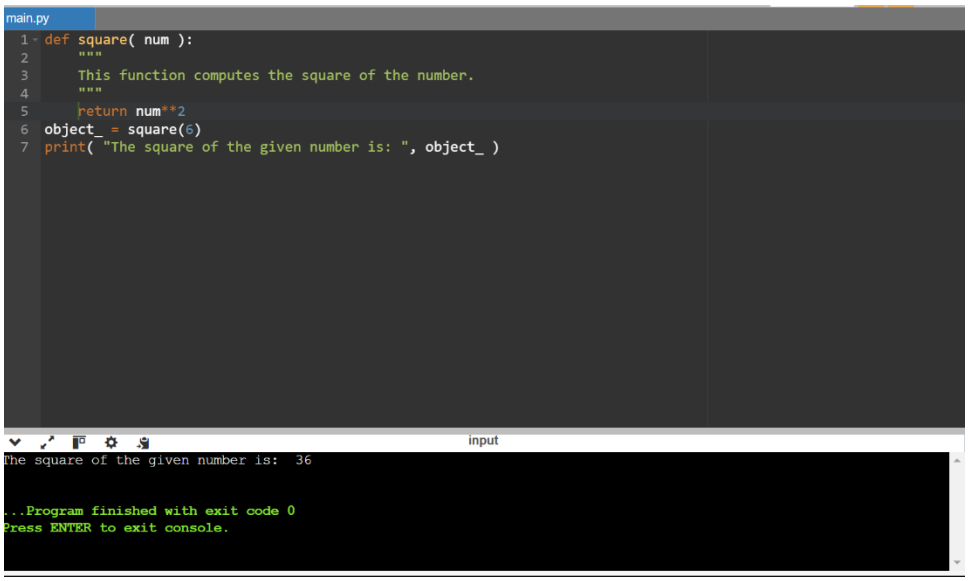
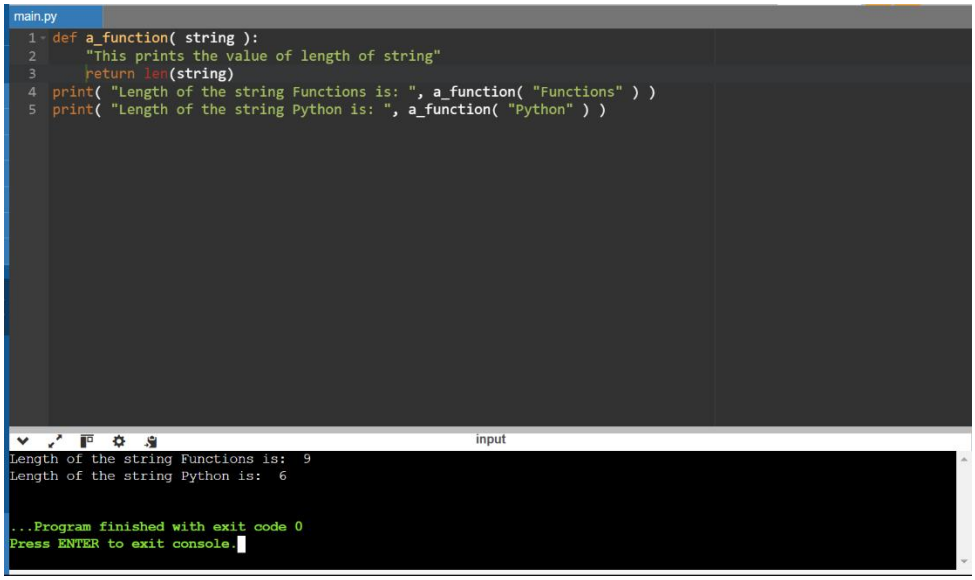
FUNCTION

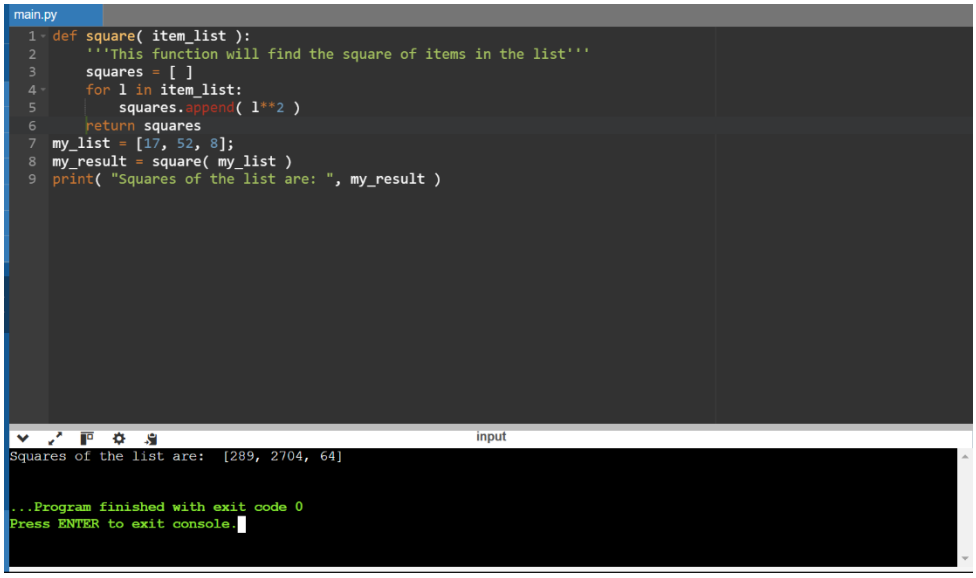
1. A function that returns the argument number's square when called.



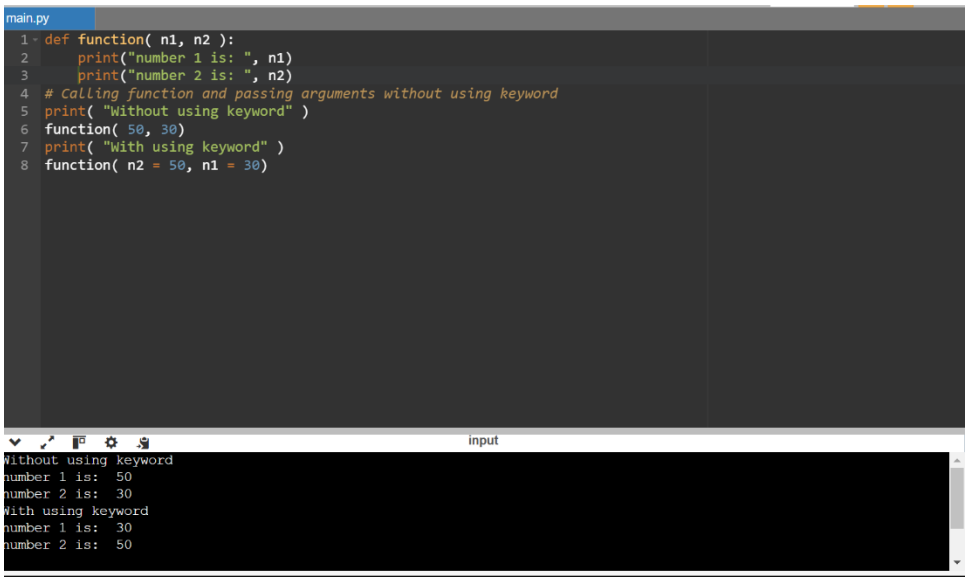
2. Example Python Code for calling a function.



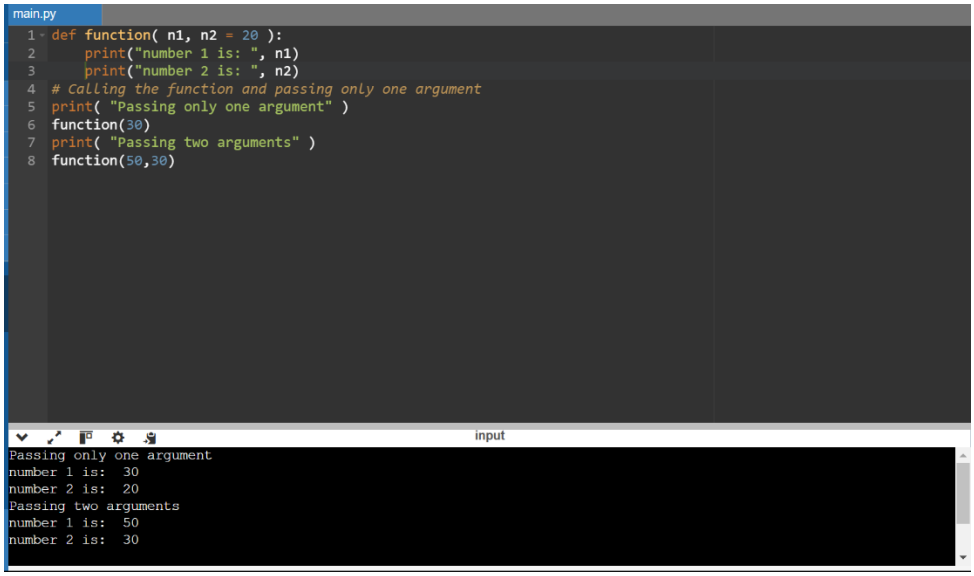
3. Python Code for Pass by Reference vs. Value



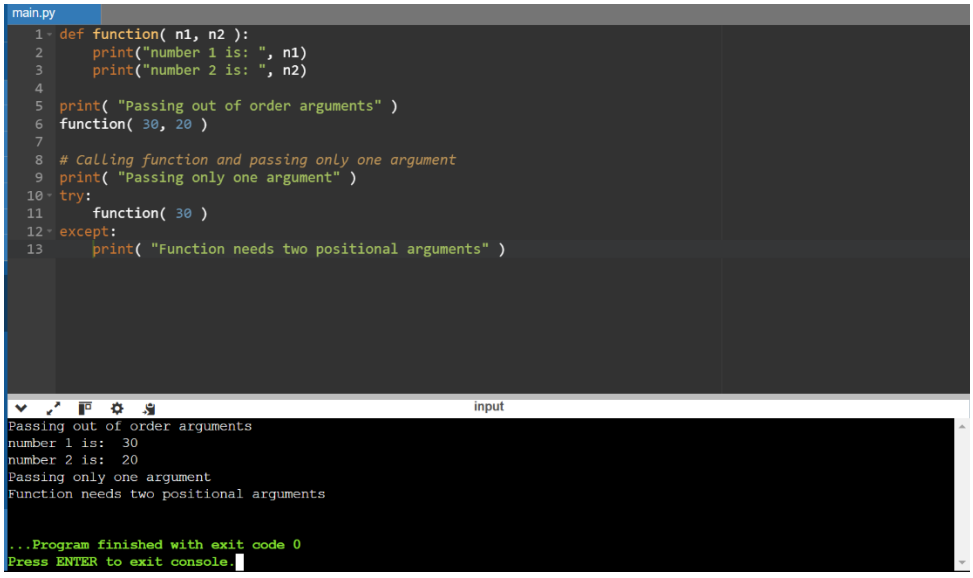
4. Python code to demonstrate the use of default arguments



