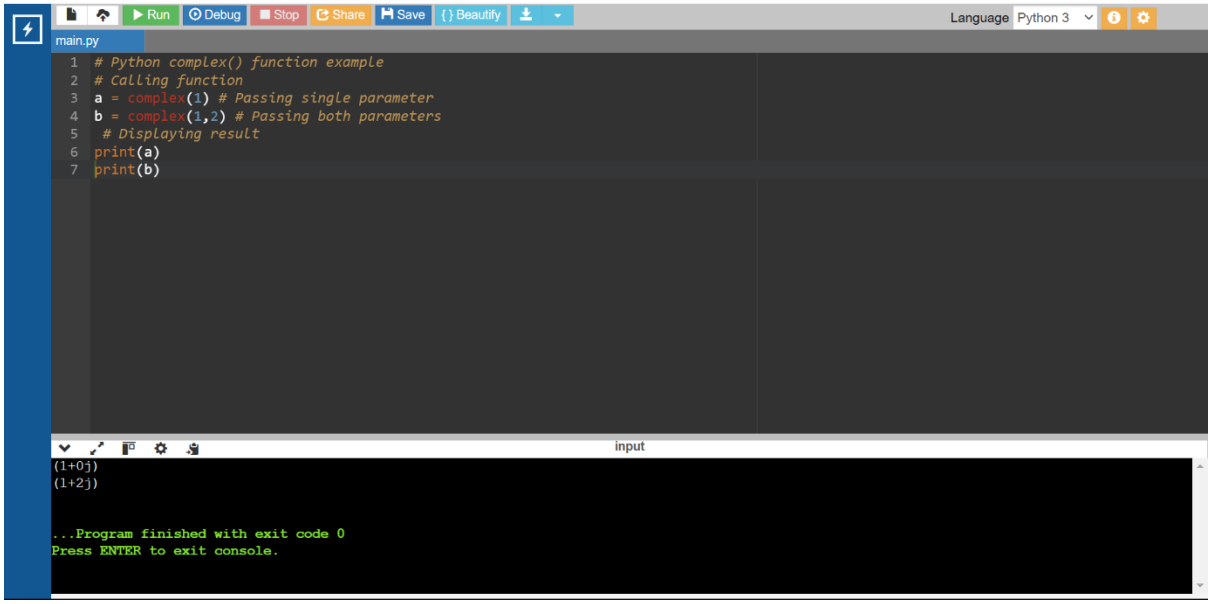
38. Python chr() Function Example



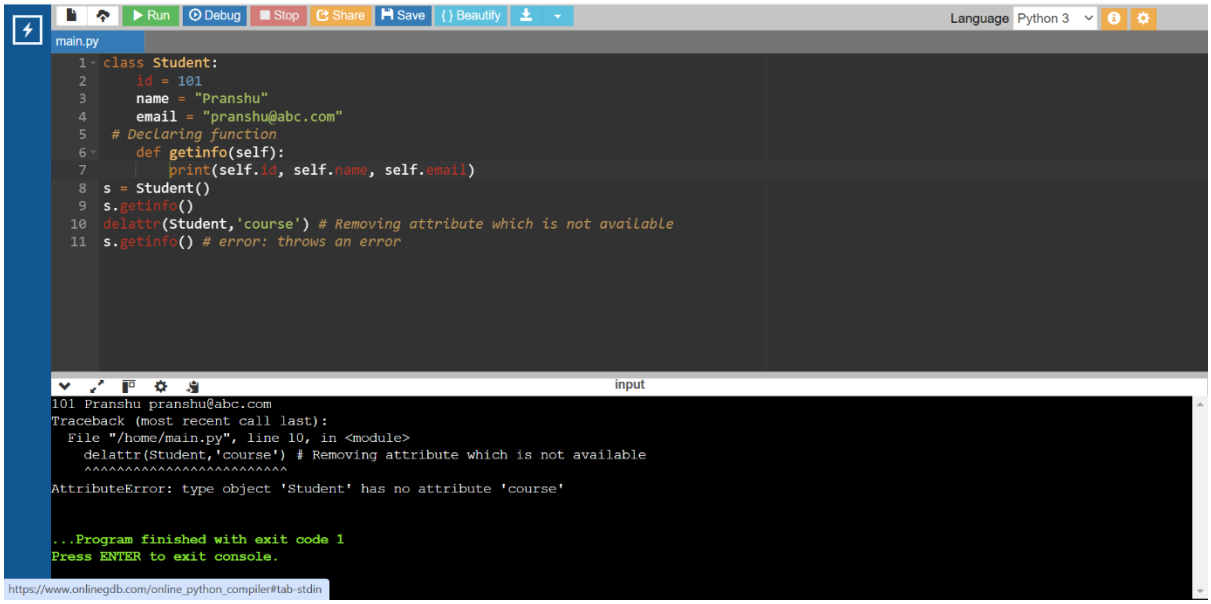
```
# Calling function
result = chr(102) # It returns string representation of a char
result2 = chr(112)
 # Displaying result
print(result)
print(result2)
 # Verify, is it string type?
print("is it string type:", type(result) is str)
```

```
f
p
is it string type: True


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

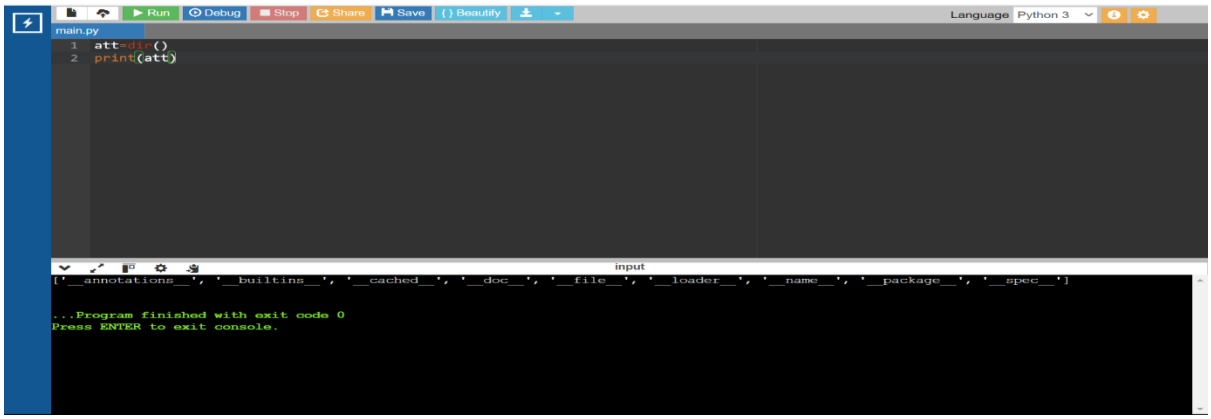
39. Python complex() Example



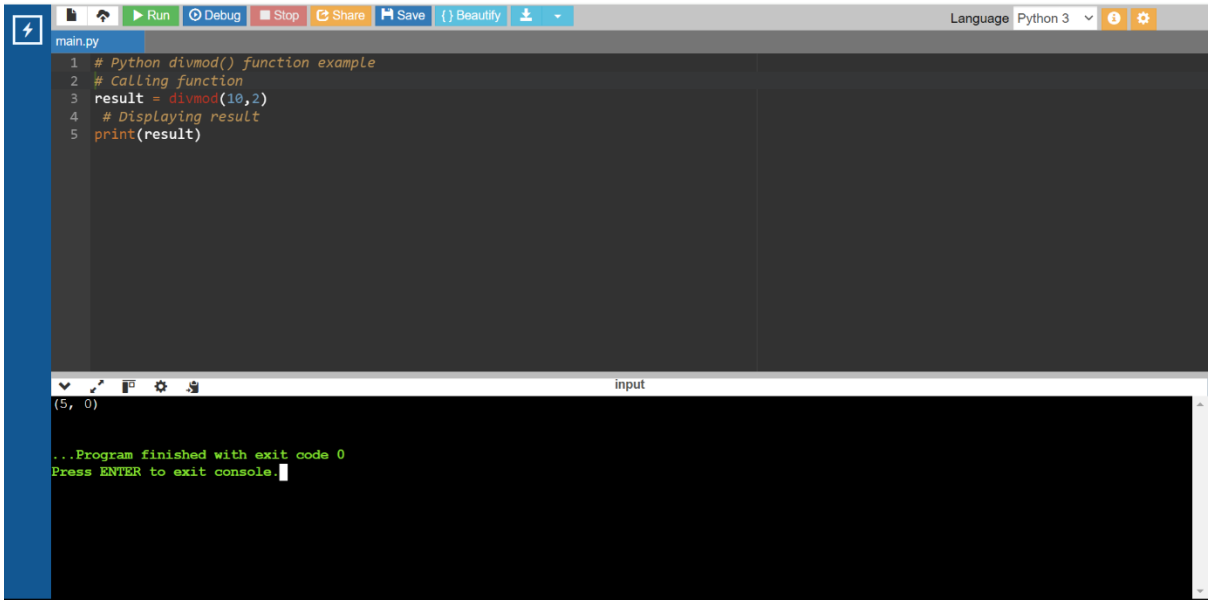
40. Python delattr() Function Example



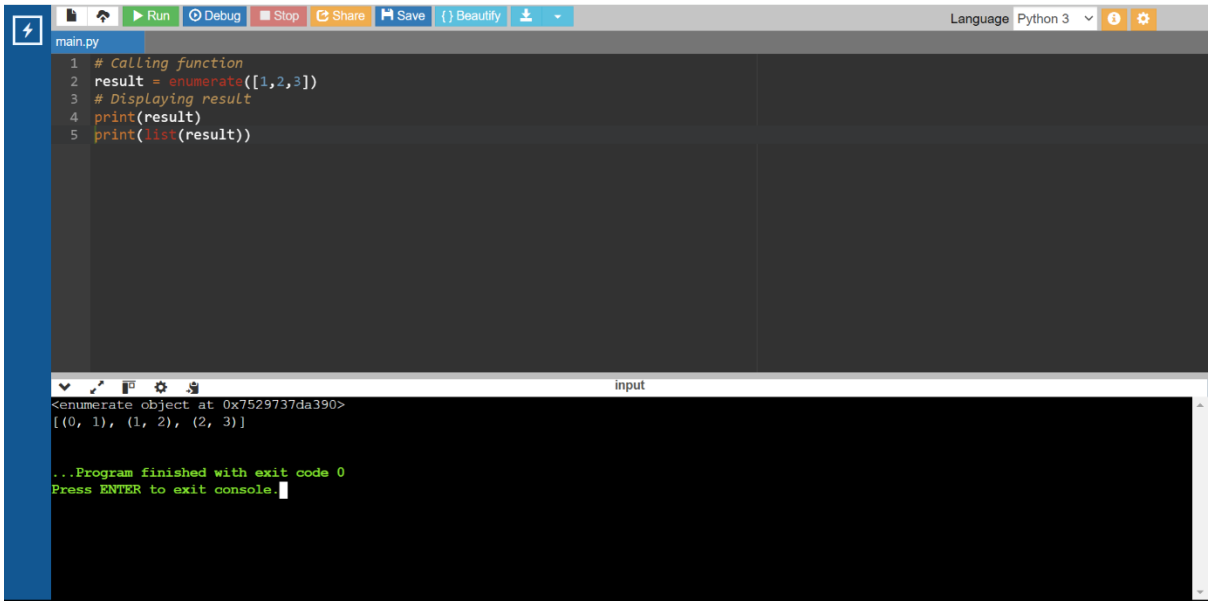
41. Python dir() Function Example



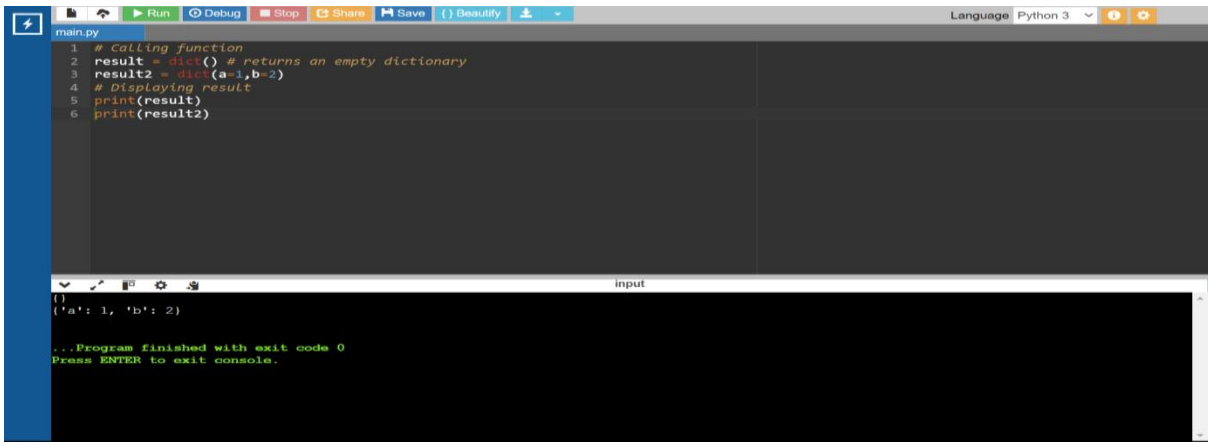
42. Python divmod() Function Example



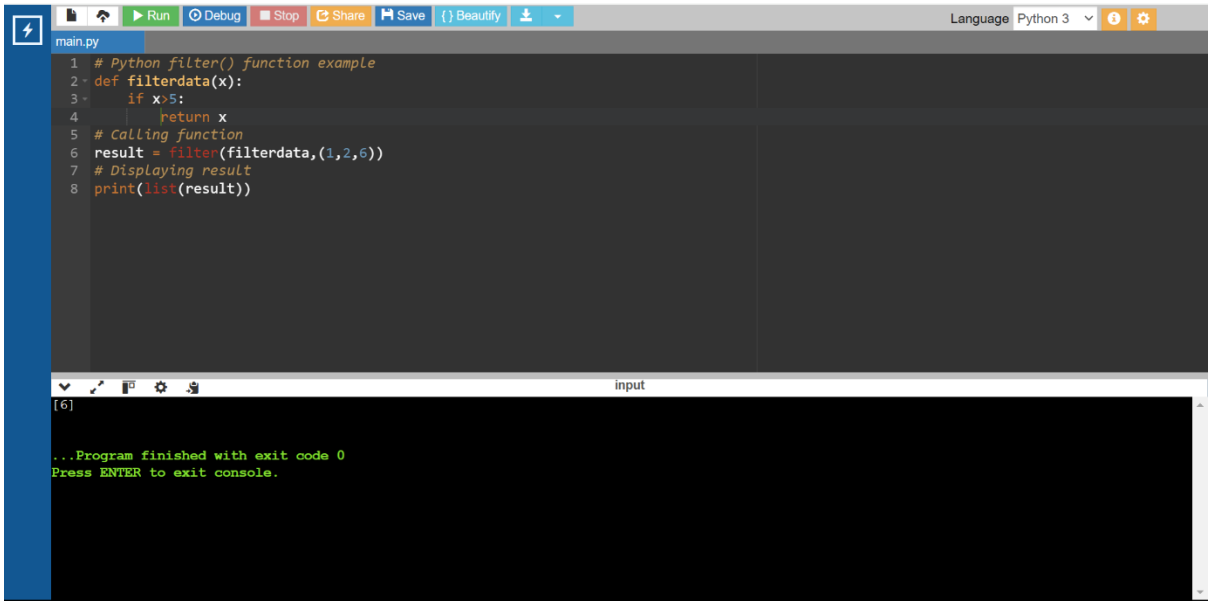
43. Python enumerate() Function Example