5. Python code to demonstrate the use of keyword arguments



6. Python code to demonstrate the use of default arguments



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

main.py

```
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)
```

input

```
['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

main.py

```
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 ) )
```

input

```
With return statement
2704
Without return statement
None


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

9. Python code to demonstrate anonymous functions

main.py

```
lambda_ = lambda argument1, argument2: argument1 + argument2;
# Calling the function and passing values
print( "Value of the function is : ", lambda_( 20, 30 ) )
print( "Value of the function is : ", lambda_( 40, 50 ) )
```

input

```
Value of the function is :  50
Value of the function is :  90


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

10. Python code to demonstrate scope and lifetime of variable

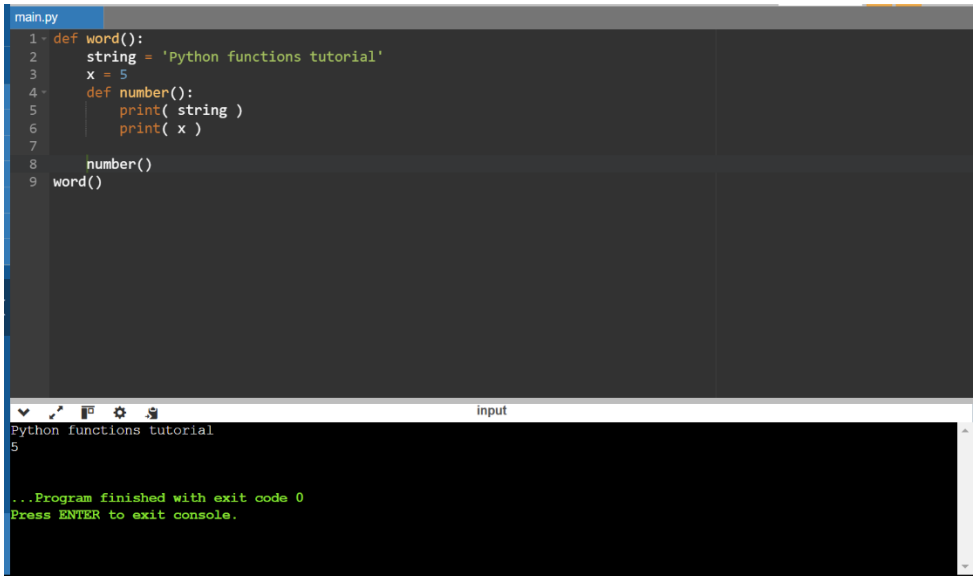
main.py

```
def number( ):
    num = 50
    print( "Value of num inside the function: ", num)

num = 10
number()
print( "Value of num outside the function:", num)
```

input

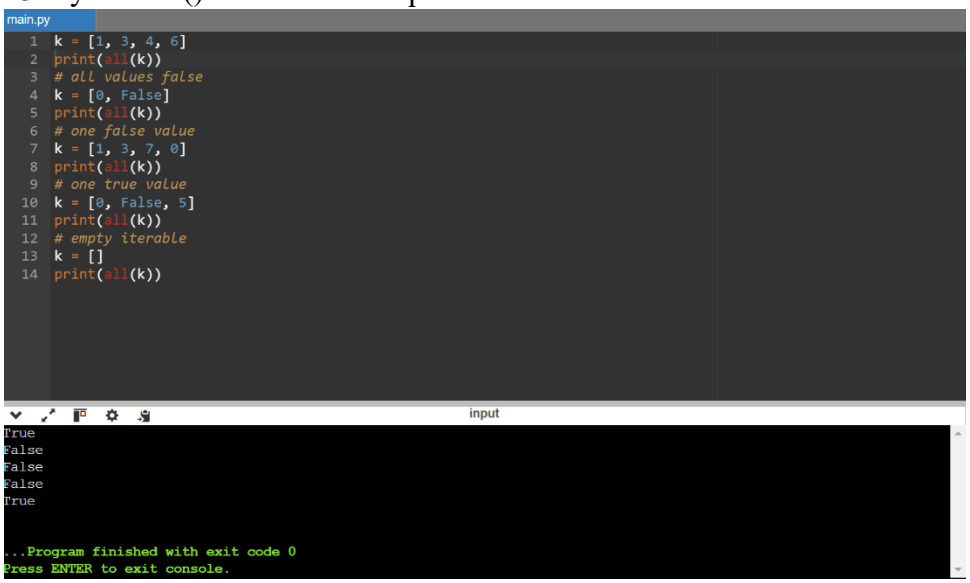
```
Value of num inside the function:  50
Value of num outside the function: 10


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

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

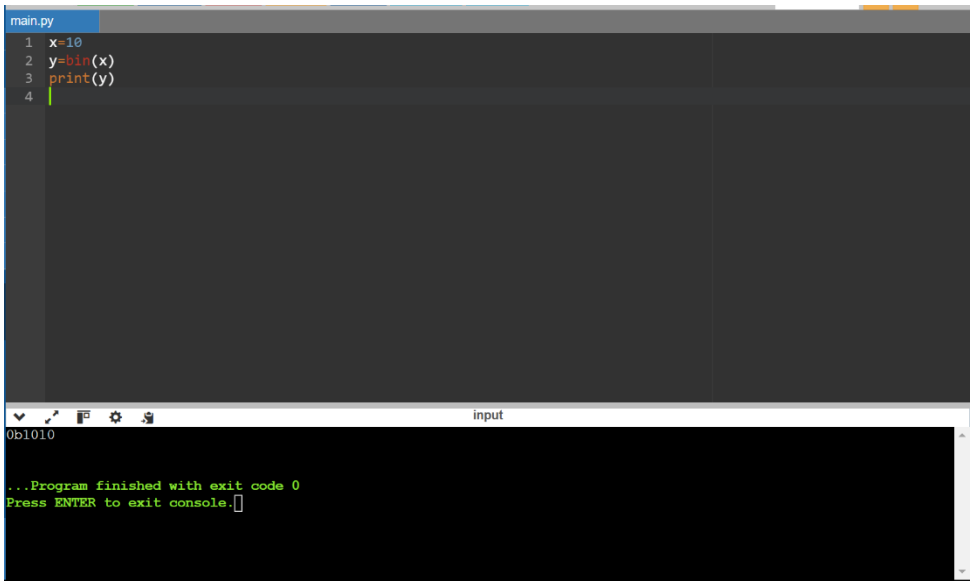


12. Python abs() Function Example

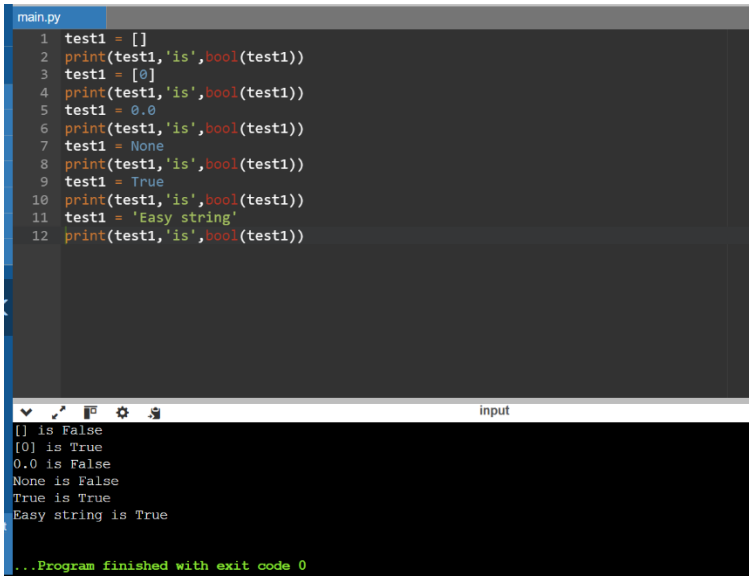
13. Python all() Function Example



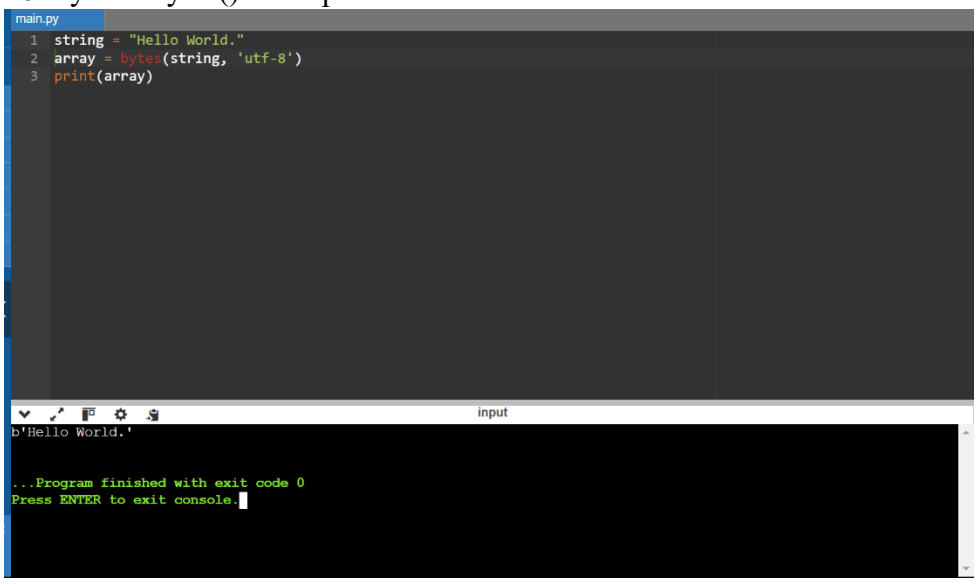
14. Python bin() Function Example



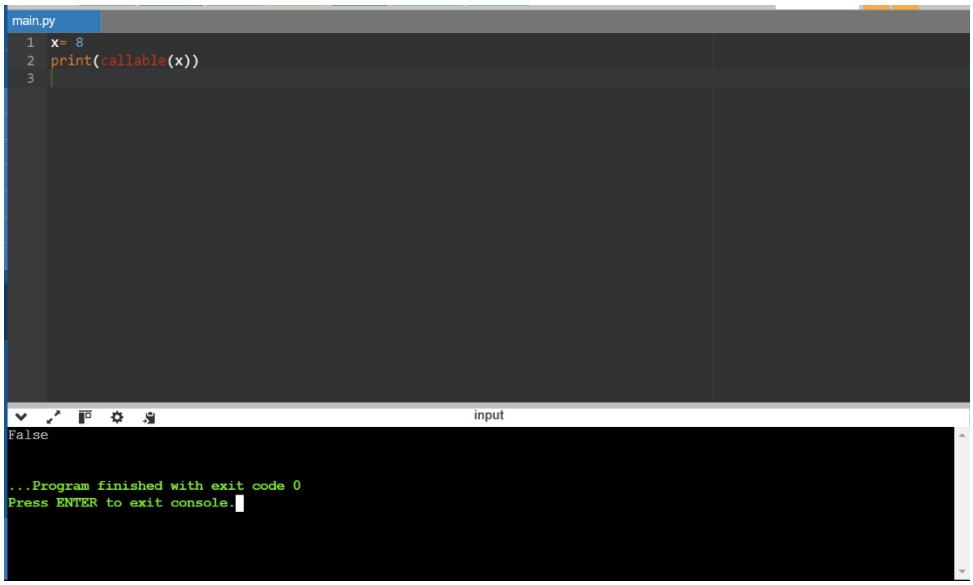
15. Python bool() Example



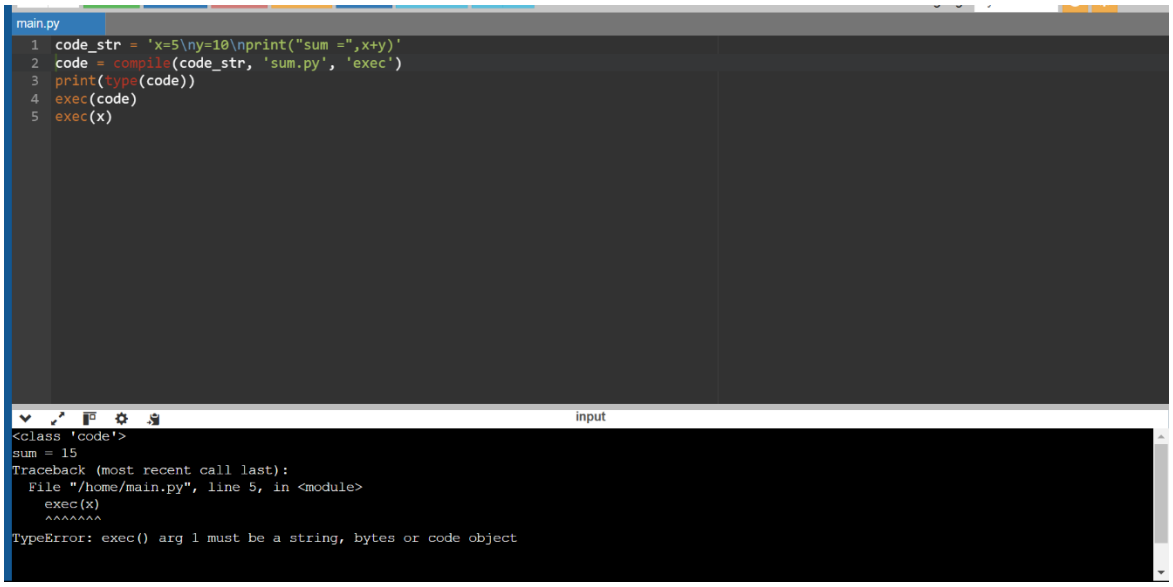
16. Python bytes() Example



17. Python callable() Function Example



18. Python compile() Function Example

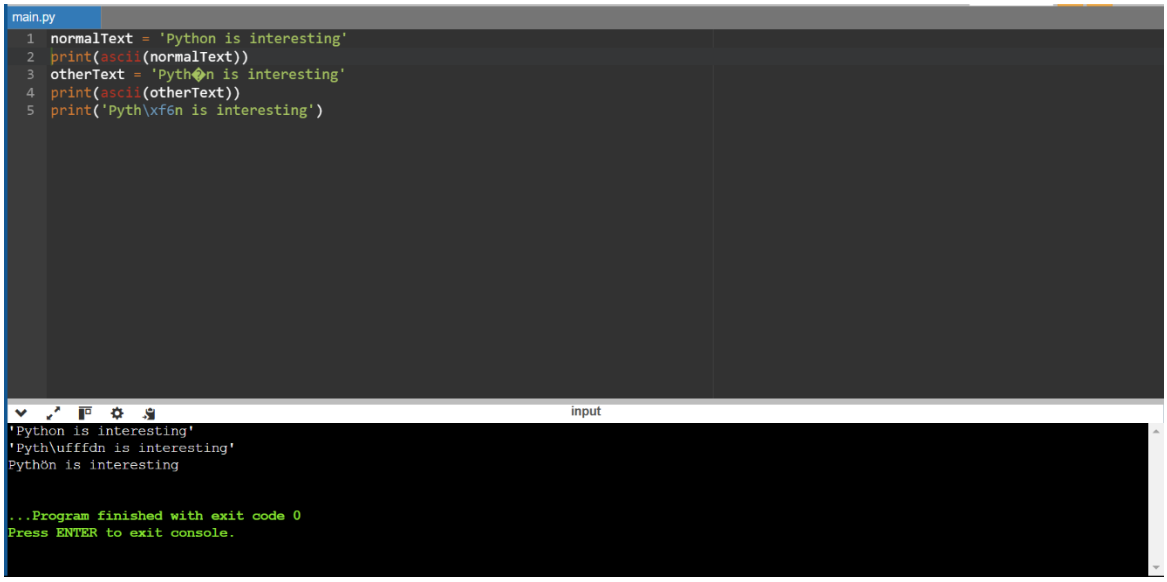


19. Python exec() Function Example

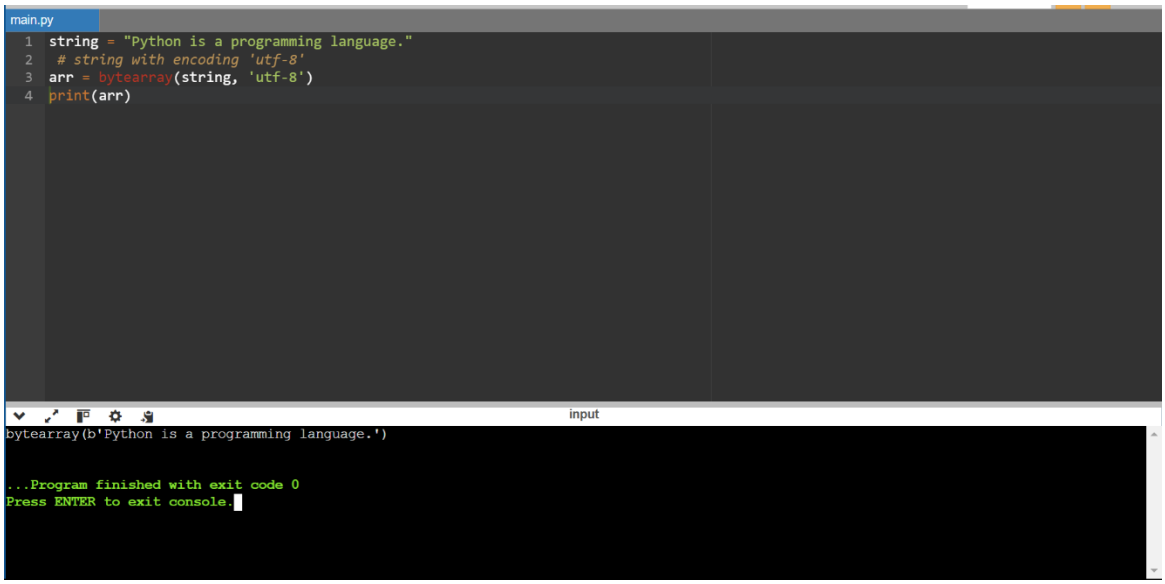
20. Python sum() Function Example

21. Python any() Function Example

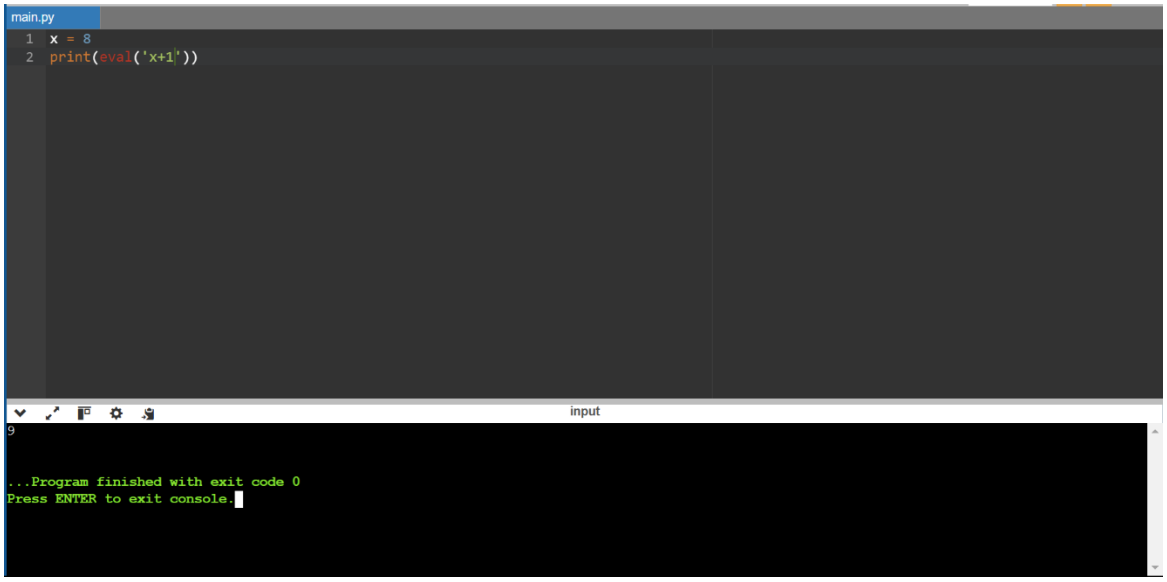
22. Python ascii() Function Example



23. Python bytearray() Example



24. Python eval() Function Example



25. Python float() Example

```
main.py
1  # for integers
2  print(float(9))
3
4  # for floats
5  print(float(8.19))
6
7  # for string floats
8  print(float("-24.27"))
9
10 # for string floats with whitespaces
11 print(float(" -17.19\n"))
12
13 # string float error
14 print(float("xyz"))
```

```
input
9.0
8.19
-24.27
-17.19
Traceback (most recent call last):
  File "/home/main.py", line 14, in <module>
    print(float("xyz"))
          ^^^^^^^^^^^^
ValueError: could not convert string to float: 'xyz'


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

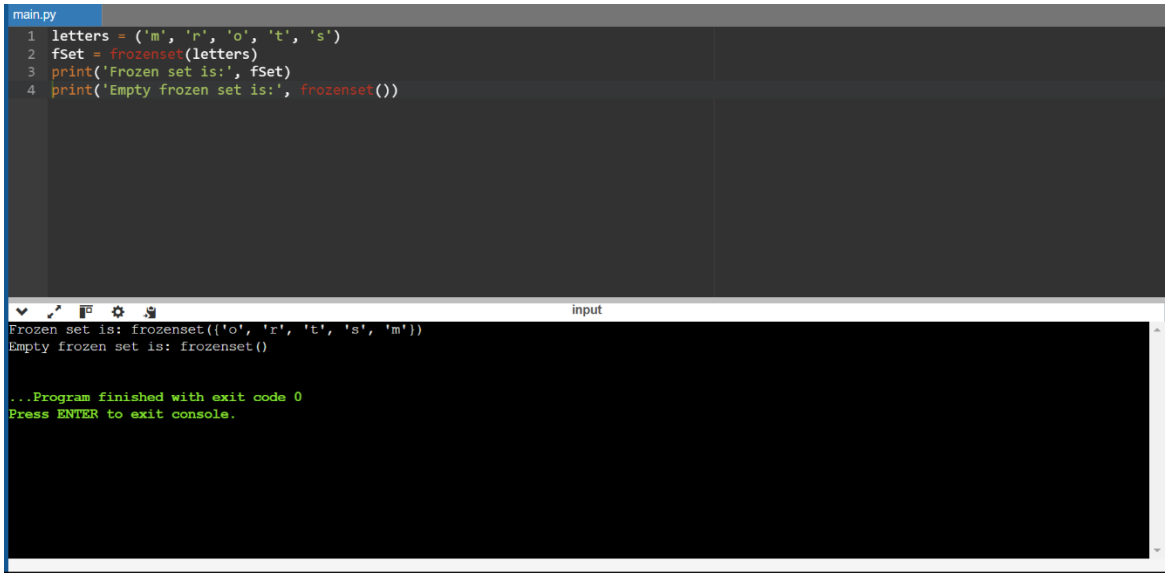
26.

26.Python format() Function