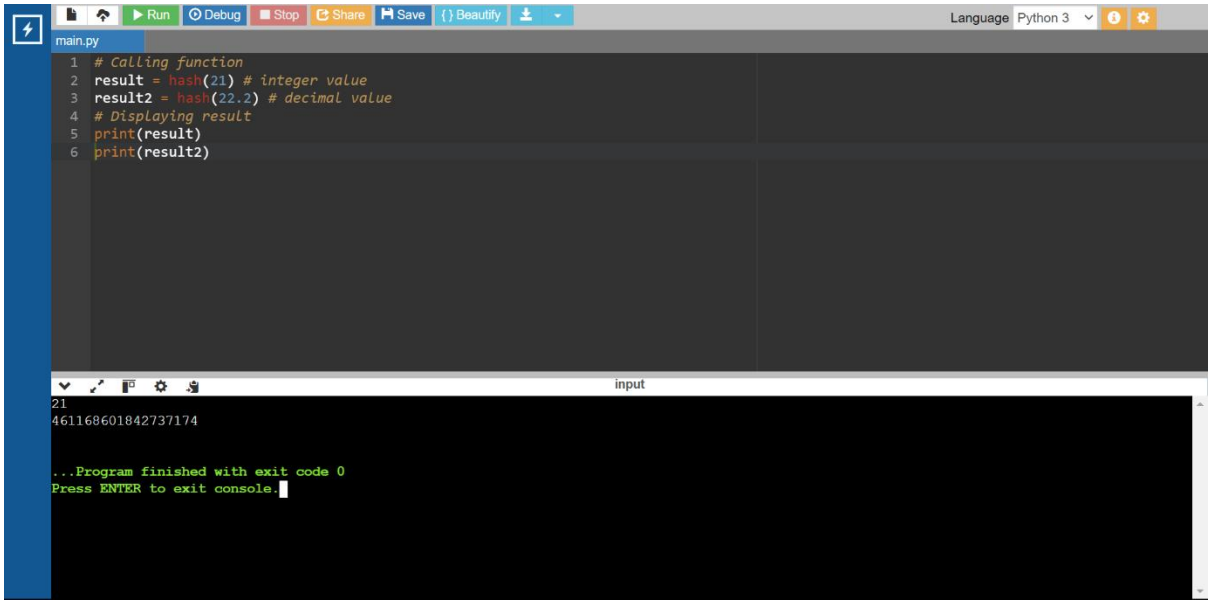
44. Python dict() Example



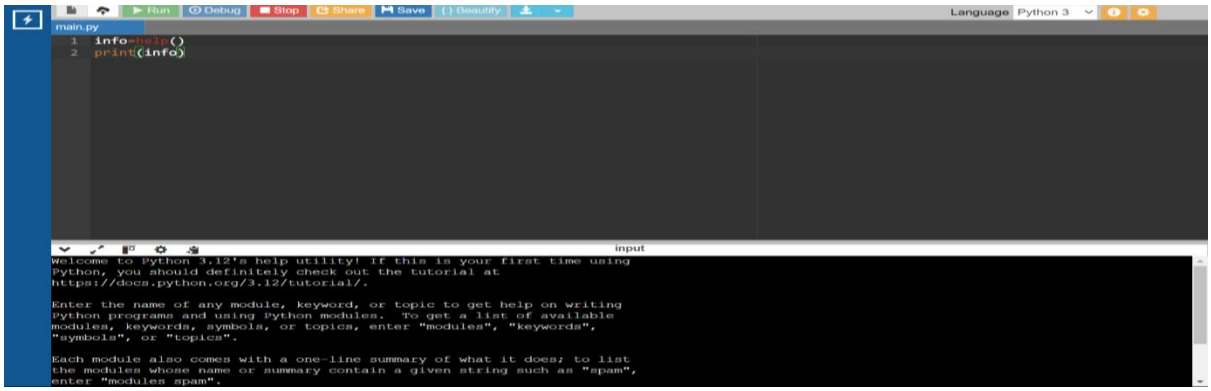
45. Python filter() Function Example



46. Python hash() Function Example

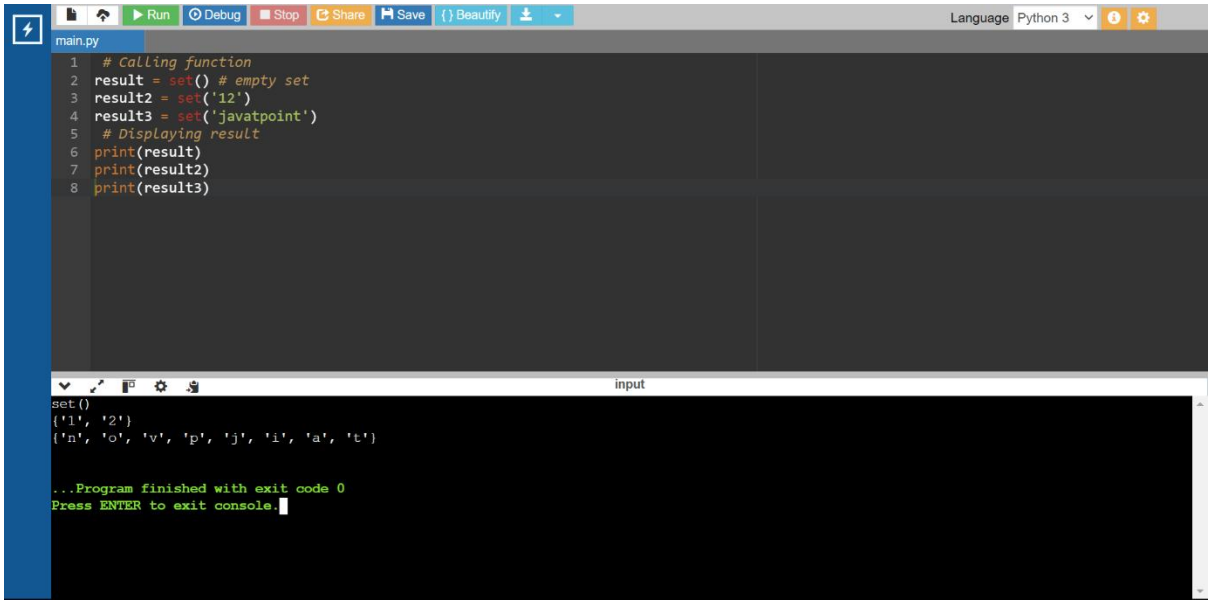


47. Python help() Function Example

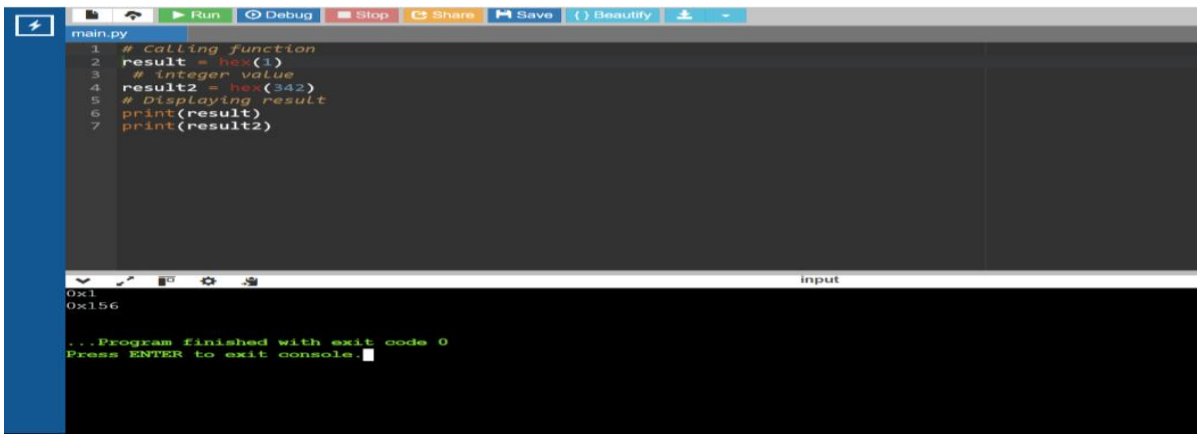


48. Python min() Function Example

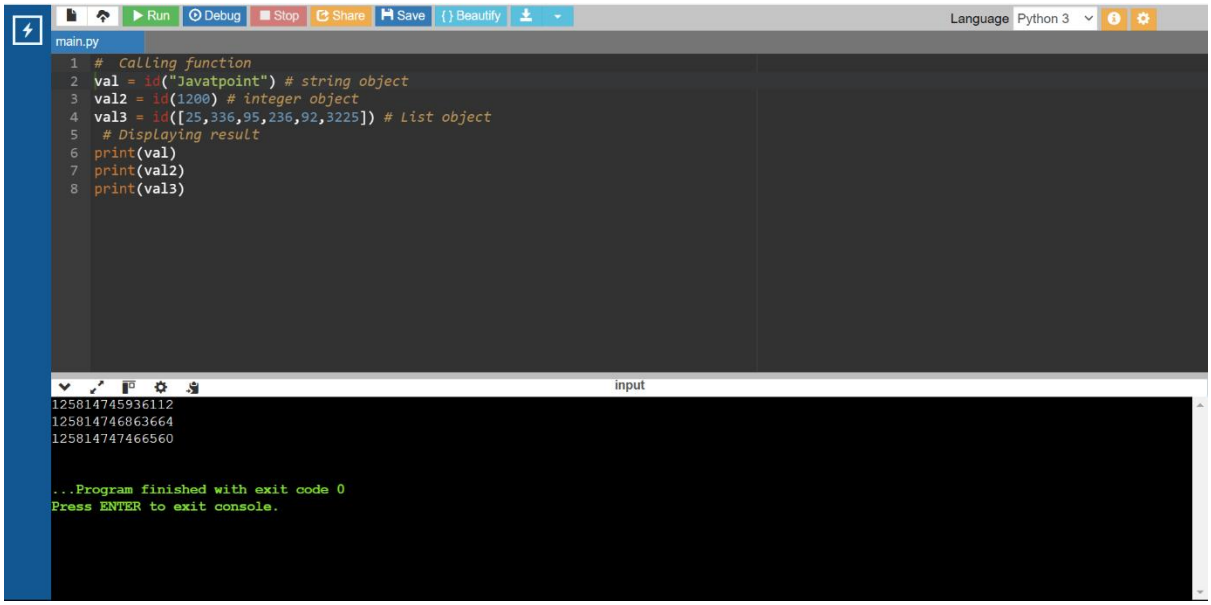
49. Python set() Function Example



50. Python hex() Function Example

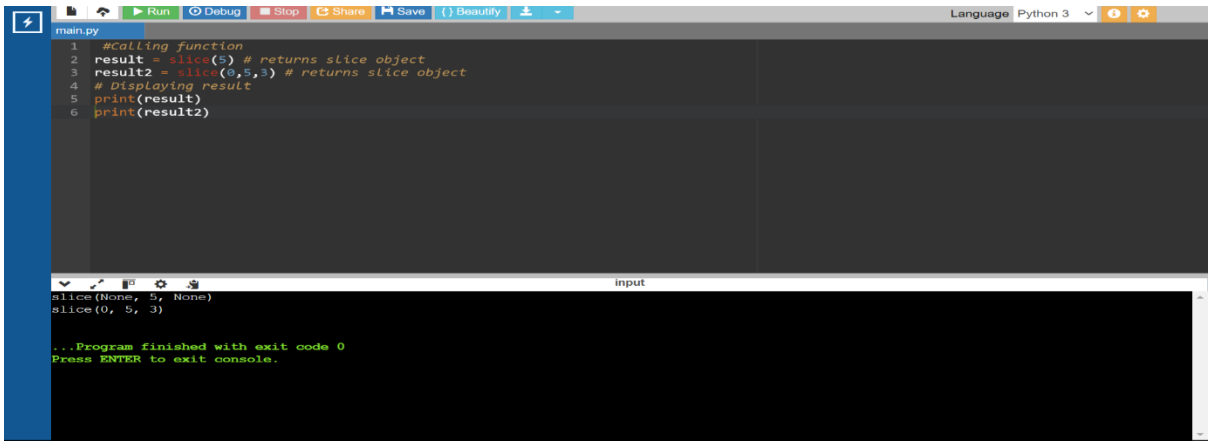


51. Python id() Function Example

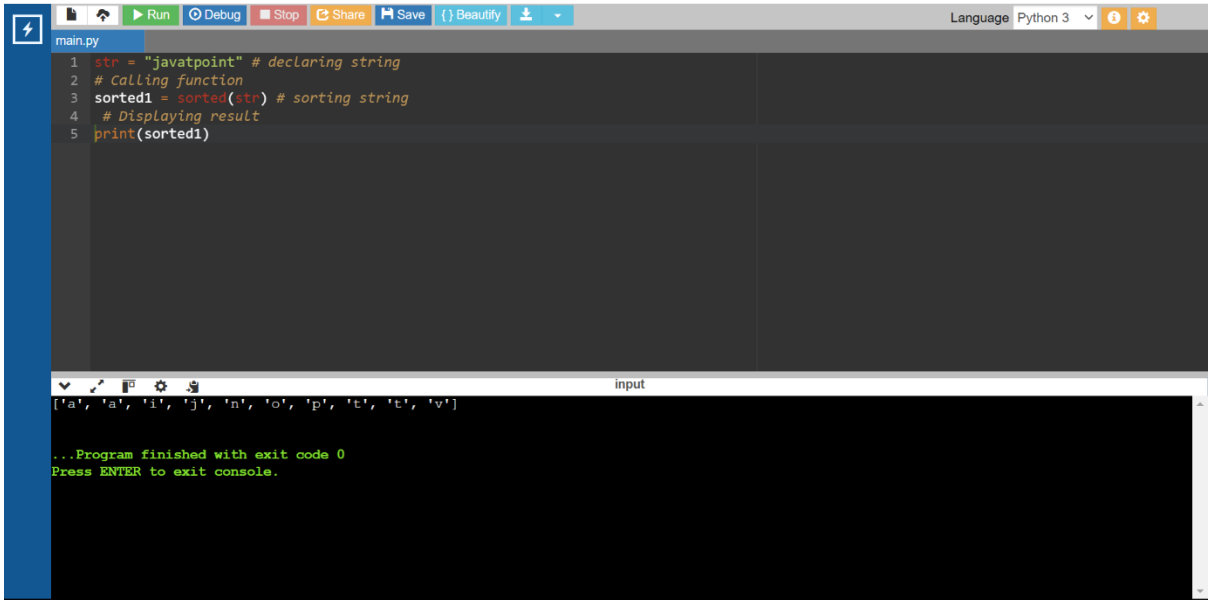


52. Python setattr() Function Example

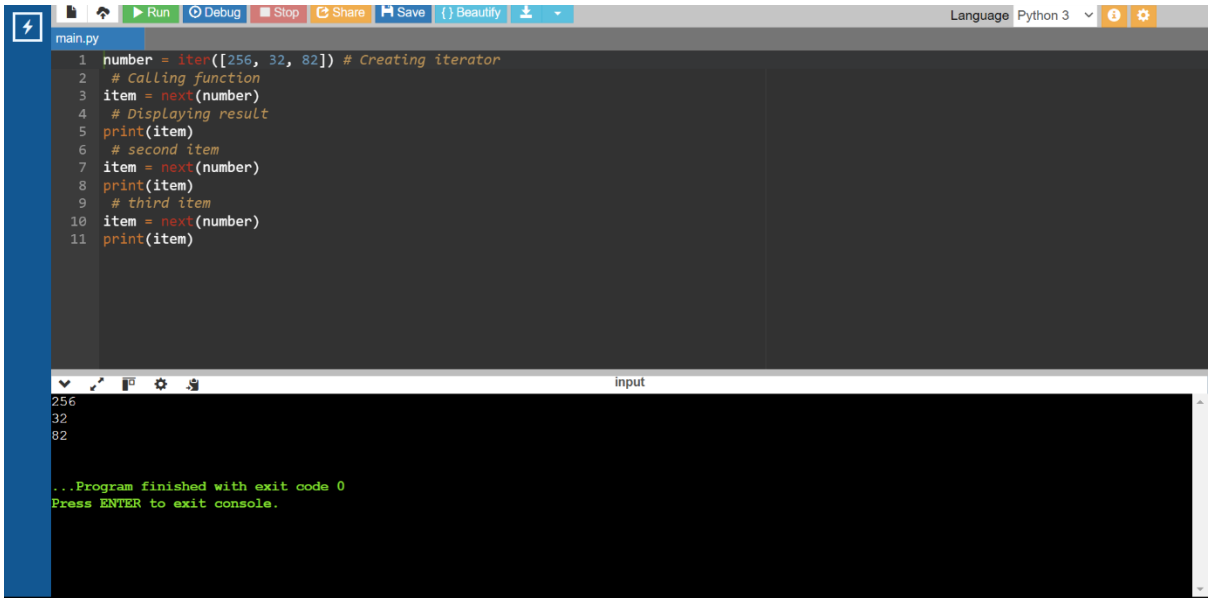
53. Python slice() Function Example



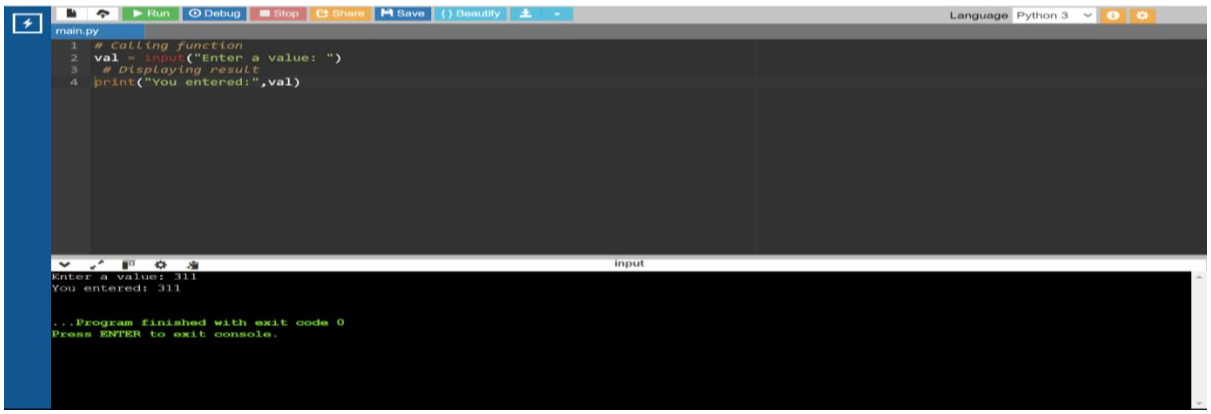
54. Python sorted() Function Example



55. Python next() Function Example



56. Python input() Function Example

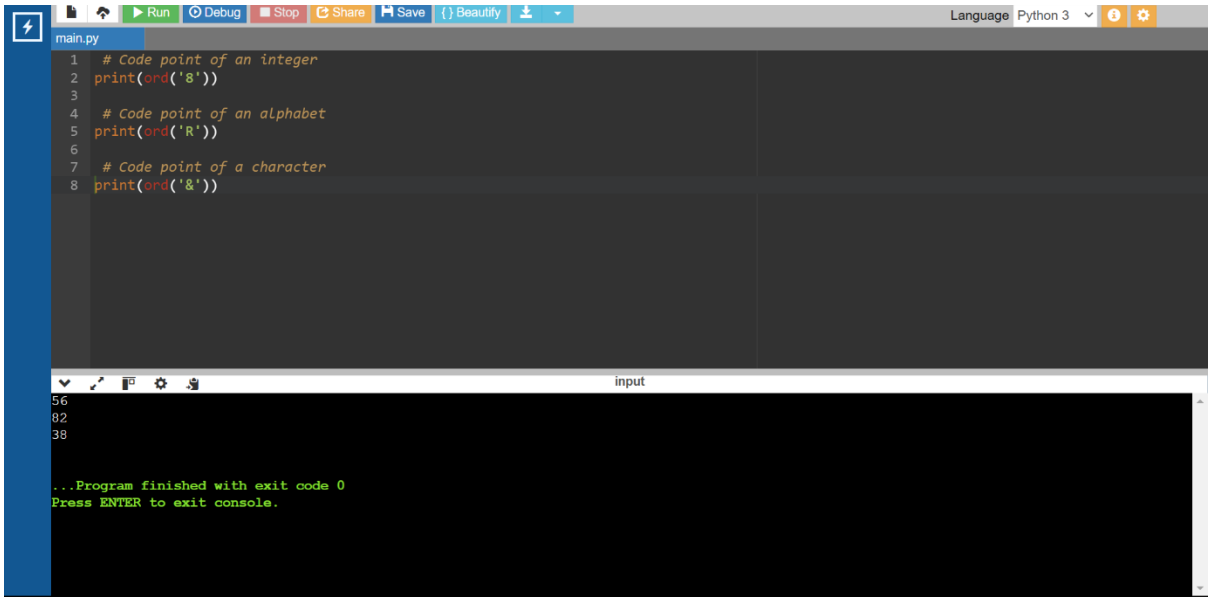