```
main.py
1  print(format(123, "d"))
2
3  # float arguments
4  print(format(123.4567898, "f"))
5
6  # binary format
7  print(format(12, "b"))
```

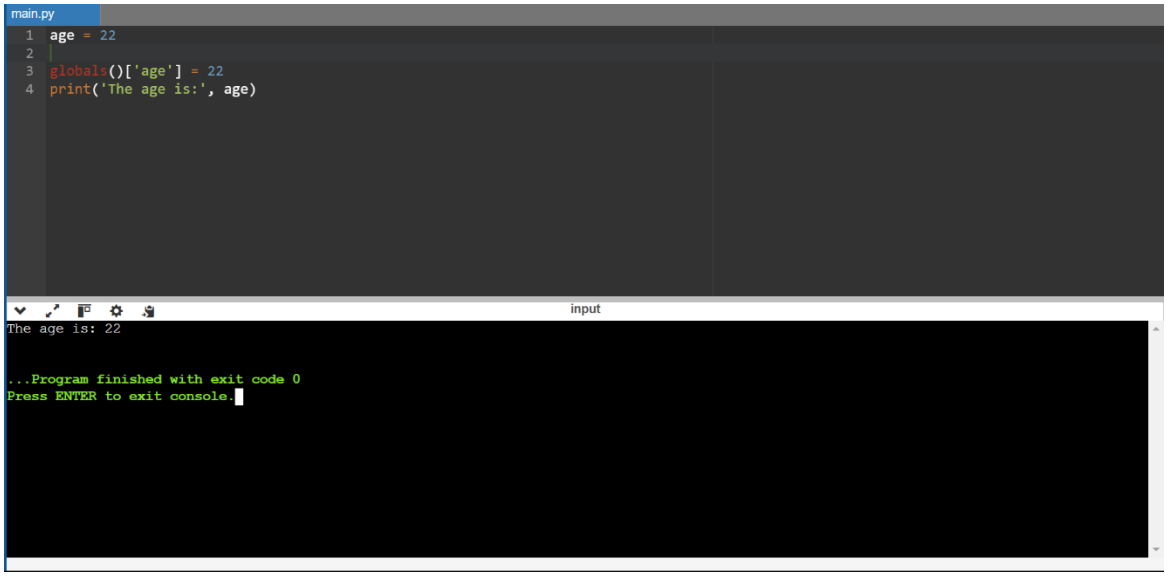
```
input
123
123.456790
1100


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

27. Python frozenset() Example

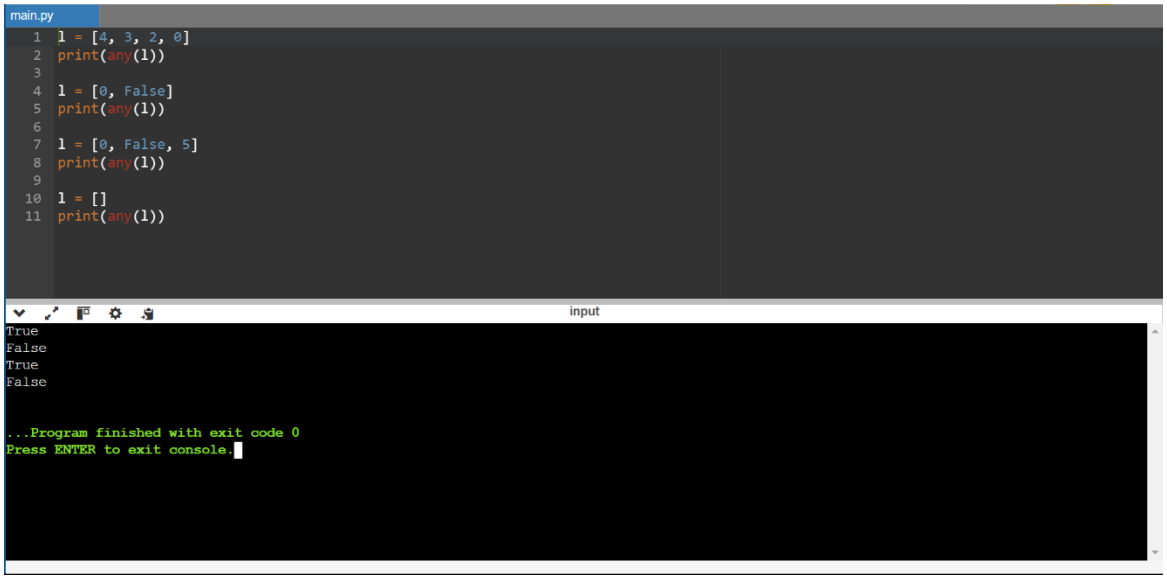


28. Python getattr() Function Example

29. Python globals() Function Example



30. Python hasattr() Function Example



31. Python iter() Function Example

32. Python len() Function Example

33. Python list() Example

main.py

```
print(list())

# string
String = 'abcde'
print(list(String))

# tuple
Tuple = (1,2,3,4,5)
print(list(Tuple))

# list
List = [1,2,3,4,5]
print(list(List))
```

input

```
[]
['a', 'b', 'c', 'd', 'e']
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5]


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

34. Python locals() Function Example

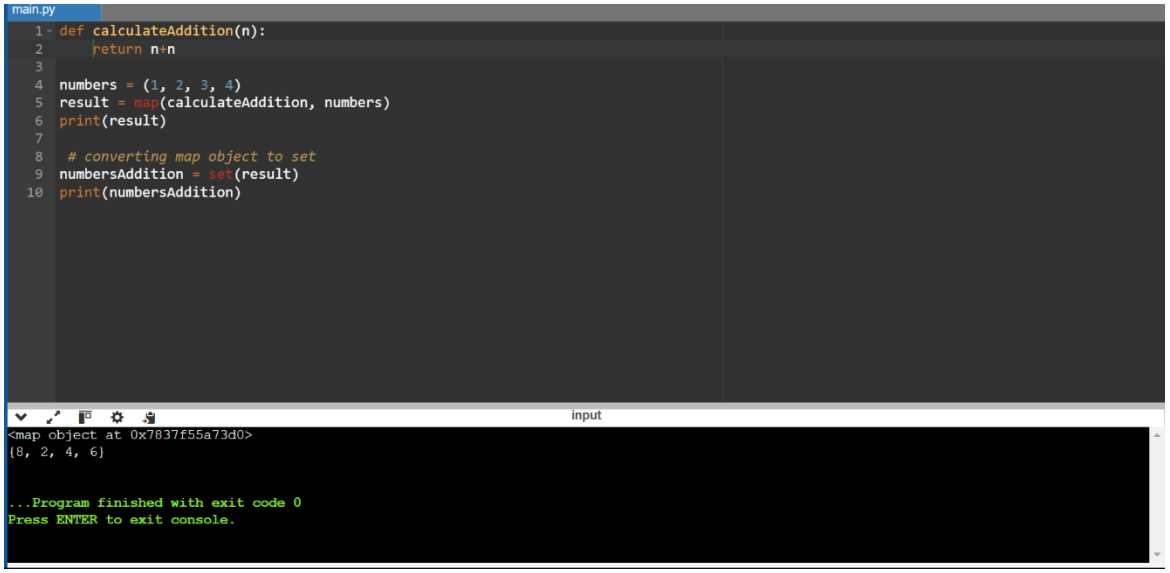
main.py

```
def localsAbsent():
    return locals()

def localsPresent():
    present = True
    return locals()

print('localsNotPresent:', localsAbsent())
print('localsPresent:', localsPresent())
```

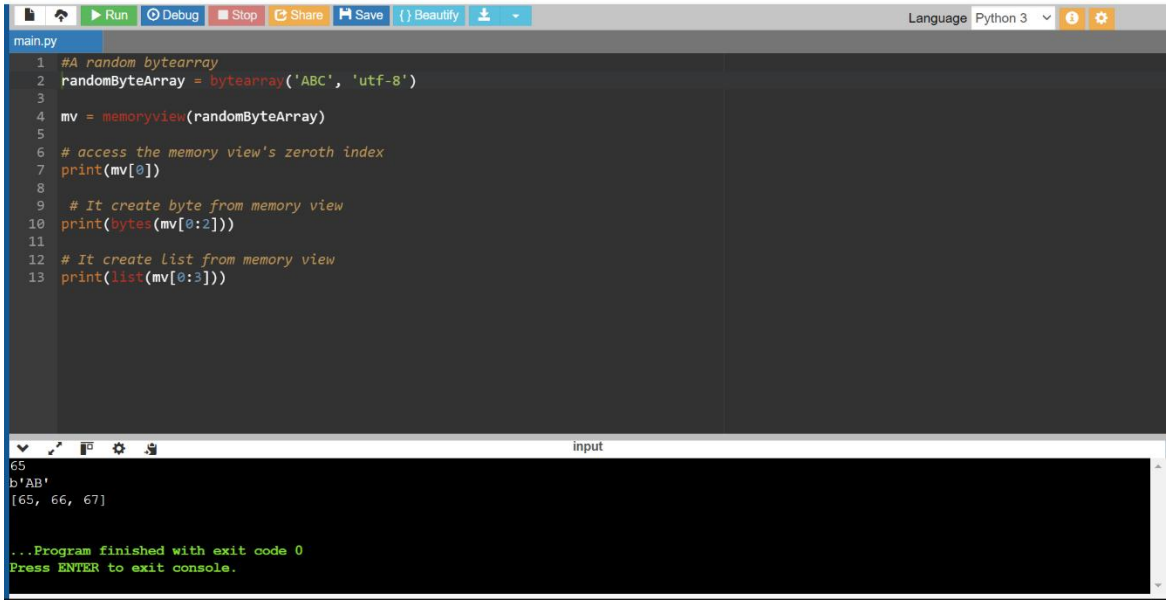
input