57. Python int() Function Example

58. Python isinstance() function Example

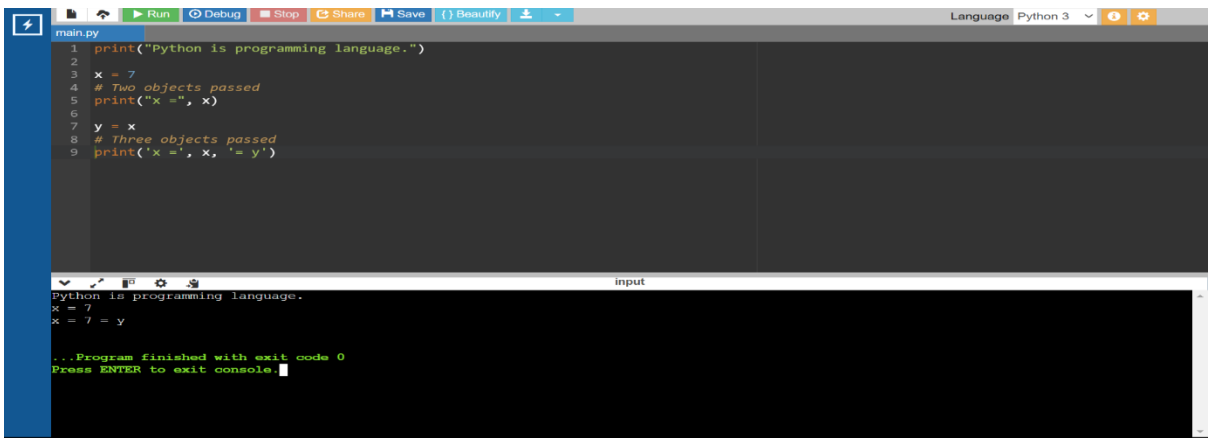
59. Python oct() function Example

60. Python ord() function Example

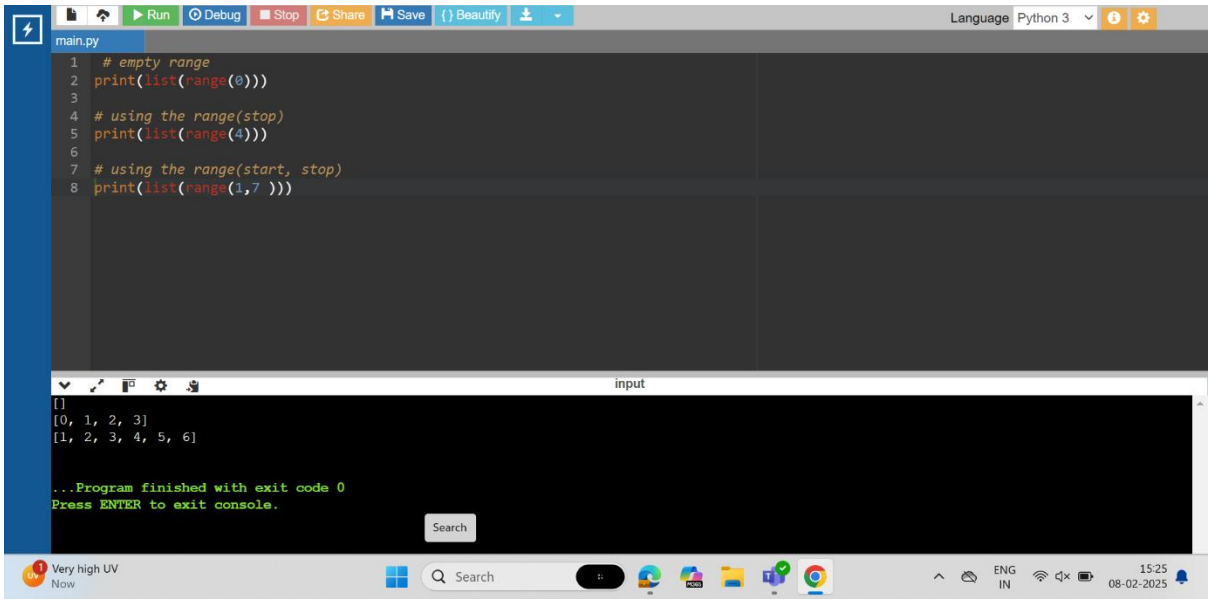


61. Python pow() function Example

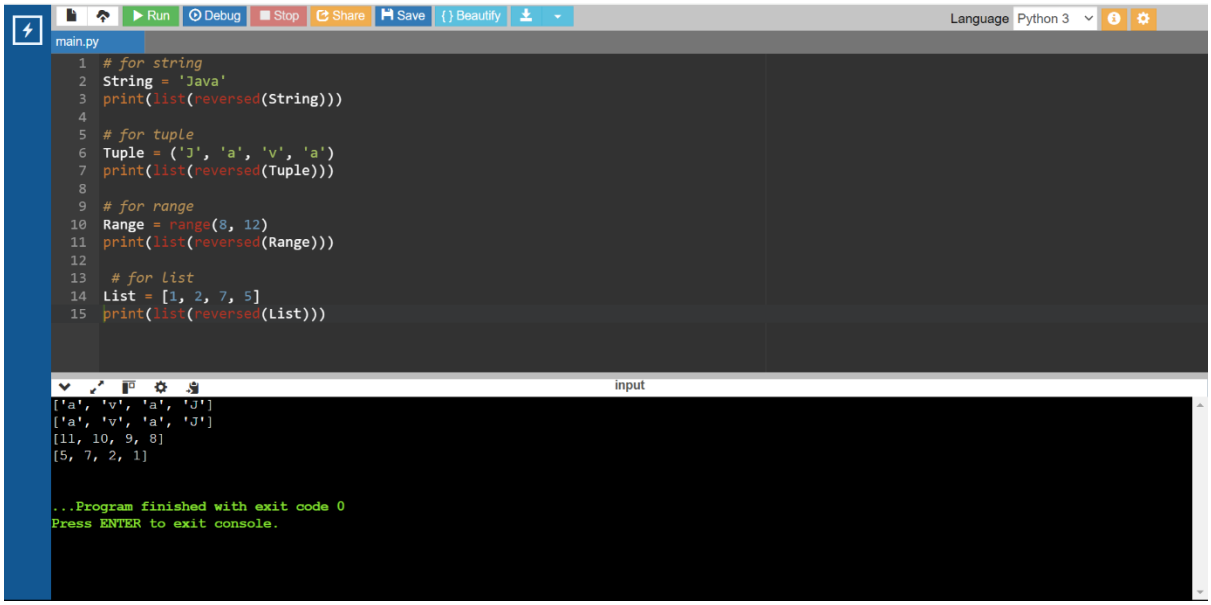
62. Python print() function Example



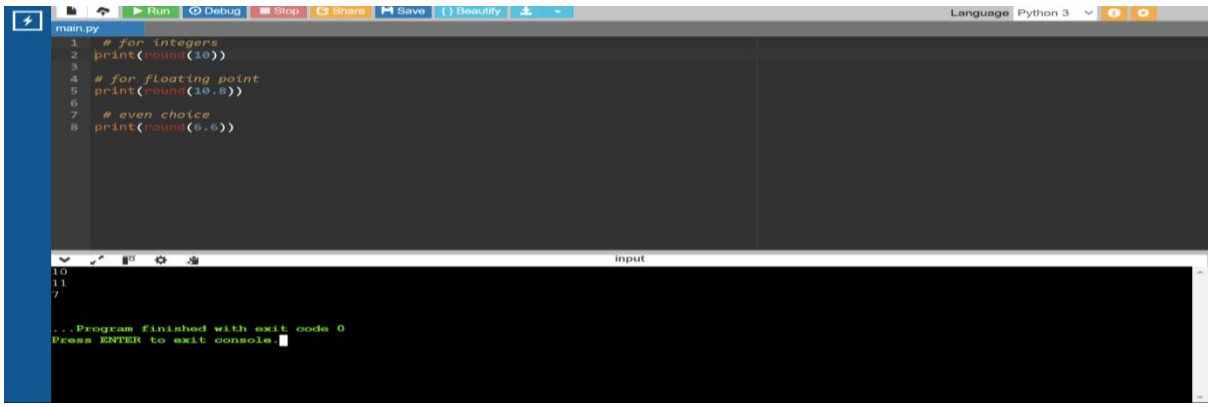
63. Python range() function Example



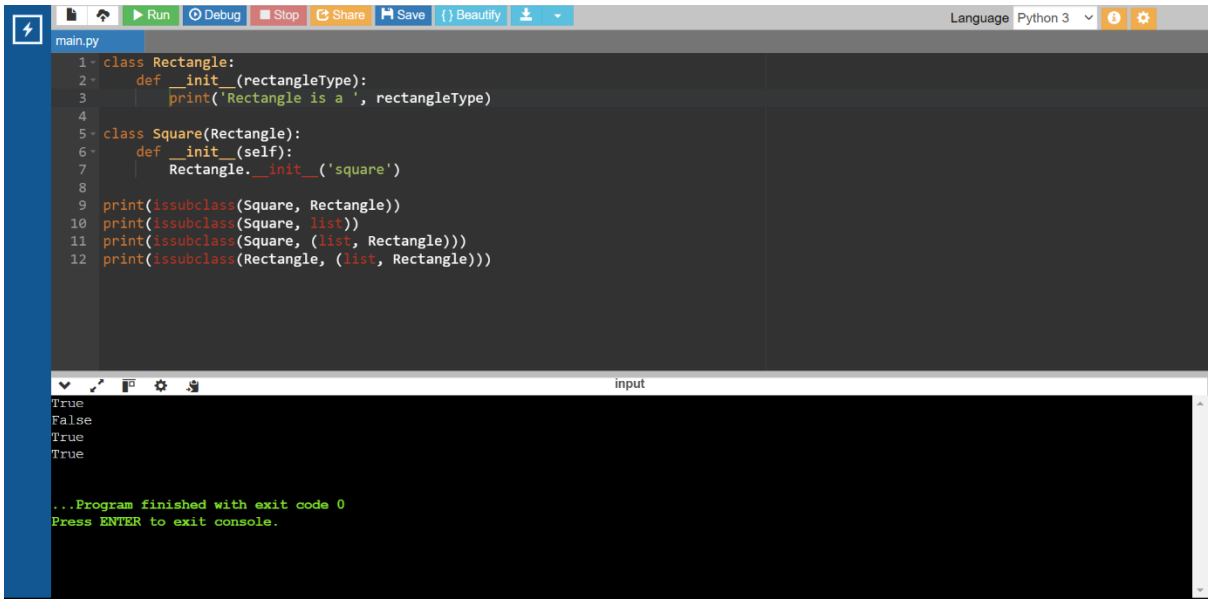
64. Python reversed() function Example



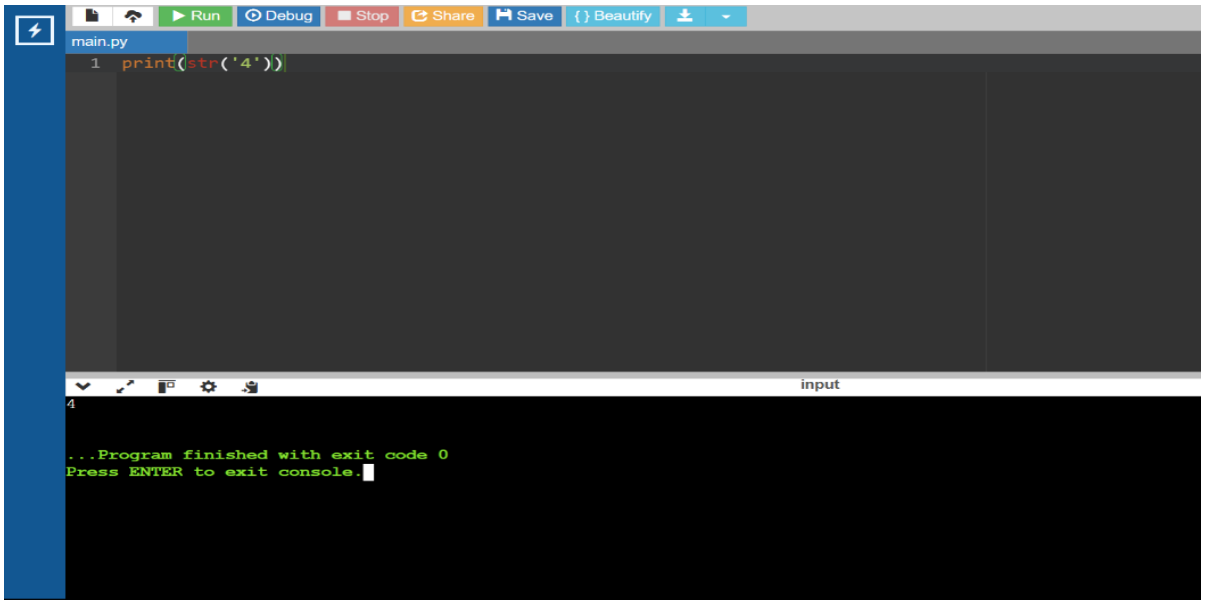
65. Python round() Function Example



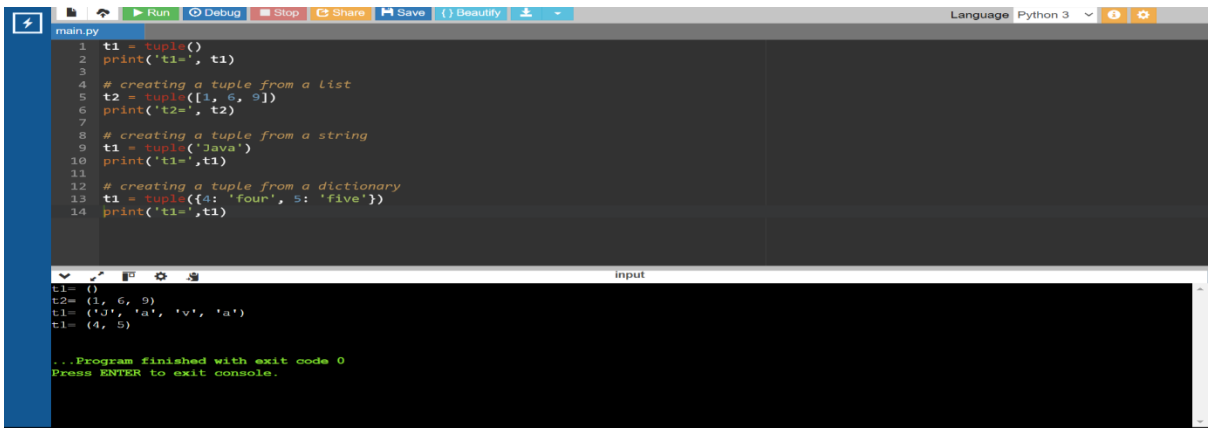
66. Python issubclass() Function Example



67. Python str() Function Example

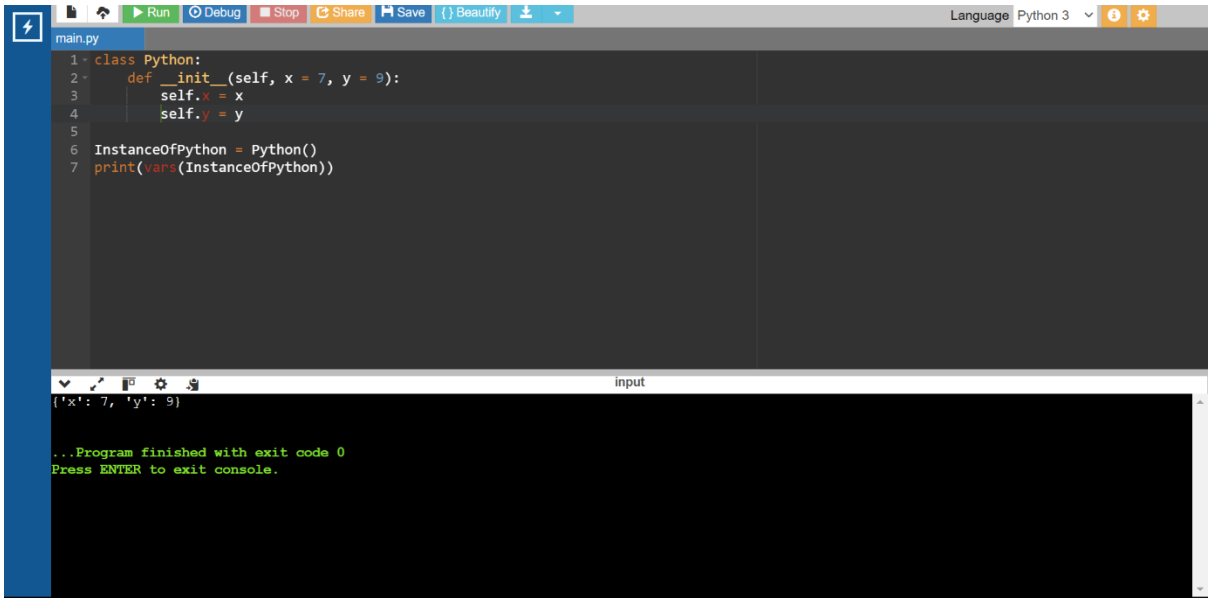


68. Python tuple() Function Example

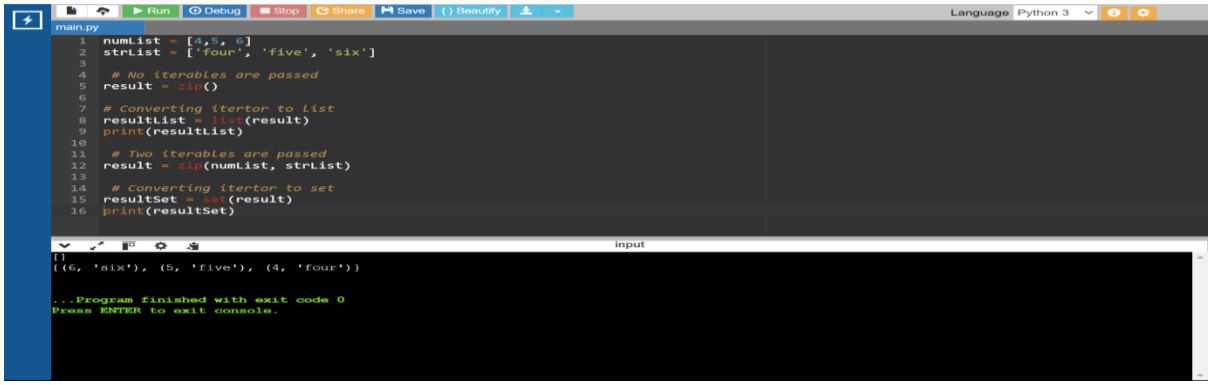


69. Python type() Function Example