```
localsNotPresent: {}
localsPresent: {'present': True}


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

35. Python map() Function Example



36. Python memoryview () Function Example



37. Python object() Example

38. Python chr() Function Example

39. Python complex() Example

main.py

```
# Python complex() function example
# Calling function
a = complex(1) # Passing single parameter
b = complex(1,2) # Passing both parameters
 # Displaying result
print(a)
print(b)
```

input

```
(1+0j)
(1+2j)


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

40. Python delattr() Function Example

main.py

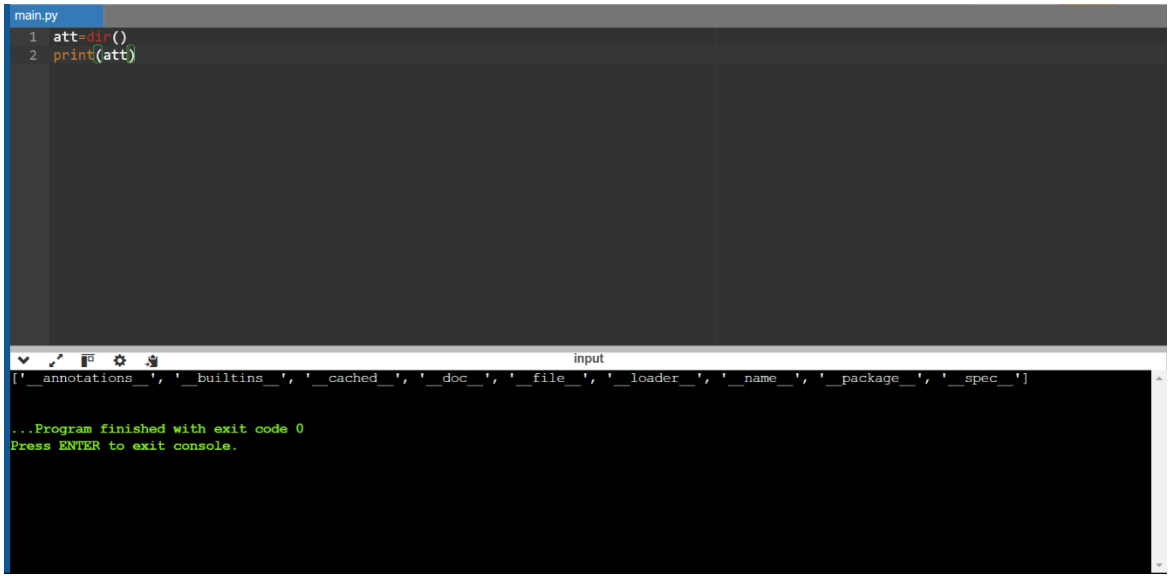
```
class Student:
    id = 101
    name = "Pranshu"
    email = "pranshu@abc.com"
 # Declaring function
    def getinfo(self):
        print(self.id, self.name, self.email)
s = Student()
s.getinfo()
delattr(Student,'course') # Removing attribute which is not available
s.getinfo() # error: throws an error
```

input

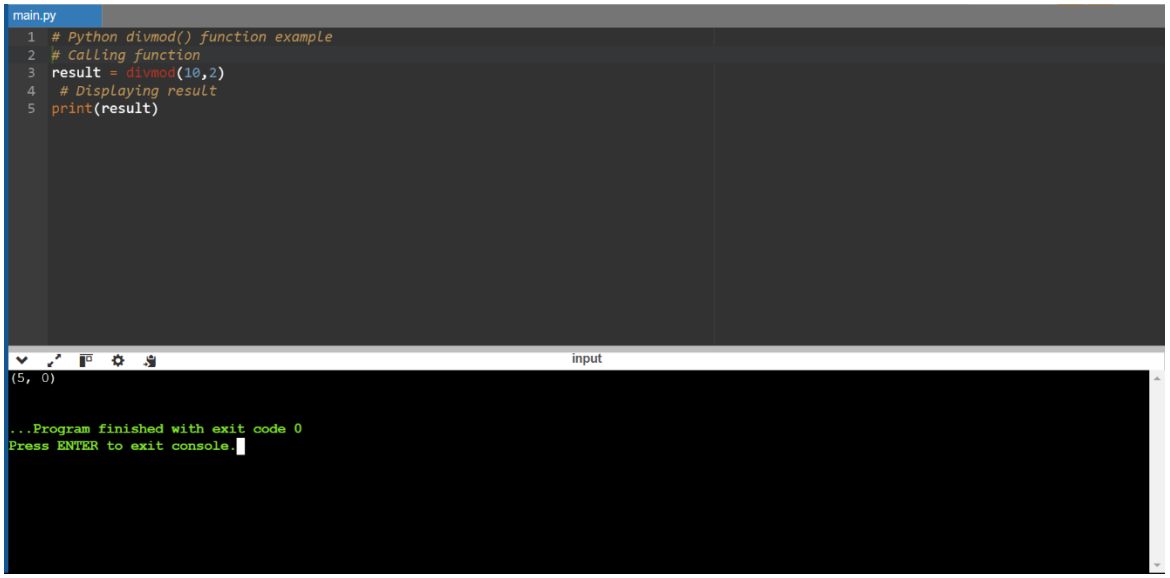
```
101 Pranshu pranshu@abc.com
Traceback (most recent call last):
  File "/home/main.py", line 10, in <module>
    delattr(Student,'course') # Removing attribute which is not available
    ^^^^^^^^^^^^^^^^^^^^^^^^^
AttributeError: type object 'Student' has no attribute 'course'