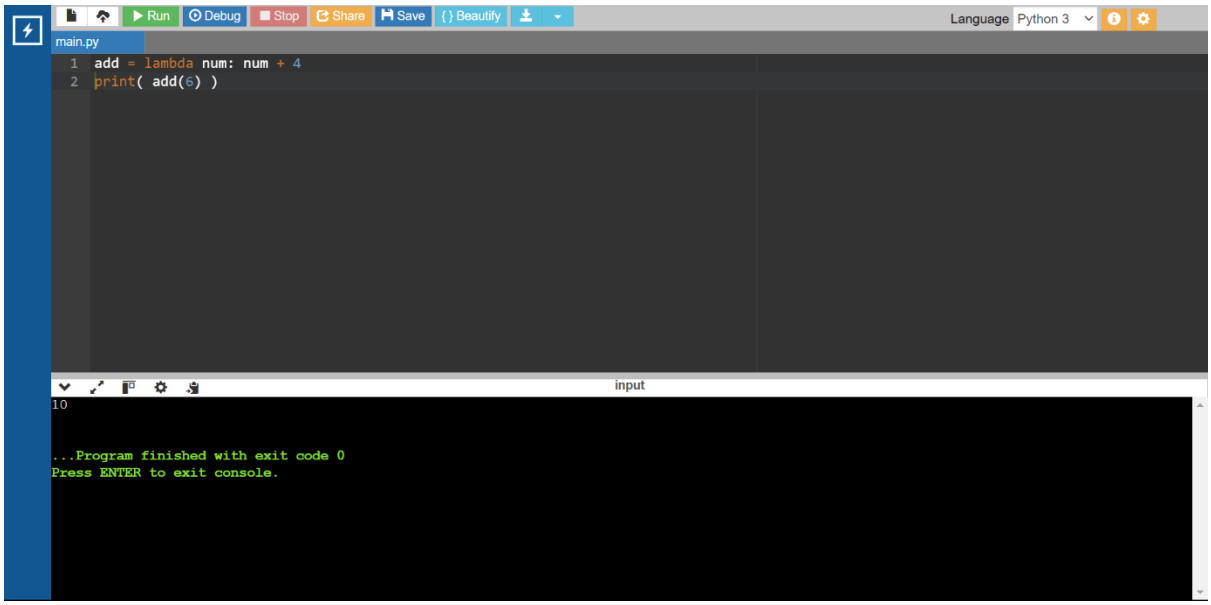
70. Python vars() Function Example



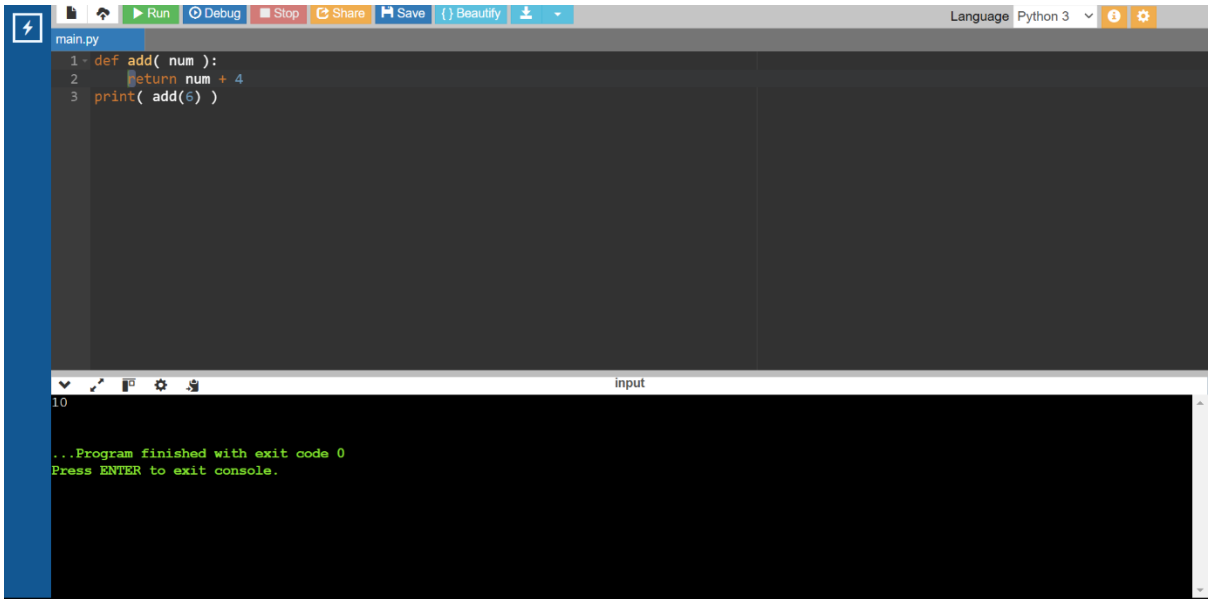
71. Python zip() Function Example



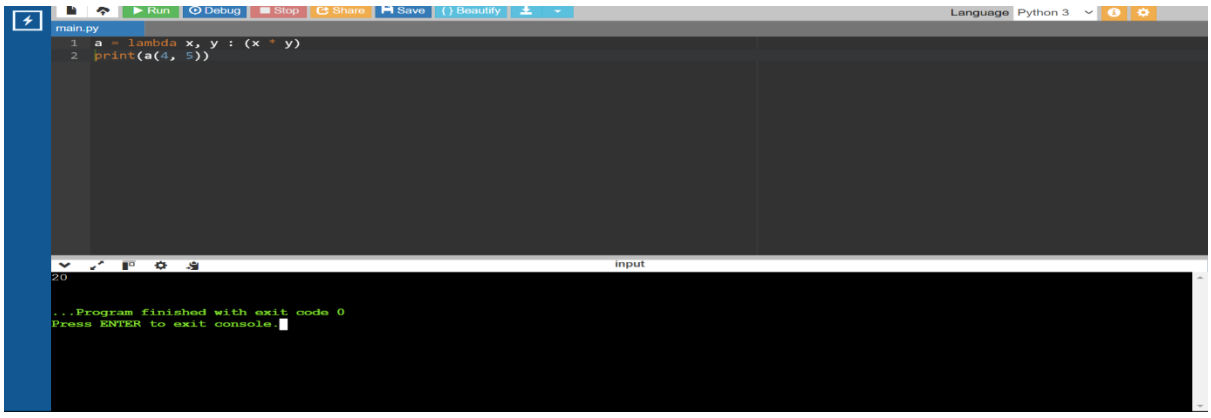
72. Code to demonstrate how we can use a lambda function for adding 4 numbers



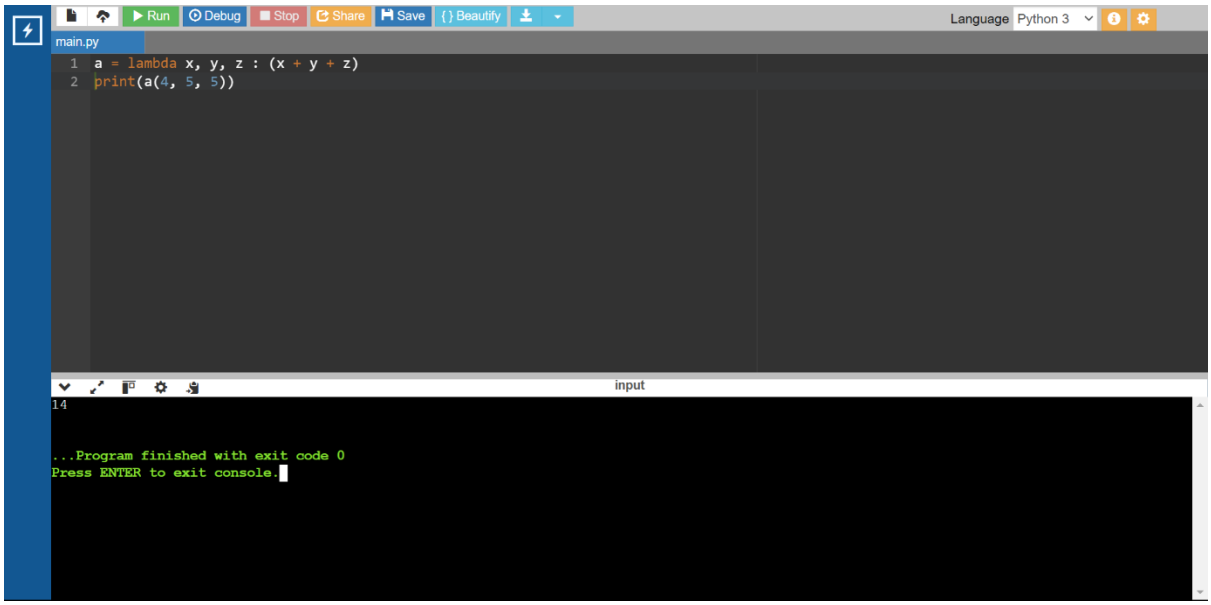
73. Example of a lambda function that adds 4 to the input number using the add function.



74. Example of a lambda function that multiply 2 numbers and return one result.

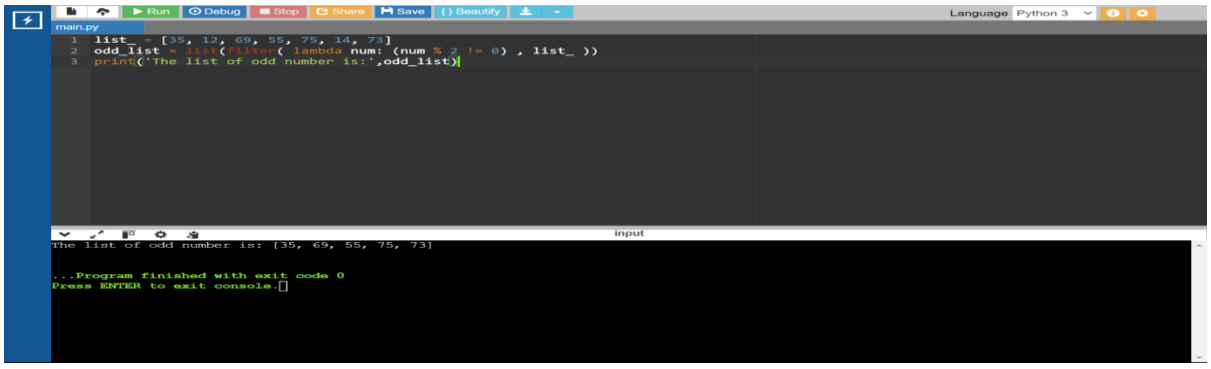


75. Another example of a lambda function that adds 2 numbers and return one result.

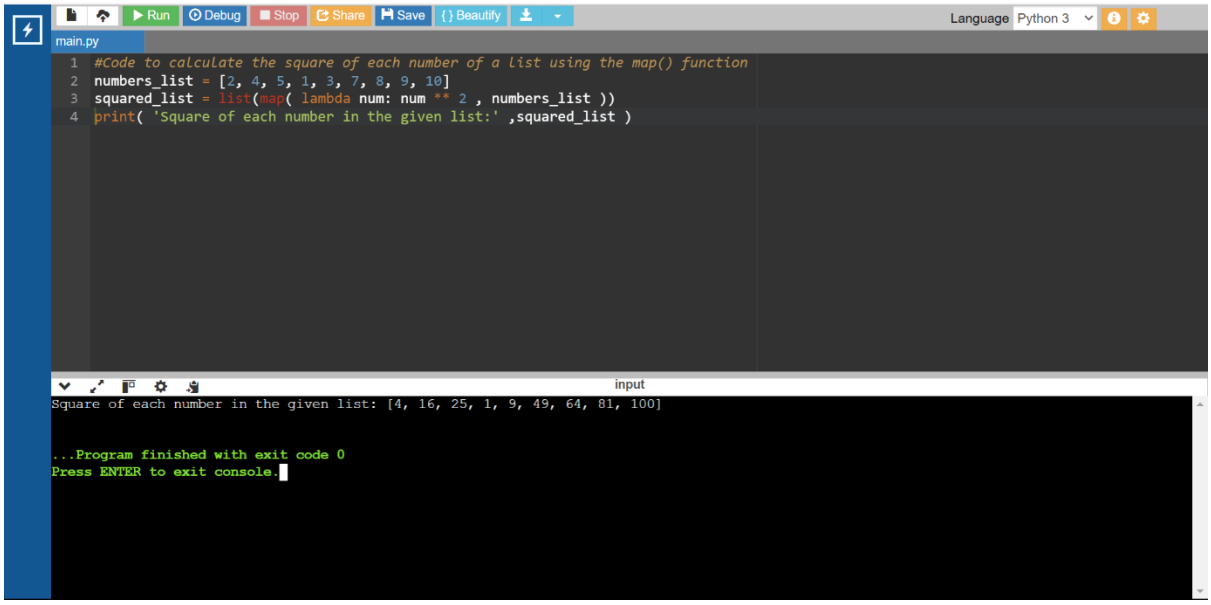


76. Python code to show the reciprocal of the given number to highlight the difference between def() and lambda().

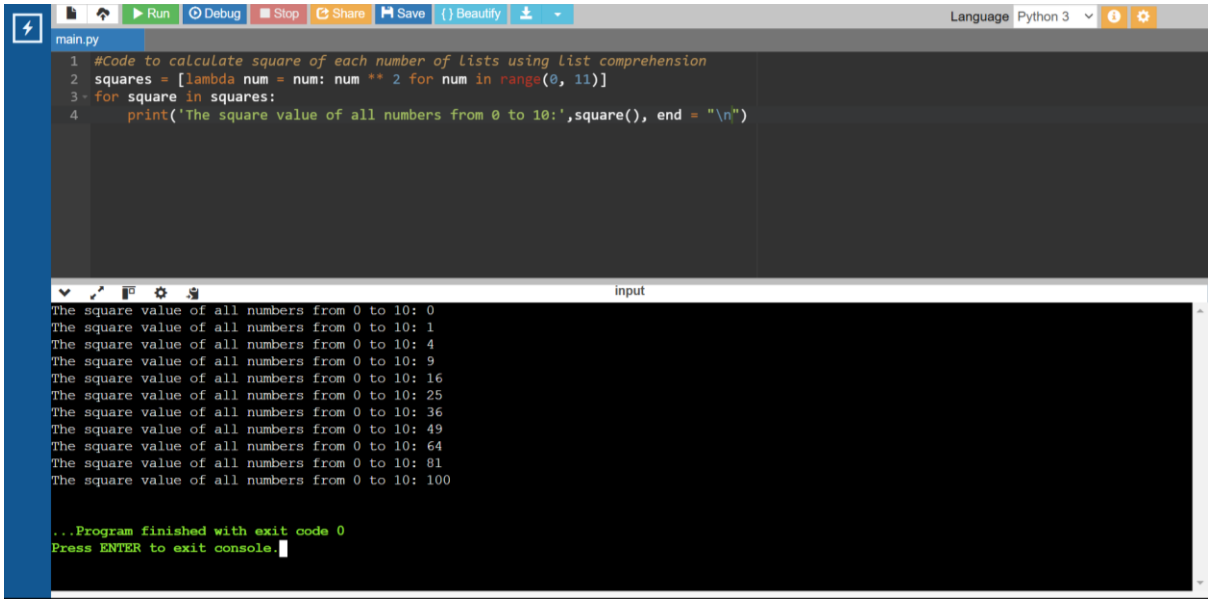
77. Example of lambda function with filter() in Python.