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

www.onlinegdb.com/online_python_compiler#tab-stdin

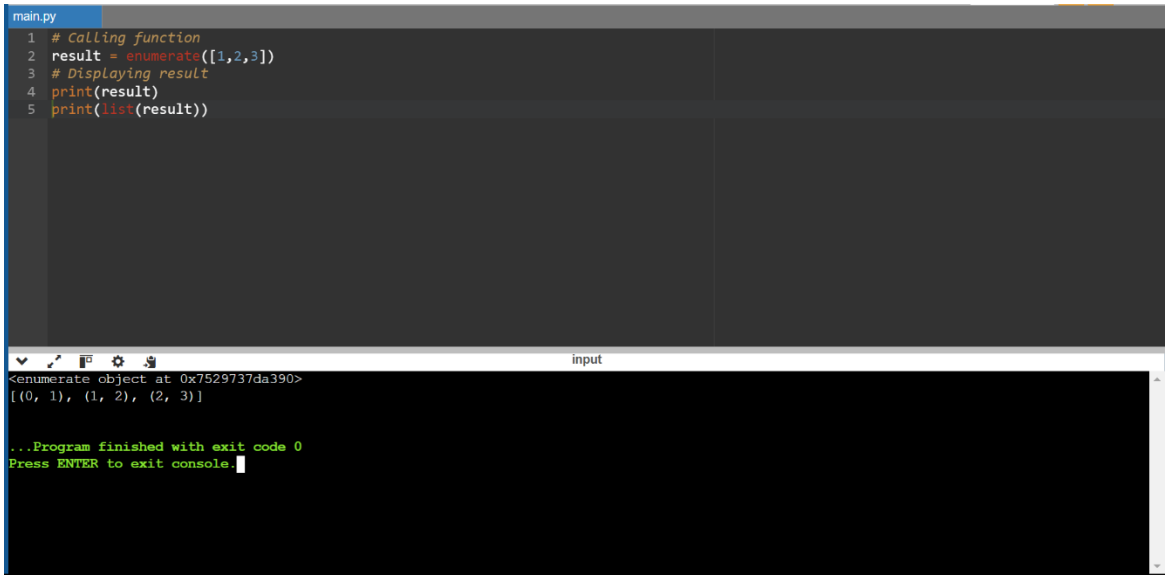
41. Python dir() Function Example



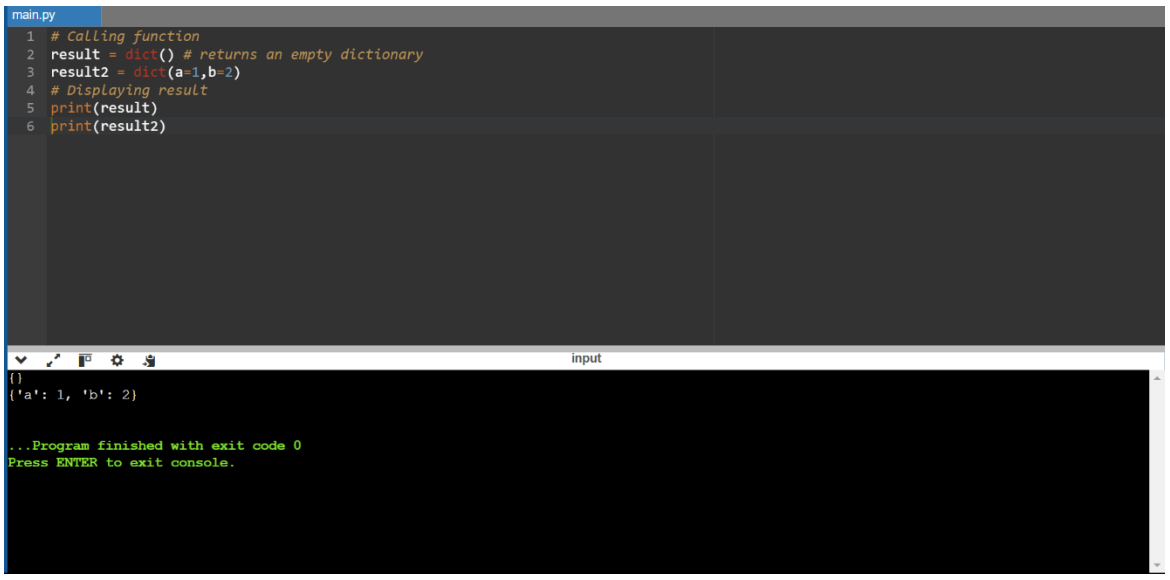
42. Python divmod() Function Example



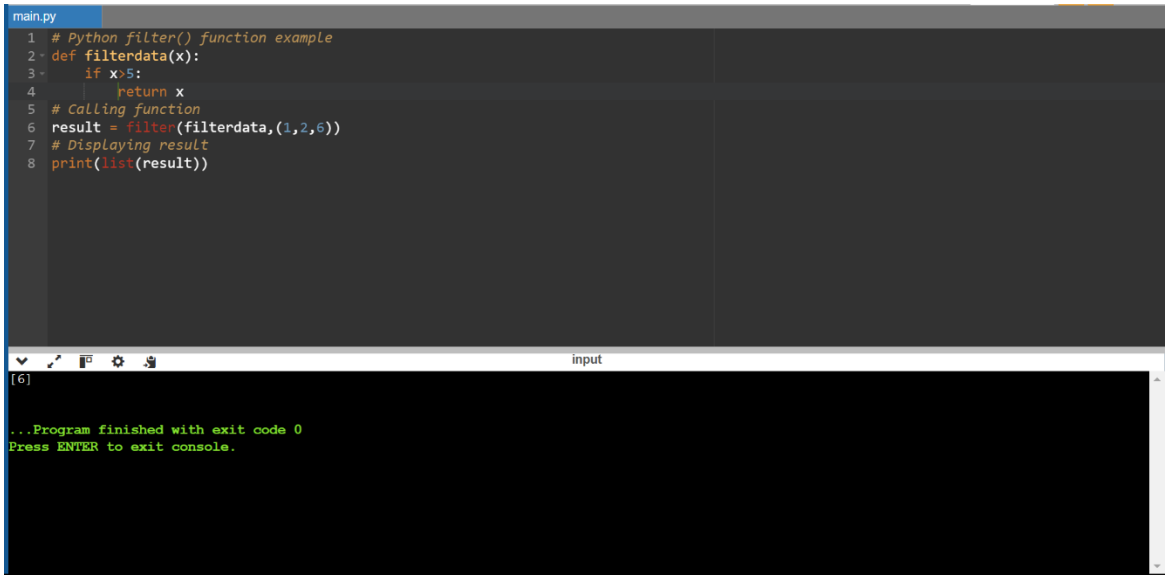
43. Python enumerate() Function Example



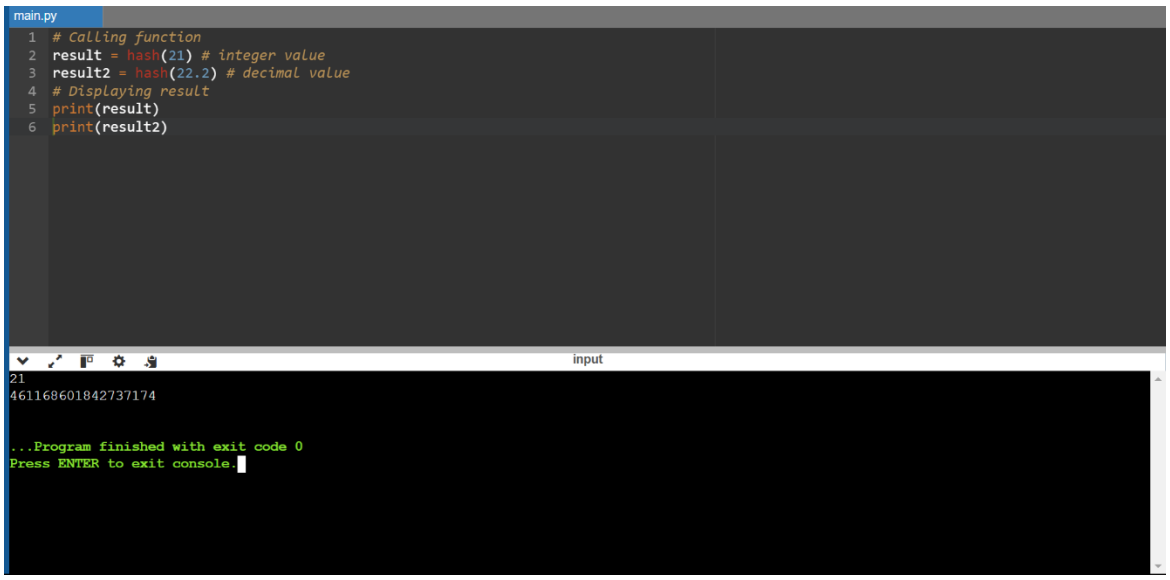
44. Python dict() Example



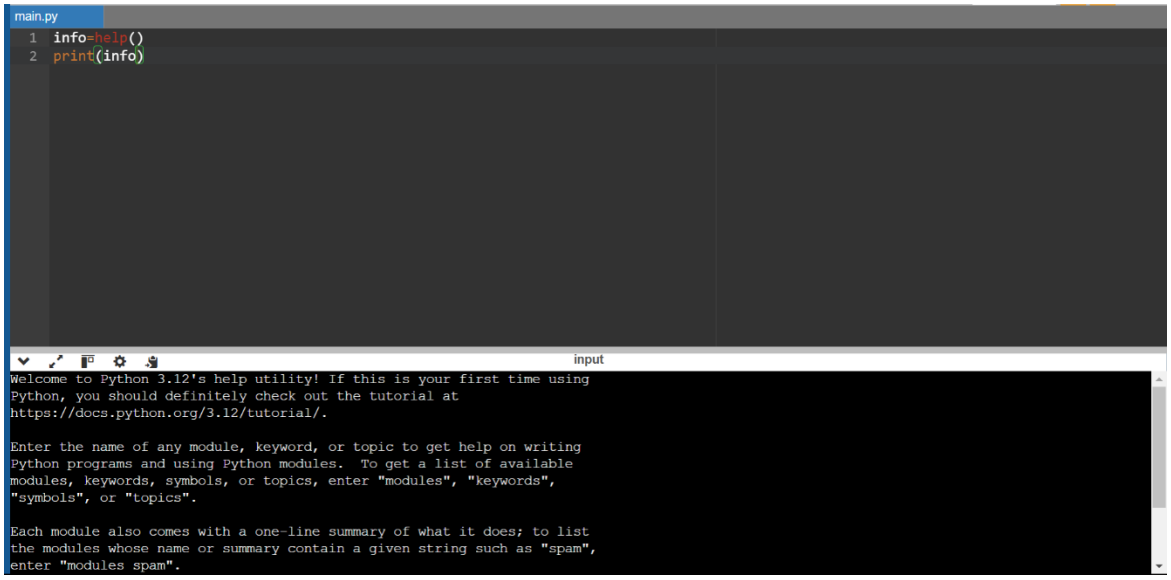
45. Python filter() Function Example



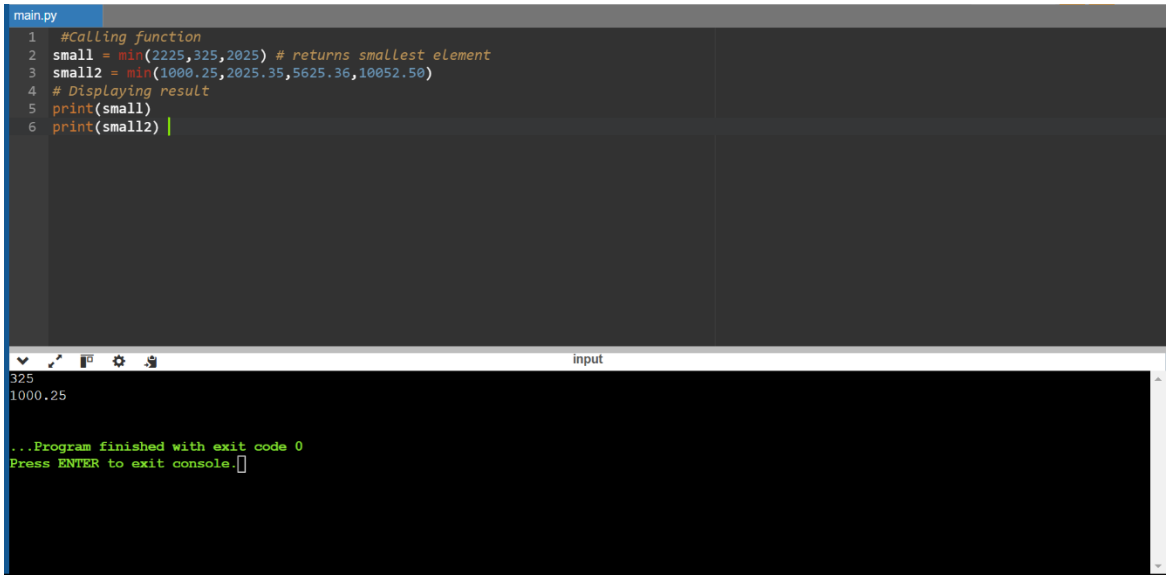
46. Python hash() Function Example



47. Python help() Function Example



48. Python min() Function Example



49. Python set() Function Example

```
main.py
1  # Calling function
2  result = set() # empty set
3  result2 = set('12')
4  result3 = set('javatpoint')
5   # Displaying result
6  print(result)
7  print(result2)
8  print(result3)
```

```
input
set()
{'1', '2'}
{'n', 'o', 'v', 'p', 'j', 'i', 'a', 't'}


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

50. Python hex() Function Example

```
main.py
1  # Calling function
2  result = hex(1)
3   # integer value
4  result2 = hex(342)
5  # Displaying result
6  print(result)
7  print(result2)
```

```
input
0x1
0x156


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

51. Python id() Function Example

main.py

```
#  Calling function
val = id("Javatpoint") # string object
val2 = id(1200) # integer object
val3 = id([25,336,95,236,92,3225]) # List object
 # Displaying result
print(val)
print(val2)
print(val3)
```

input

```
125814745936112
125814746863664
125814747466560


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

52. Python setattr() Function Example

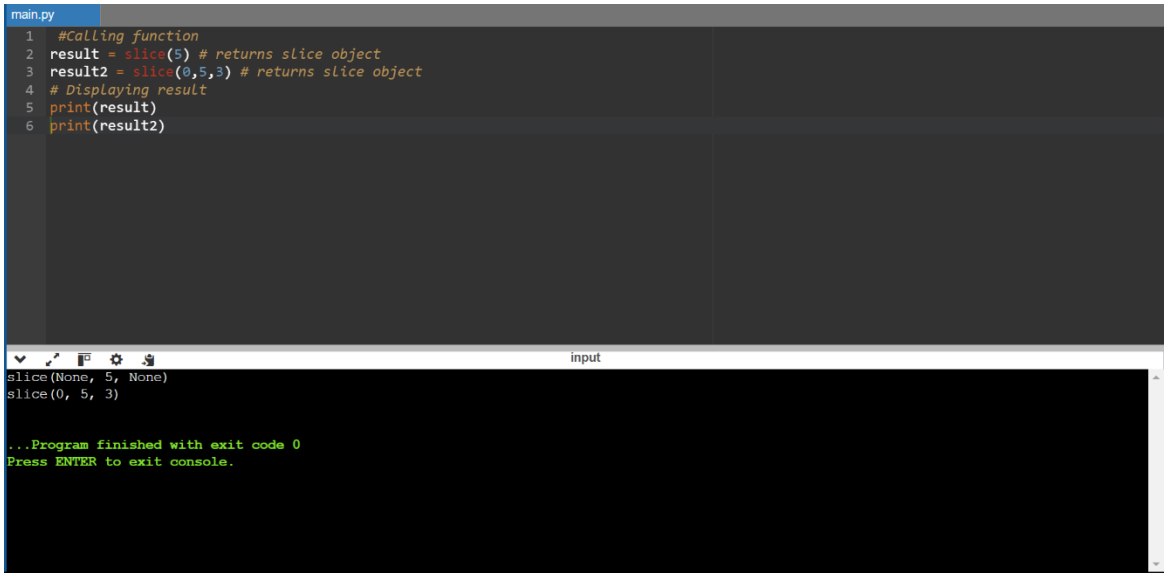
main.py

```
class Student:
    id = 0
    name = ""

    def __init__(self, id, name):
        self.id = id
        self.name = name

student = Student(102,"Sohan")
print(student.id)
print(student.name)
#print(student.email) product error
setattr(student, 'email','sohan@abc.com') # adding new attribute
print(student.email)
```

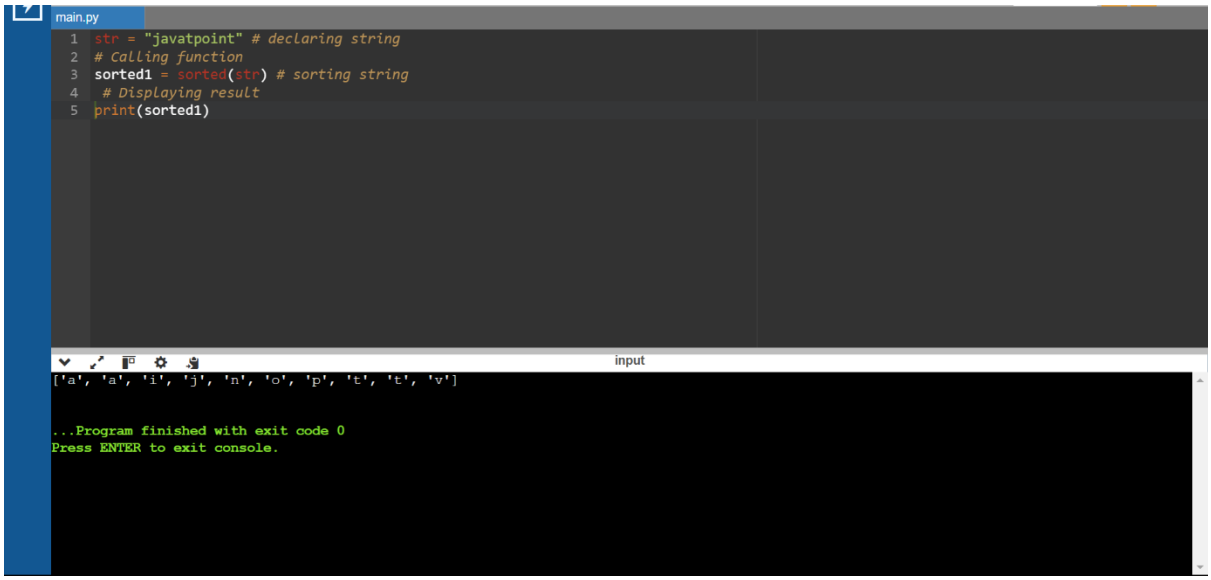
input