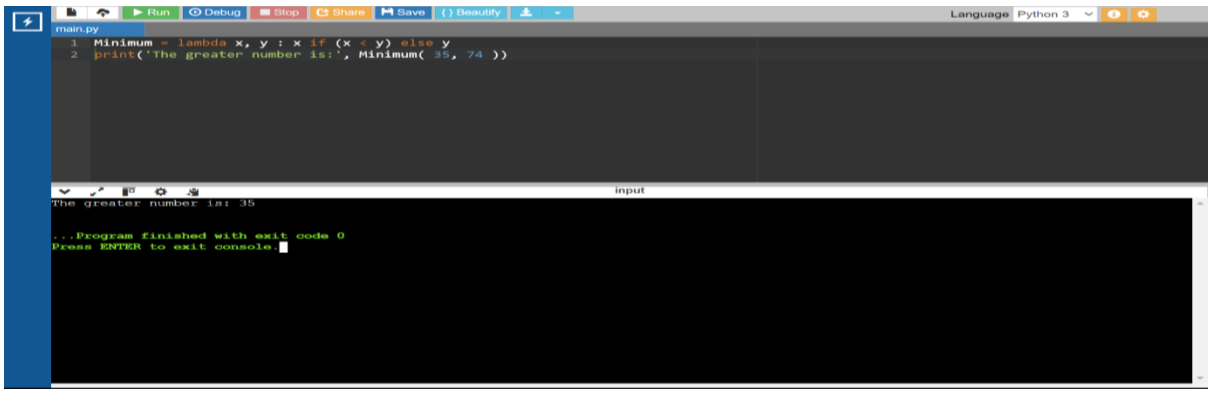
78. Code to calculate the square of each number of a list using the map() function



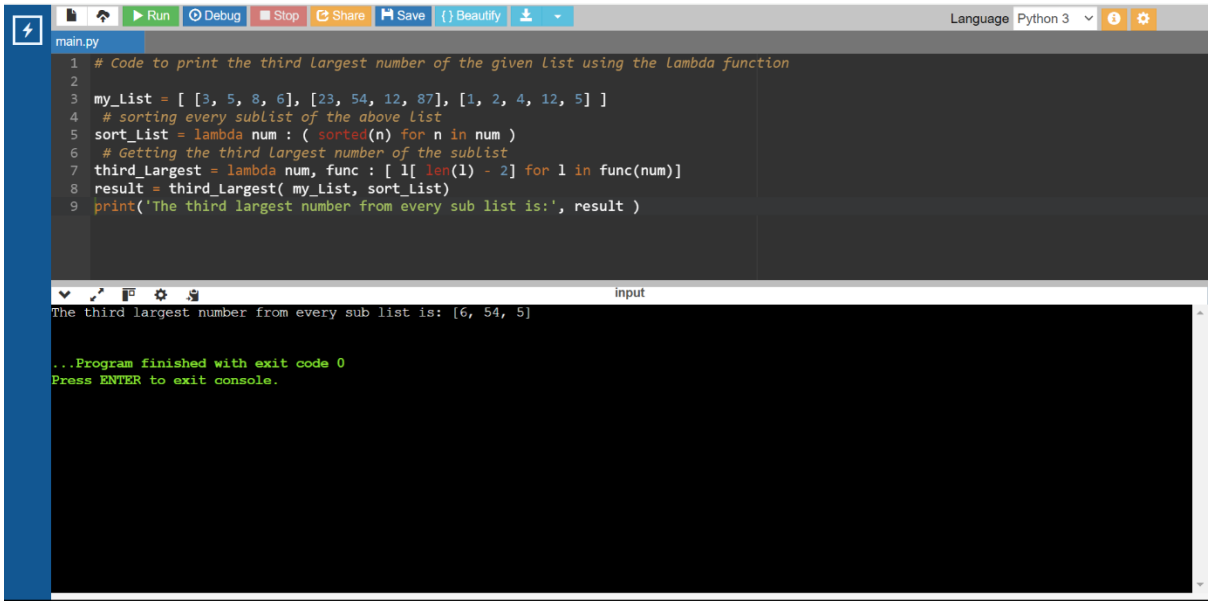
79. Code to calculate square of each number of lists using list comprehension



80. Code to use lambda function with if-else

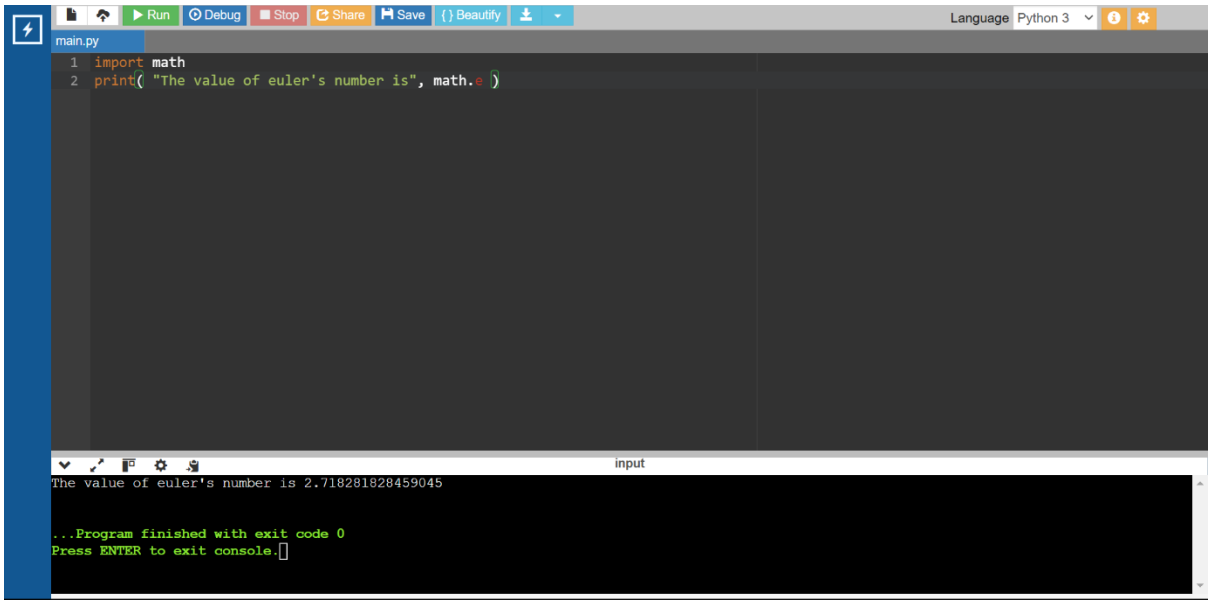


81. Code to print the third largest number of the given list using the lambda function

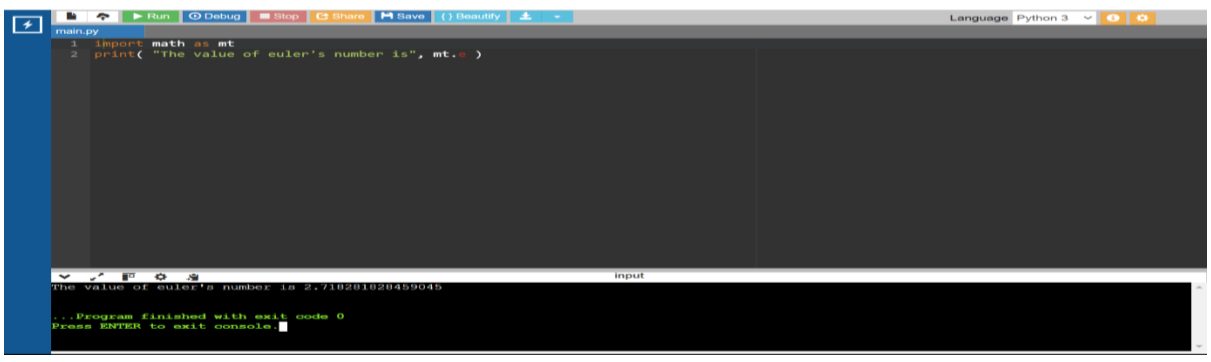


MODULES

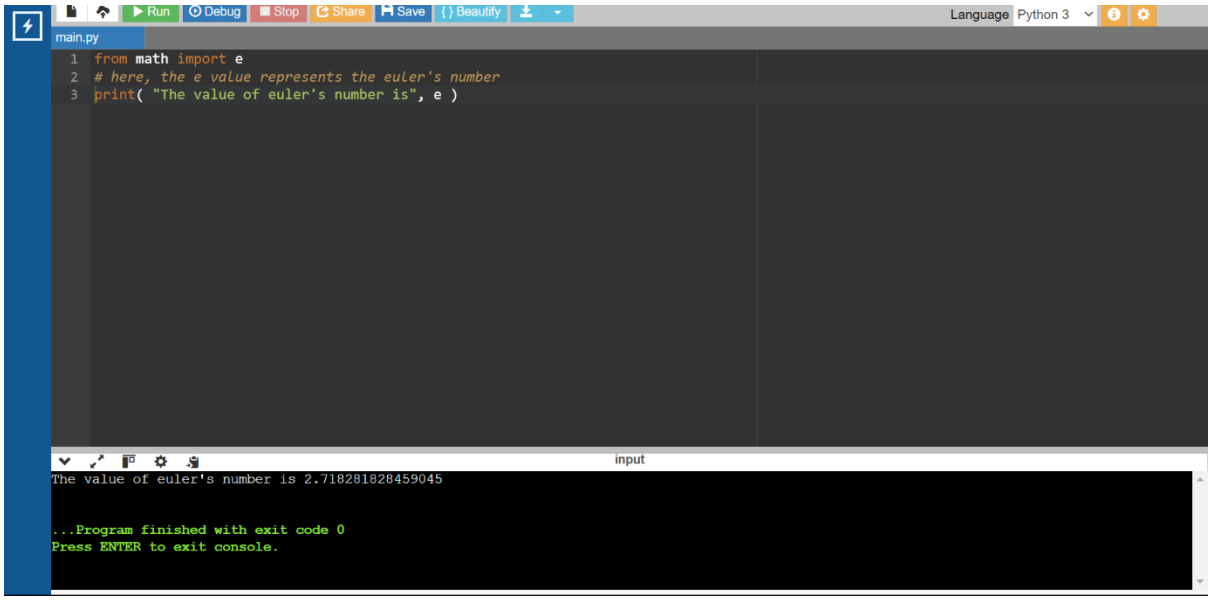
82. Python import Statement



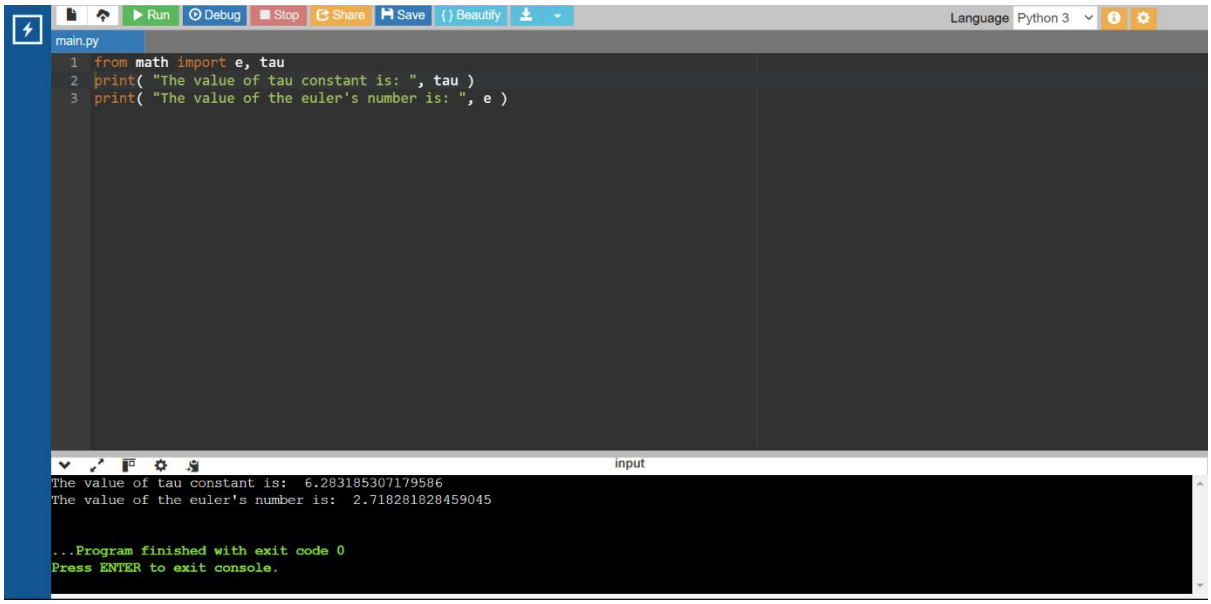
83. Importing and also Renaming



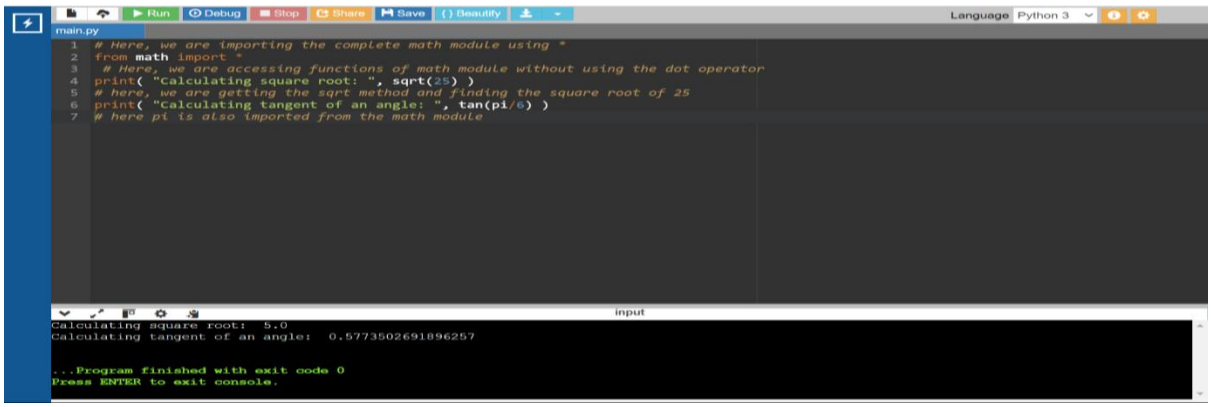
84. Python from...import Statement



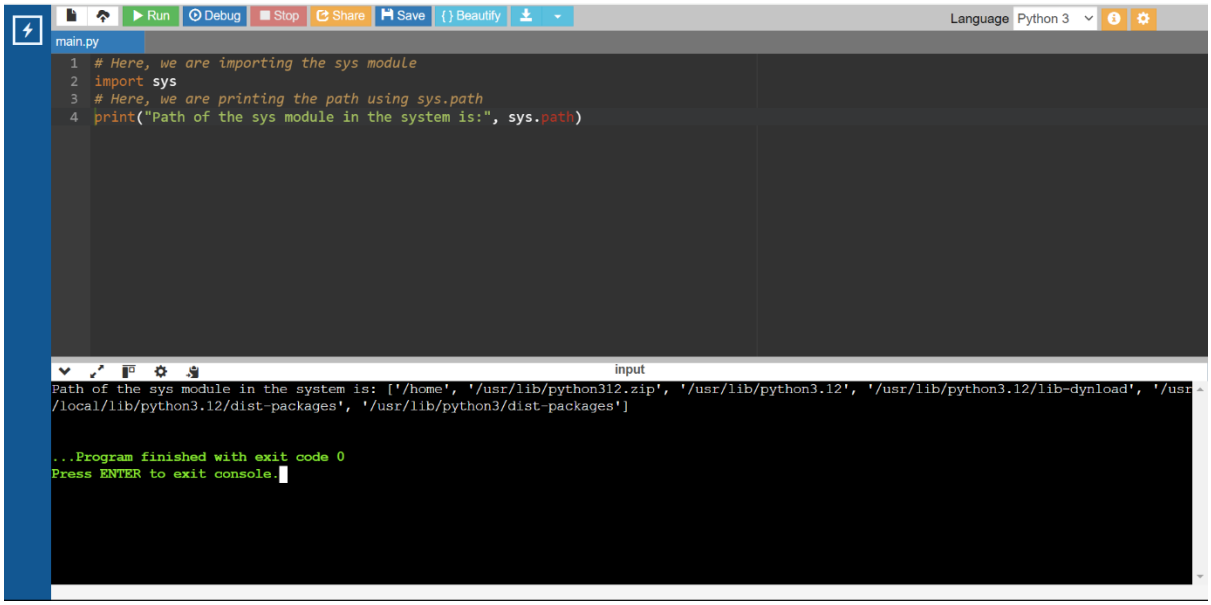
85. We are creating a simple Python program to show how to import multiple



86. we are importing the complete math module using *

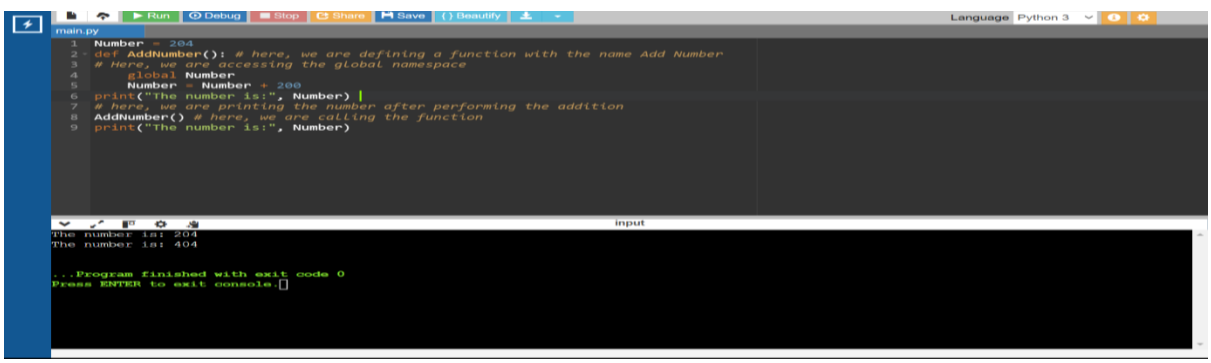


87. Example to print the path.



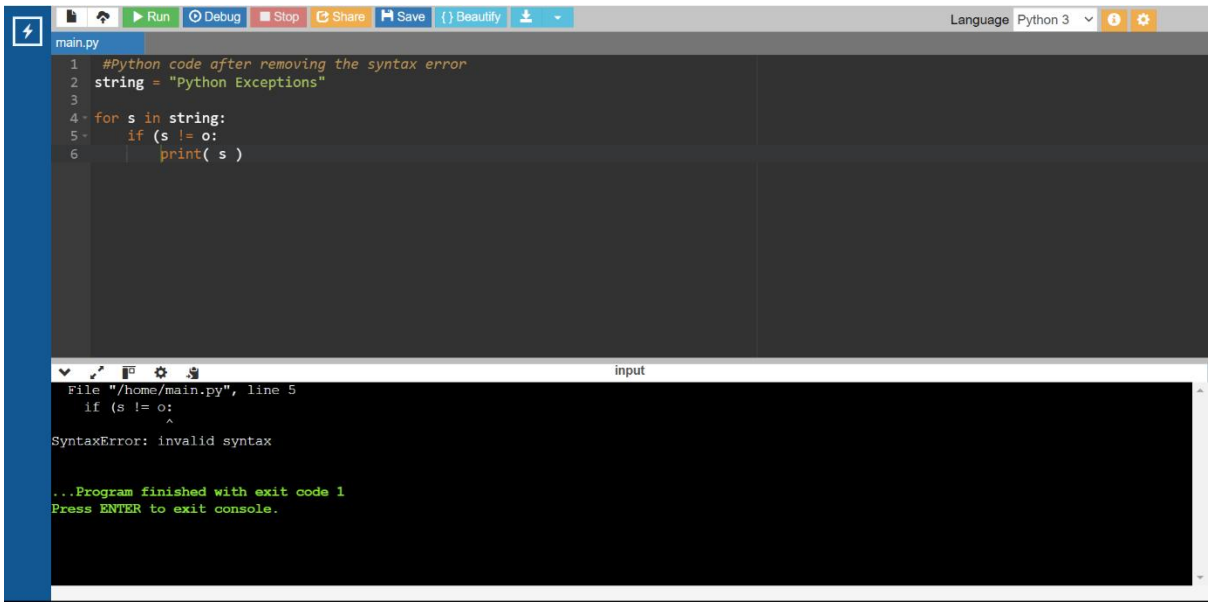
88. A simple Python program to print the directory of a module

89. Namespaces and scoping.

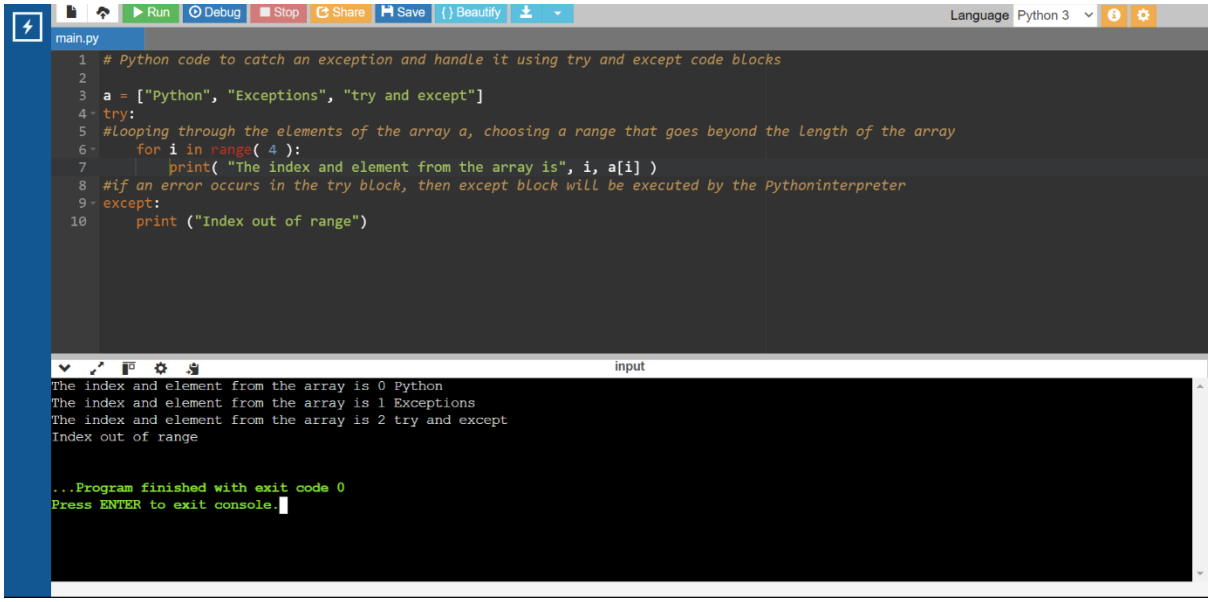


EXCEPTION

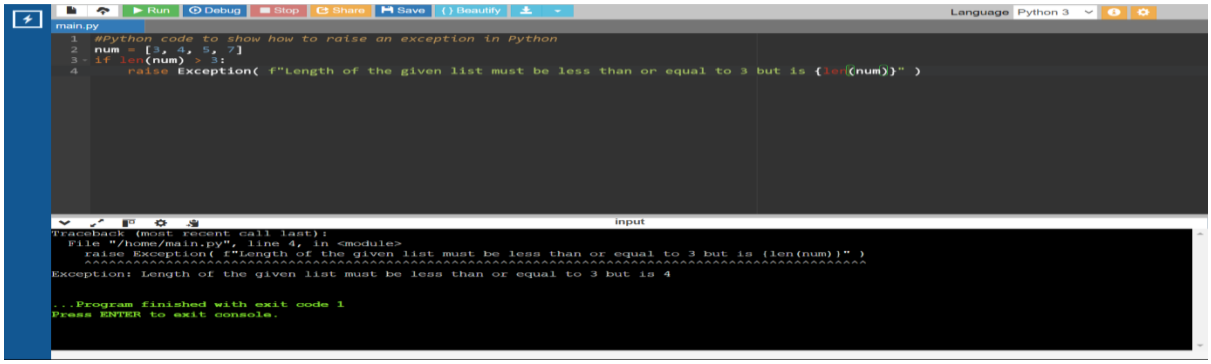
90. Exceptions versus Syntax Errors.



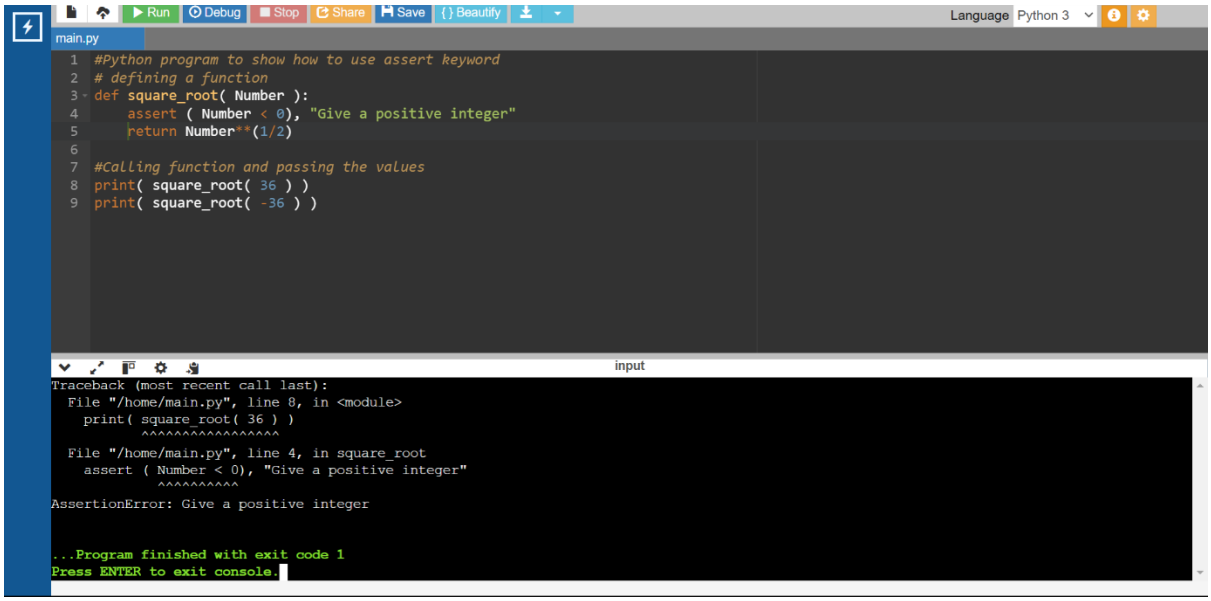
91. Python code to catch an exception and handle it using try and except code blocks.



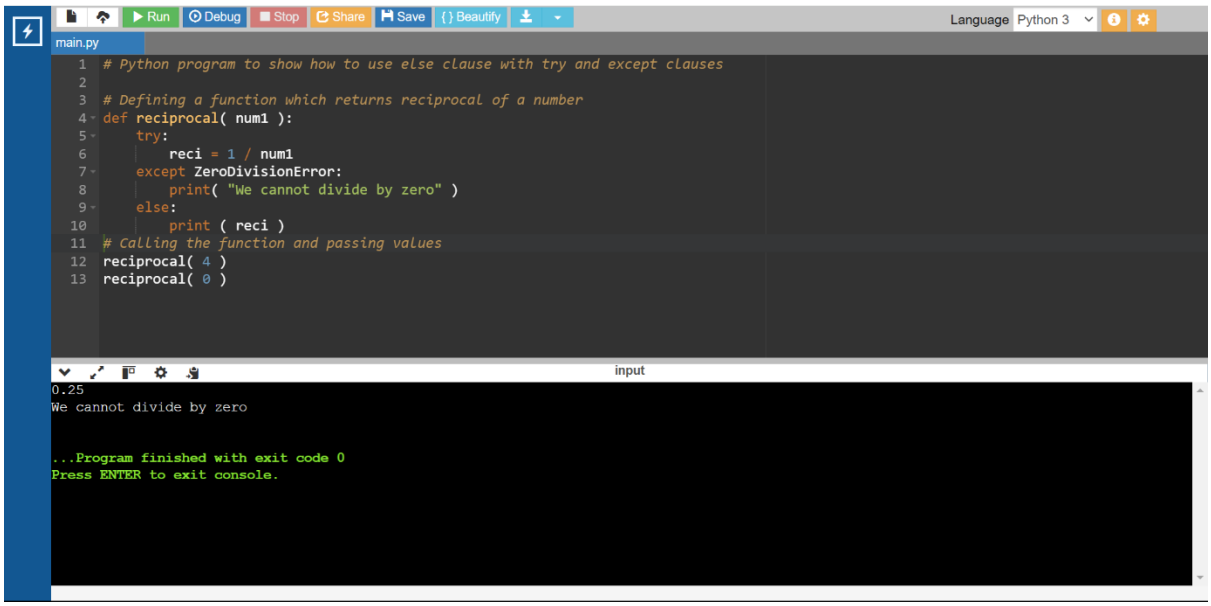
92. Python code to show how to raise an exception in Python



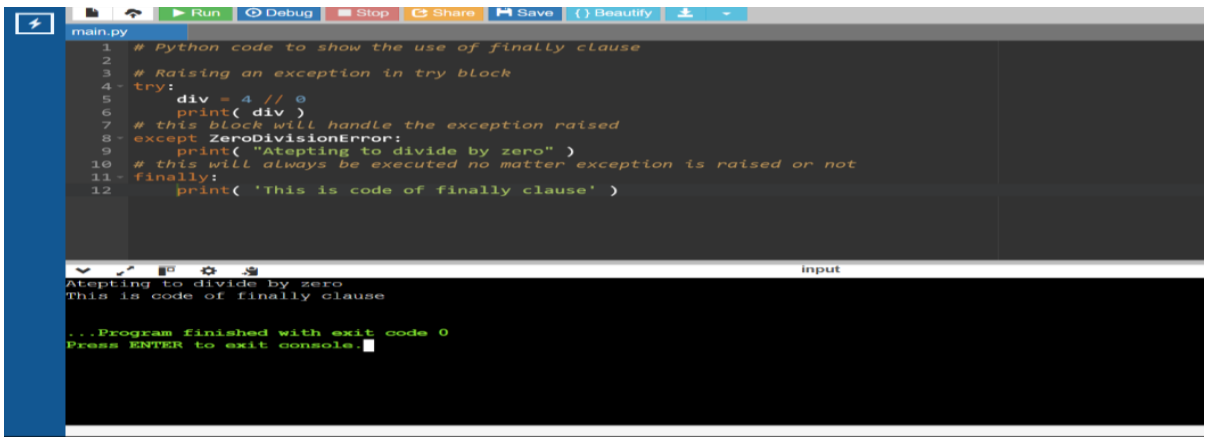
93. The assert Statement



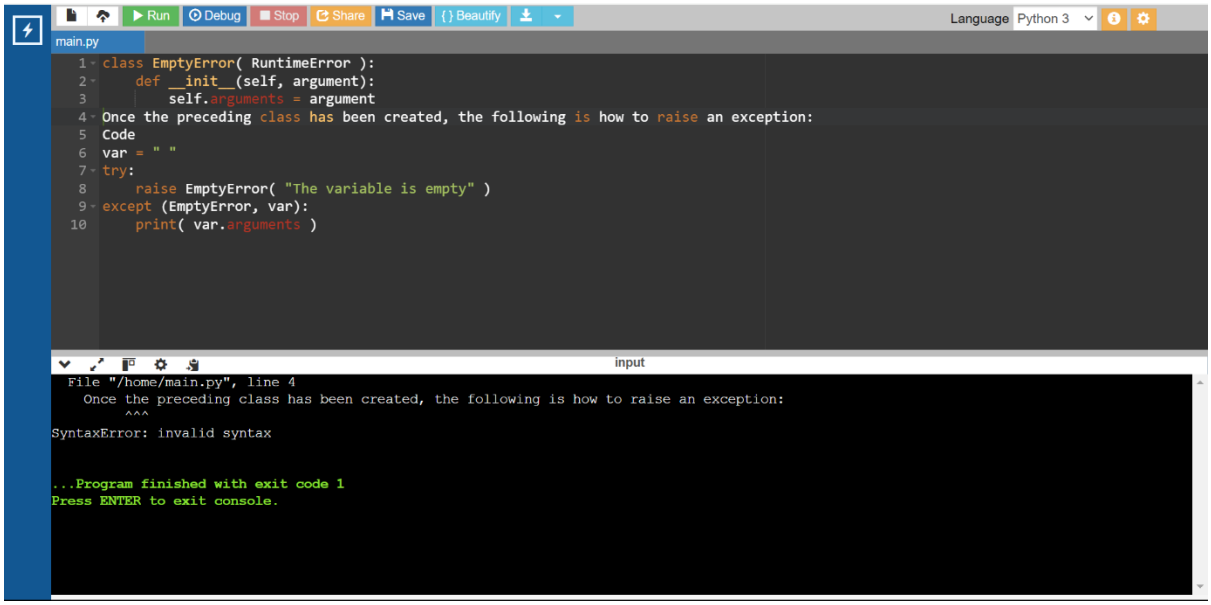
94. Try with Else Clause



95. Finally Keyword in Python



96. User-Defined Exceptions

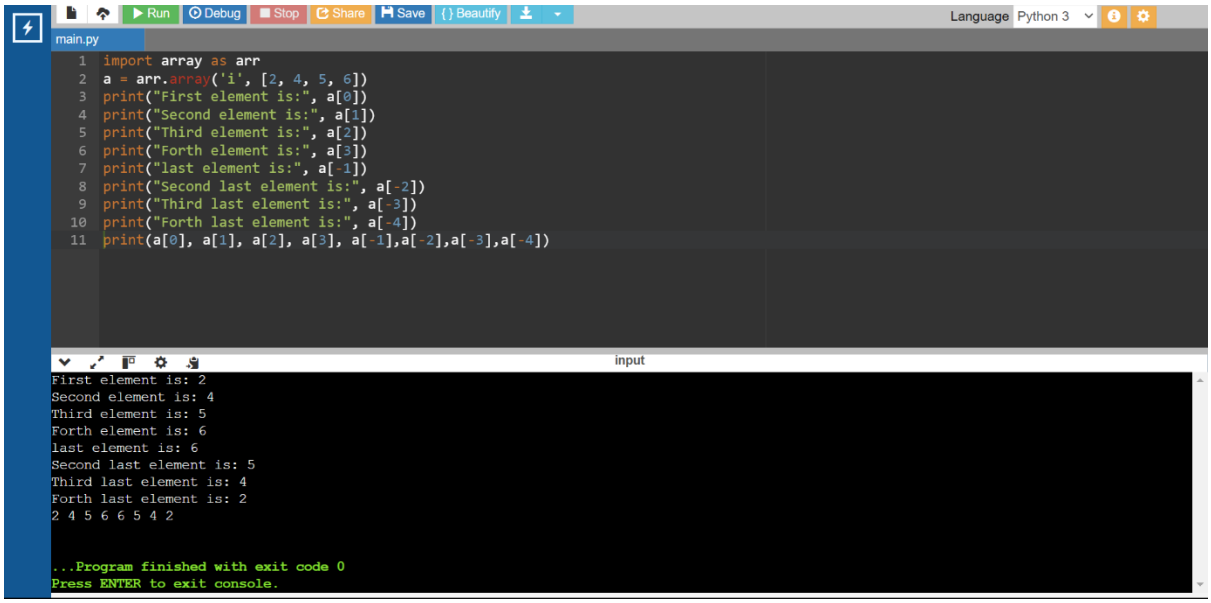


```
class EmptyError( RuntimeError ):
    def __init__(self, argument):
        self.arguments = argument
Once the preceding class has been created, the following is how to raise an exception:
Code
var = " "
try:
    raise EmptyError( "The variable is empty" )
except (EmptyError, var):
    print( var.arguments )
```

```
  File "/home/main.py", line 4
    Once the preceding class has been created, the following is how to raise an exception:
         ^^^
SyntaxError: invalid syntax


...Program finished with exit code 1
Press ENTER to exit console.
```

ARRAY

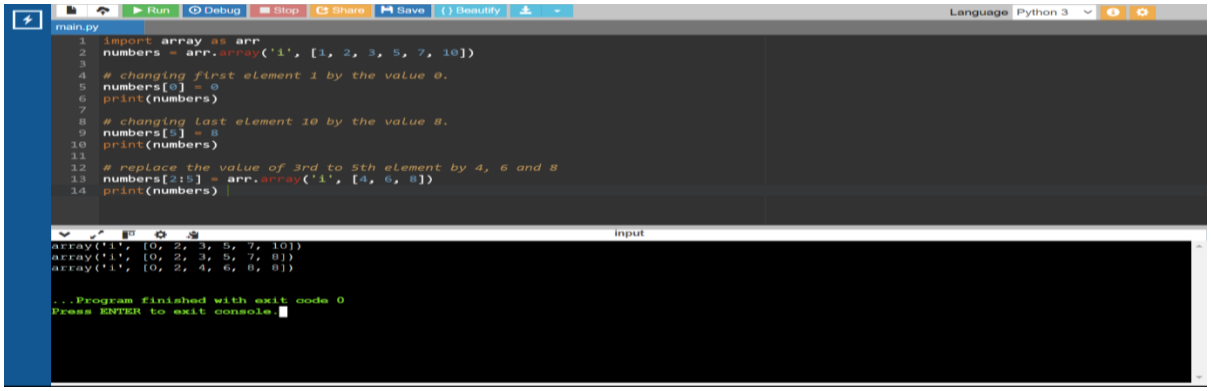
97. Accessing array elements



```
import array as arr
a = arr.array('i', [2, 4, 5, 6])
print("First element is:", a[0])
print("Second element is:", a[1])
print("Third element is:", a[2])
print("Forth element is:", a[3])
print("last element is:", a[-1])
print("Second last element is:", a[-2])
print("Third last element is:", a[-3])
print("Forth last element is:", a[-4])
print(a[0], a[1], a[2], a[3], a[-1],a[-2],a[-3],a[-4])
```

```
First element is: 2
Second element is: 4
Third element is: 5
Forth element is: 6
last element is: 6
Second last element is: 5
Third last element is: 4
Forth last element is: 2
2 4 5 6 6 5 4 2


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

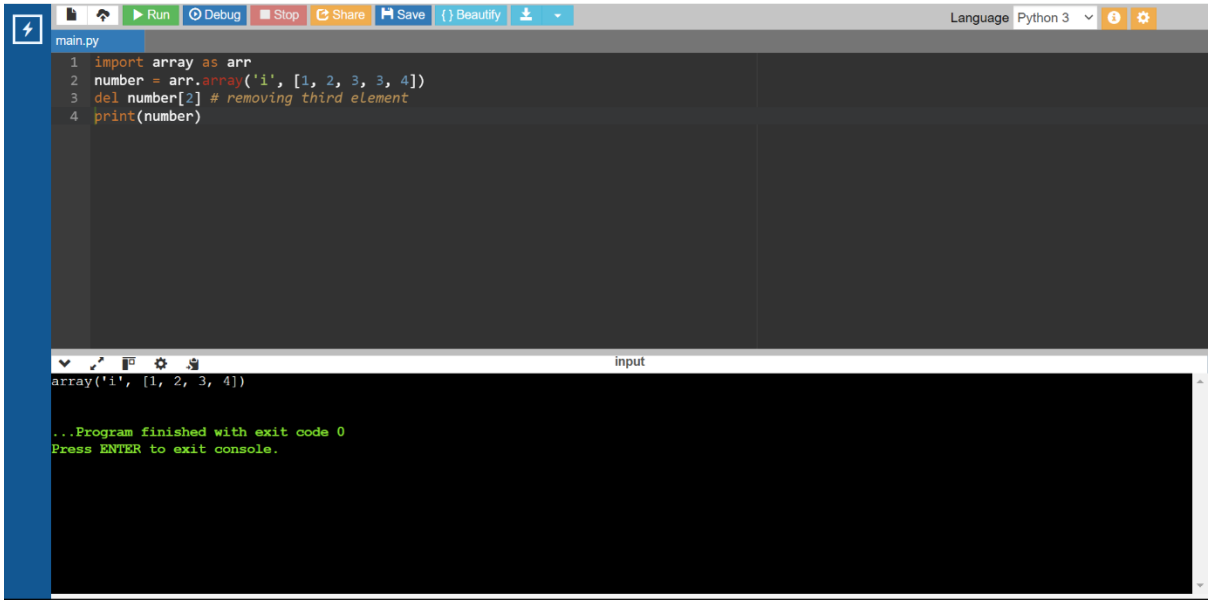
98. How to change or add elements?



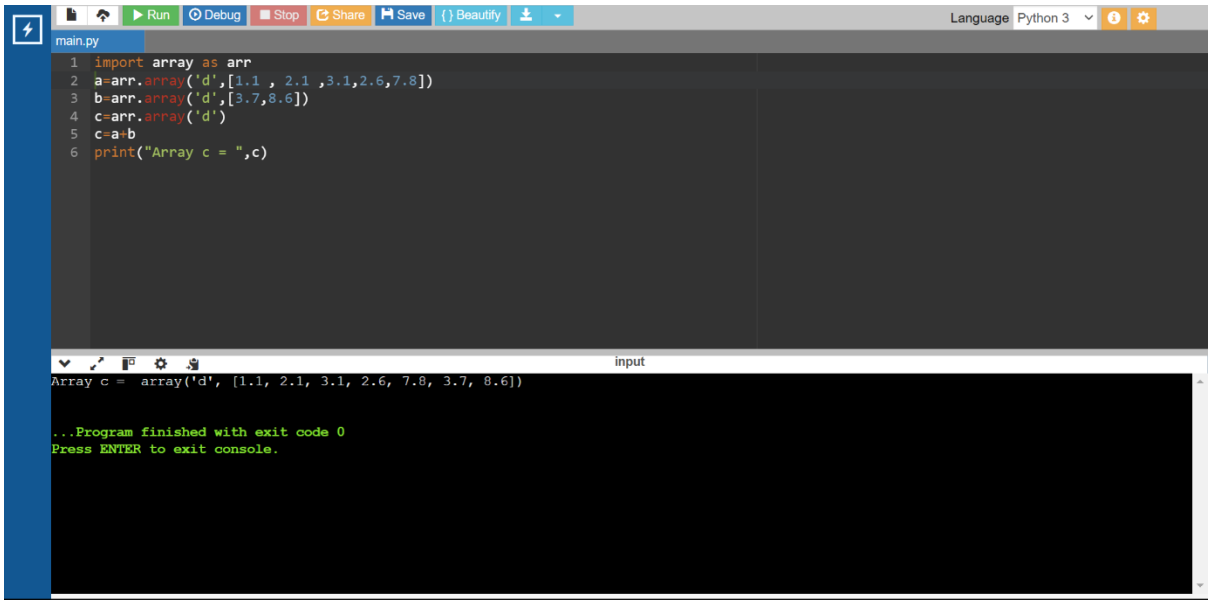
```
import array as arr
numbers = arr.array('i', [1, 2, 3, 5, 7, 10])

# changing first element 1 by the value 0.
numbers[0] = 0
print(numbers)

# changing last element 10 by the value 8.
numbers[5] = 8
print(numbers)

# replace the value of 3rd to 5th element by 4, 6 and 8
numbers[2:5] = arr.array('i', [4, 6, 8])
print(numbers)
```

```
array('i', [0, 2, 3, 5, 7, 10])
array('i', [0, 2, 3, 5, 7, 8])
array('i', [0, 2, 4, 6, 8, 8])

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

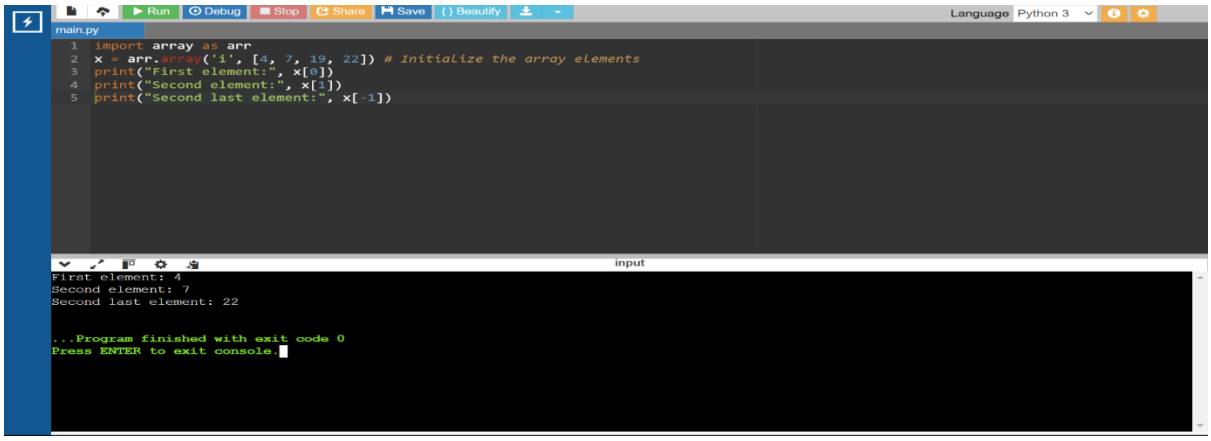
99. Delete Elements from an Array



100. Array Concatenation



101. Array Concatenation



DECORATORS:

102. Sample example

```
def func1(msg):
    print(msg)
func1("Hii, welcome to function ")
func2 = func1
func2("Hii, welcome to function ")
```

```
Hii, welcome to function
Hii, welcome to function


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

103. Inner Function

```
def func():
    print("We are in first function")
def func1():
    print("This is first child function")

def func2():
    print("This is second child function")
func1()
func2()
func()
```

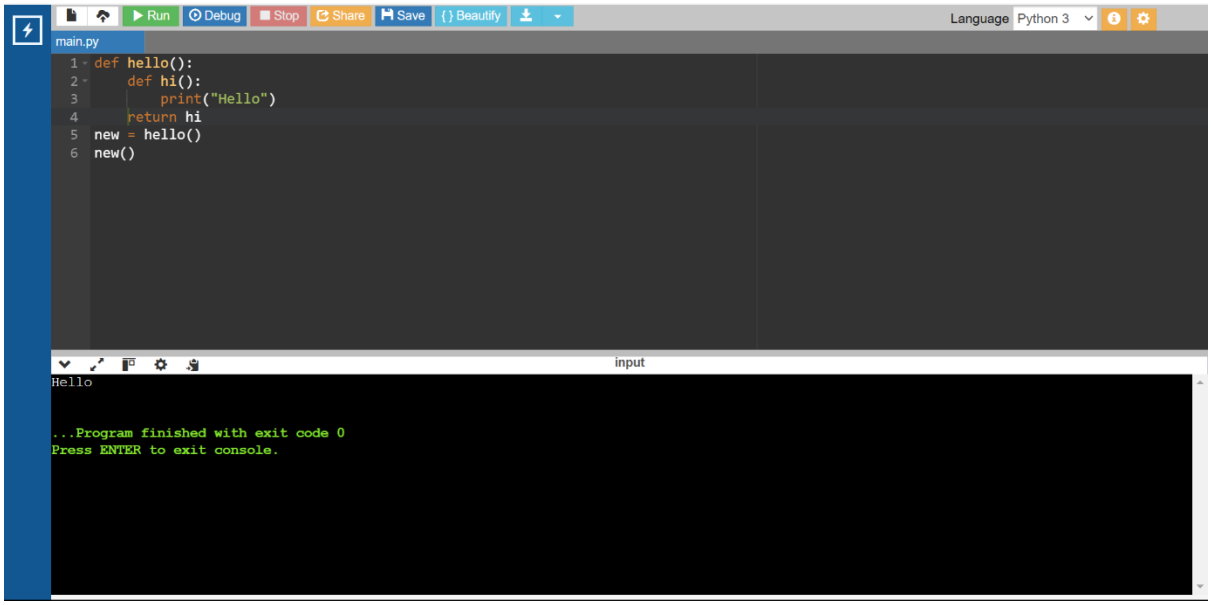
```
This is first child function
This is second child function
We are in first function


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

104. Inner Function

```
def add(x):
    return x+1
def sub(x):
    return x-1
def operator(func, x):
    temp = func(x)
    return temp
print(operator(sub,10))
print(operator(add,20))
```

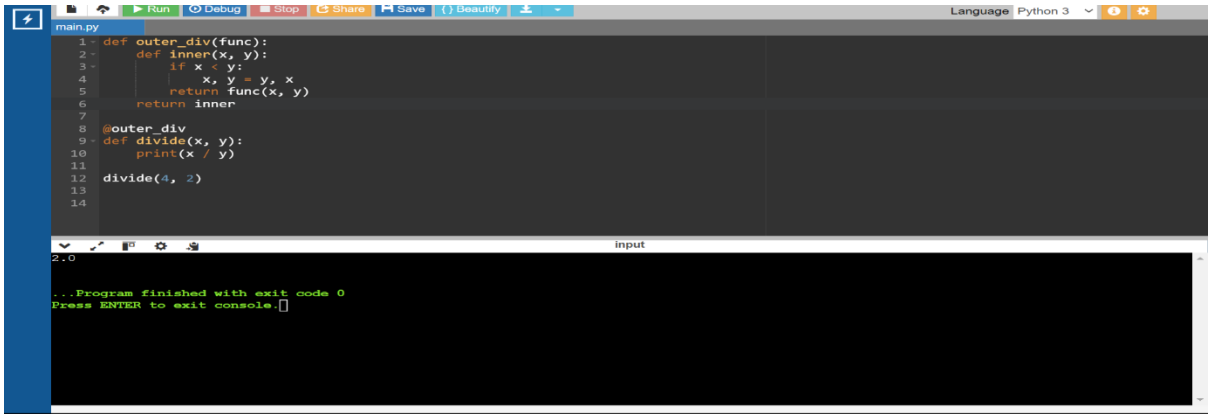
```
9
21


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

105. Inner Function

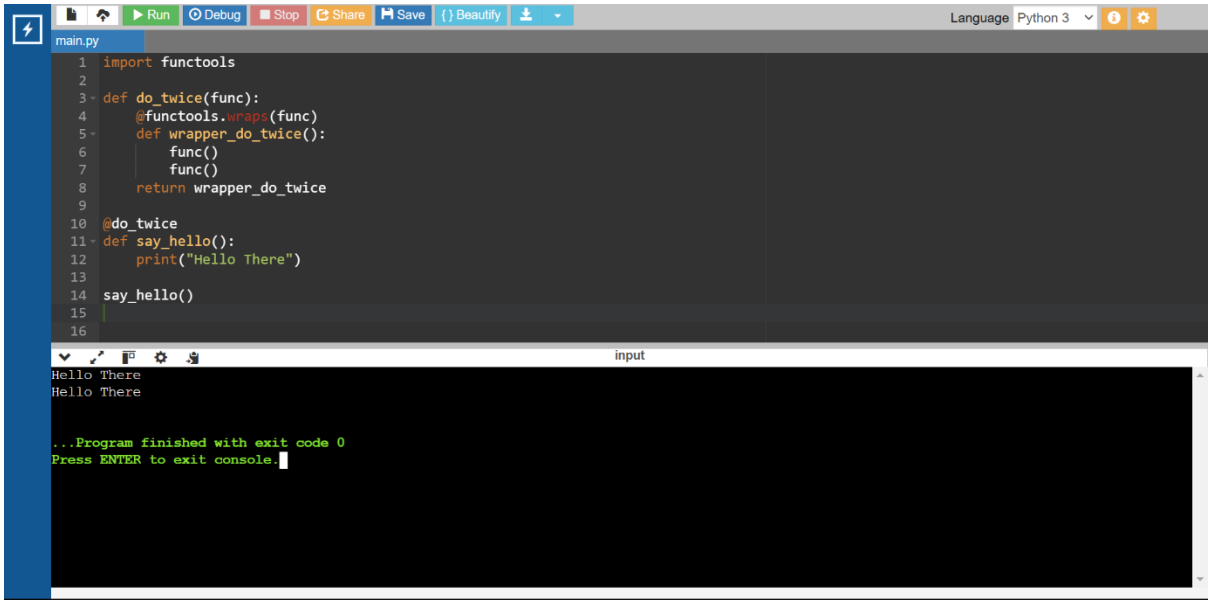


106. Decorating functions with parameters

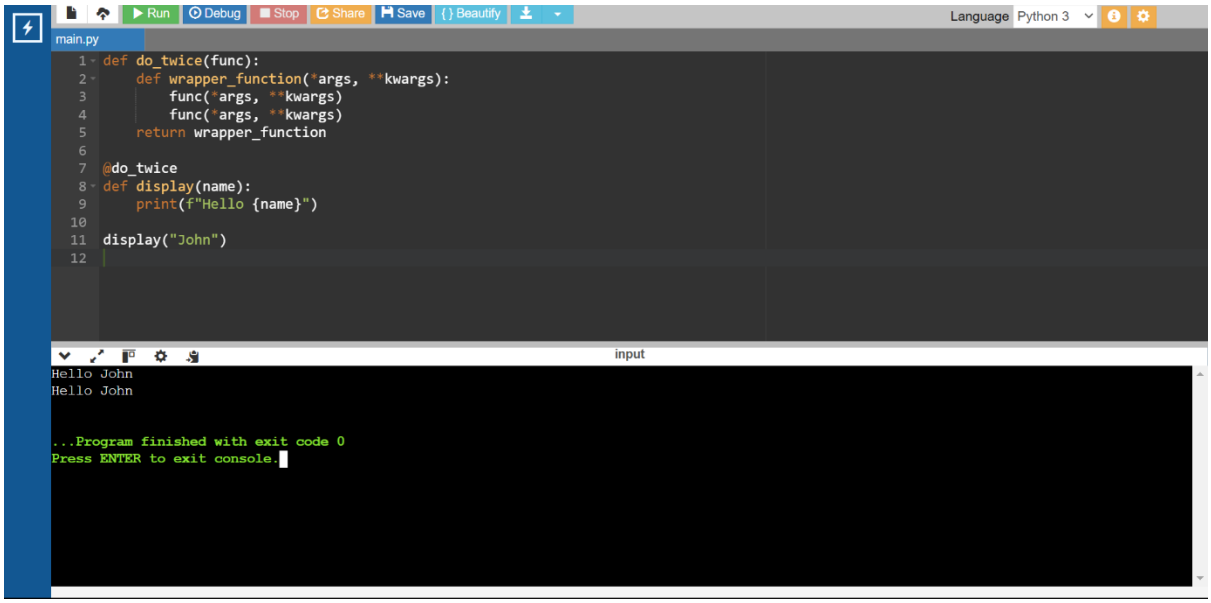
107. Syntactic Decorator



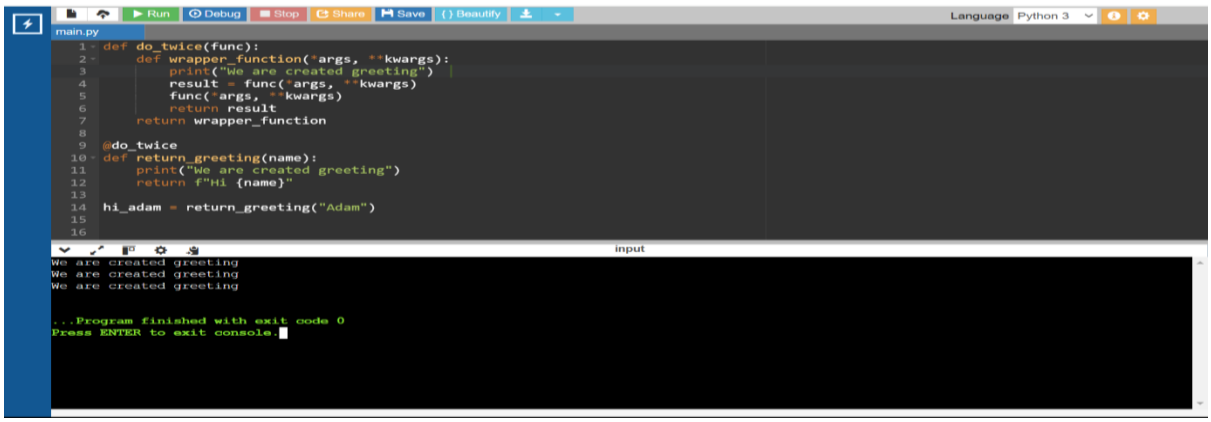
108. Reusing Decorator



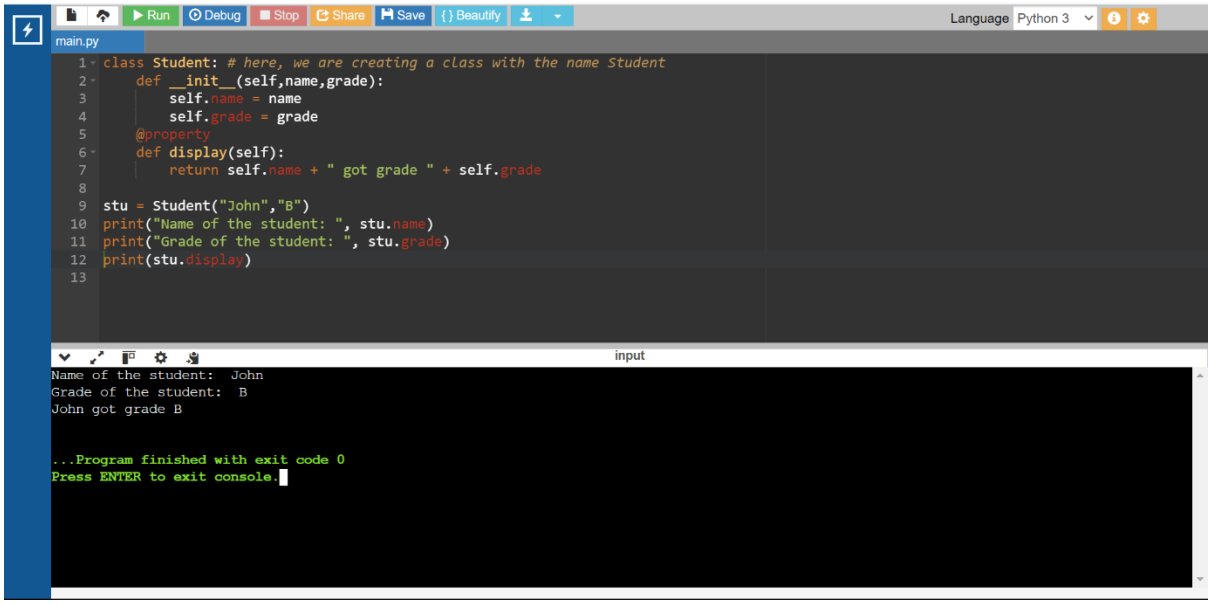
109. Python Decorator with Argument



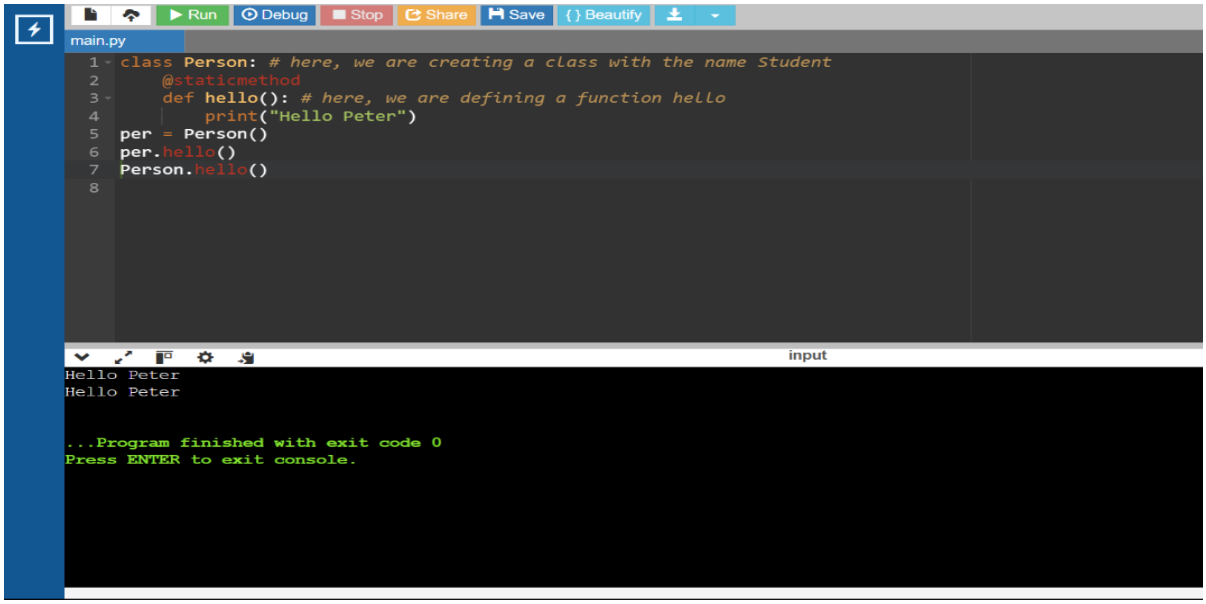
110. Returning Values from Decorated Functions



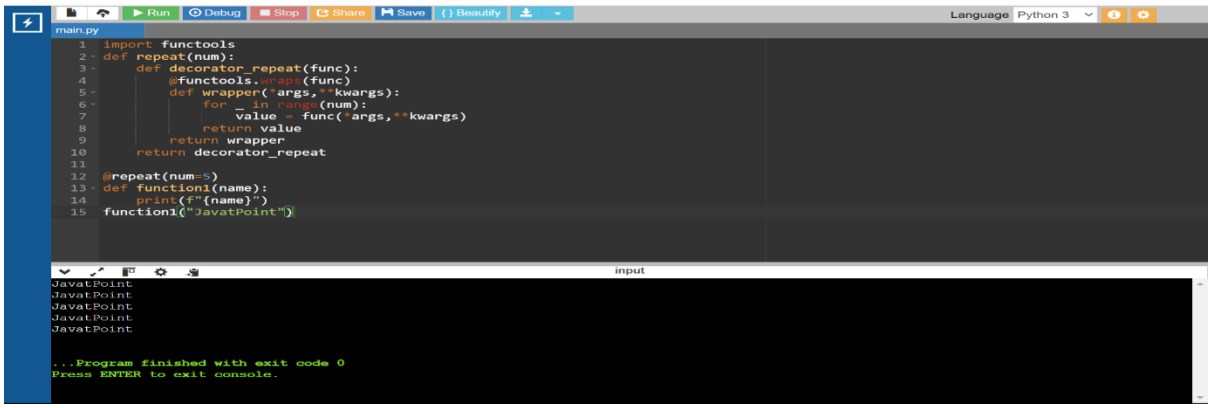
111. Class Decorators - @property decorator



112. Class Decorators - @staticmethod decorator



113. Decorator with Arguments



114. Stateful Decorators

115. Classes as Decorators