```
102
Sohan
sohan@abc.com


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

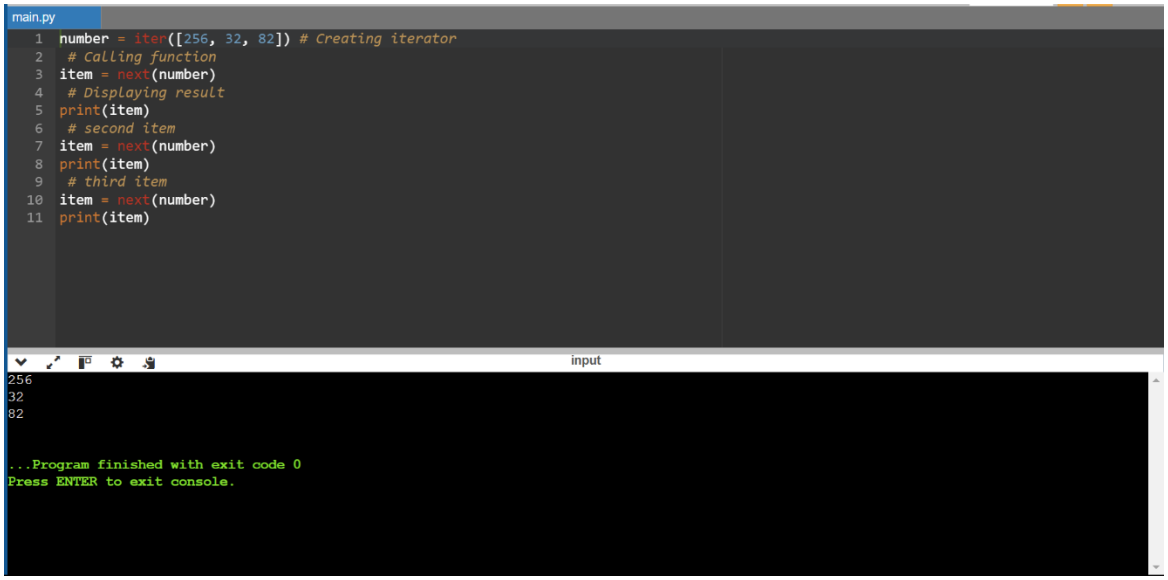
53. Python slice() Function Example



54. Python sorted() Function Example

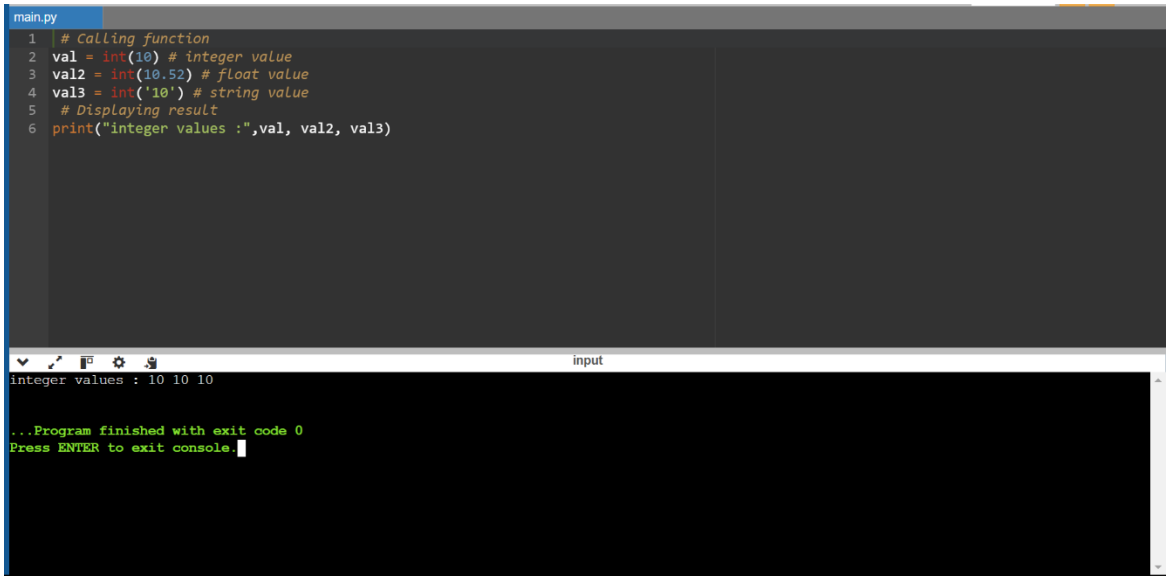


55. Python next() Function Example

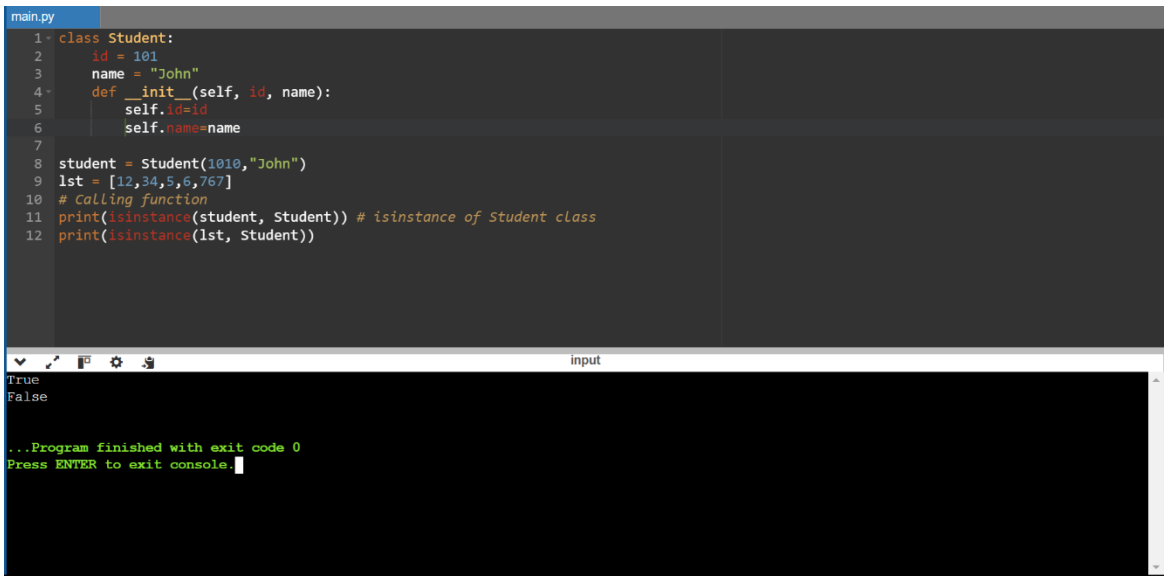


56. Python input() Function Example

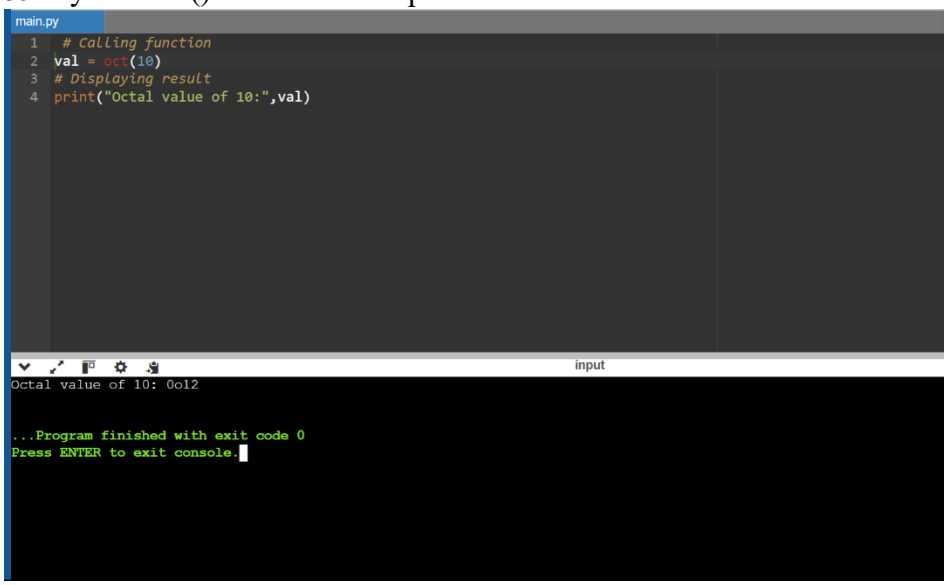
57. Python int() Function Example



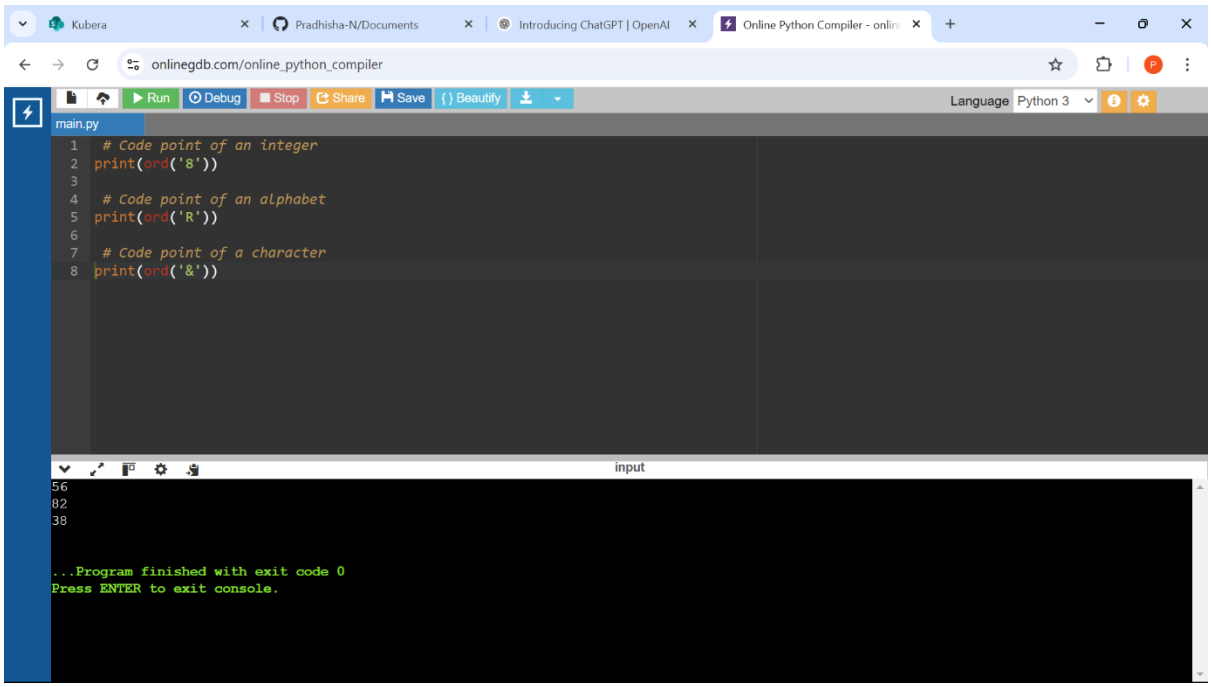
58. Python isinstance() function Example



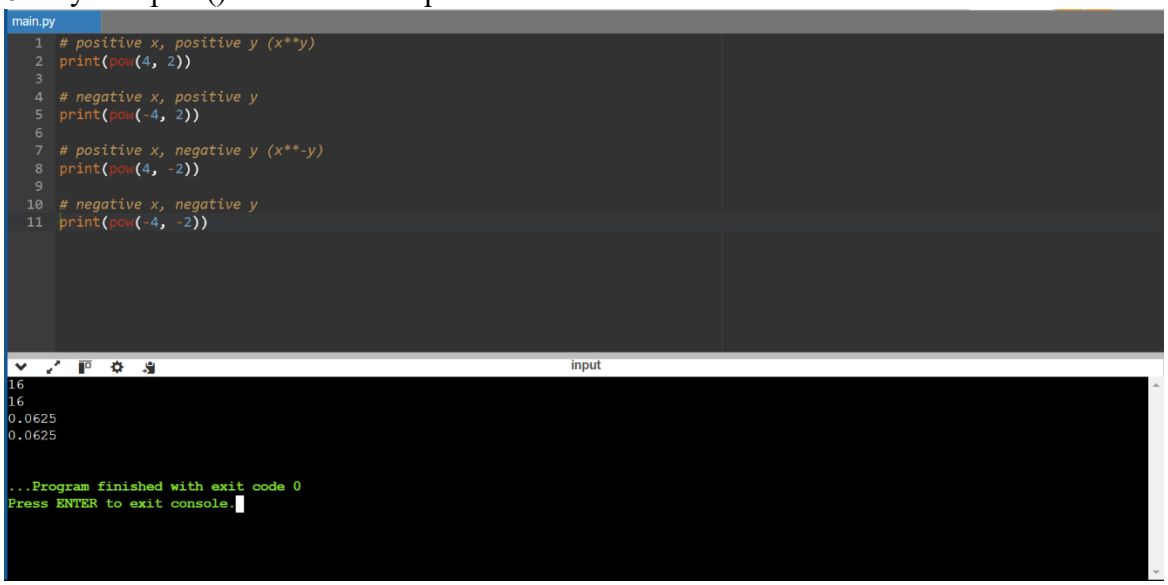
59. Python oct() function Example



60. Python ord() function Example



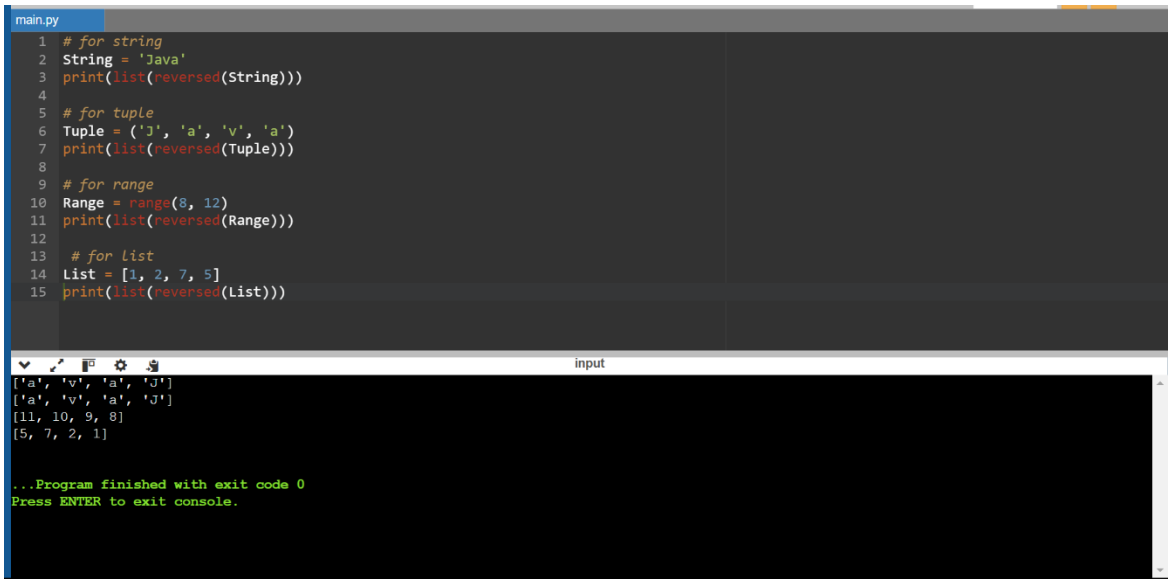
61. Python pow() function Example



62. Python print() function Example

63. Python range() function Example

64. Python reversed() function Example



65. Python round() Function Example

main.py

```
# for integers
print(round(10))

# for floating point
print(round(10.8))

# even choice
print(round(6.6))
```

input

```
10
11
7


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

66. Python issubclass() Function Example

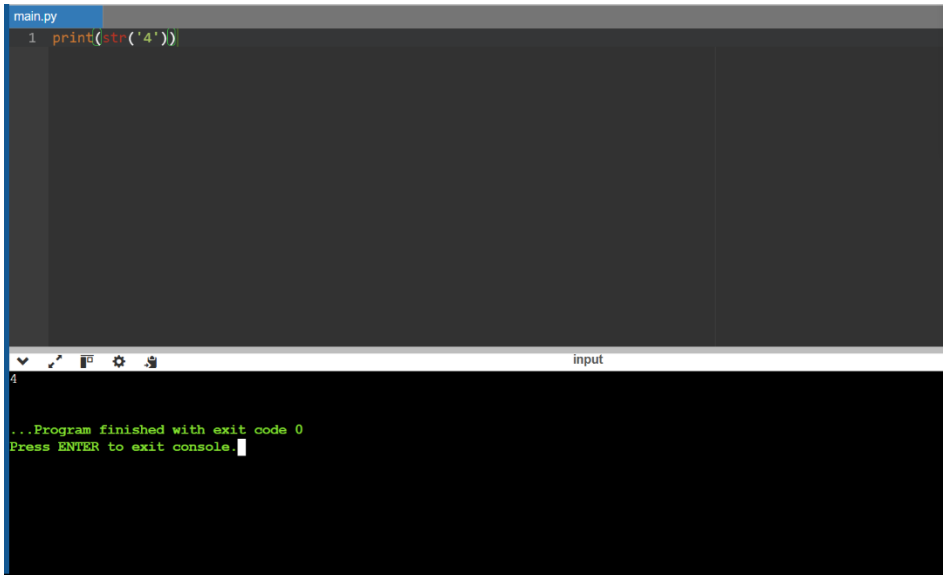
main.py

```
class Rectangle:
    def __init__(rectangleType):
        print('Rectangle is a ', rectangleType)

class Square(Rectangle):
    def __init__(self):
        Rectangle.__init__('square')

print(issubclass(Square, Rectangle))
print(issubclass(Square, list))
print(issubclass(Square, (list, Rectangle)))
print(issubclass(Rectangle, (list, Rectangle)))
```

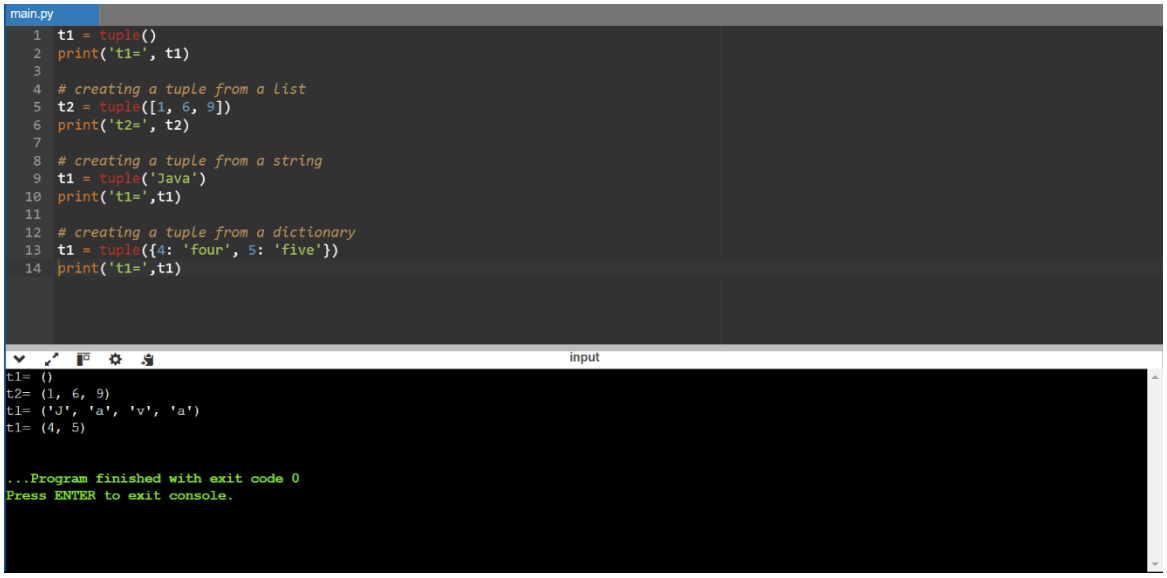
input

```
True
False
True
True


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

67. Python str() Function Example

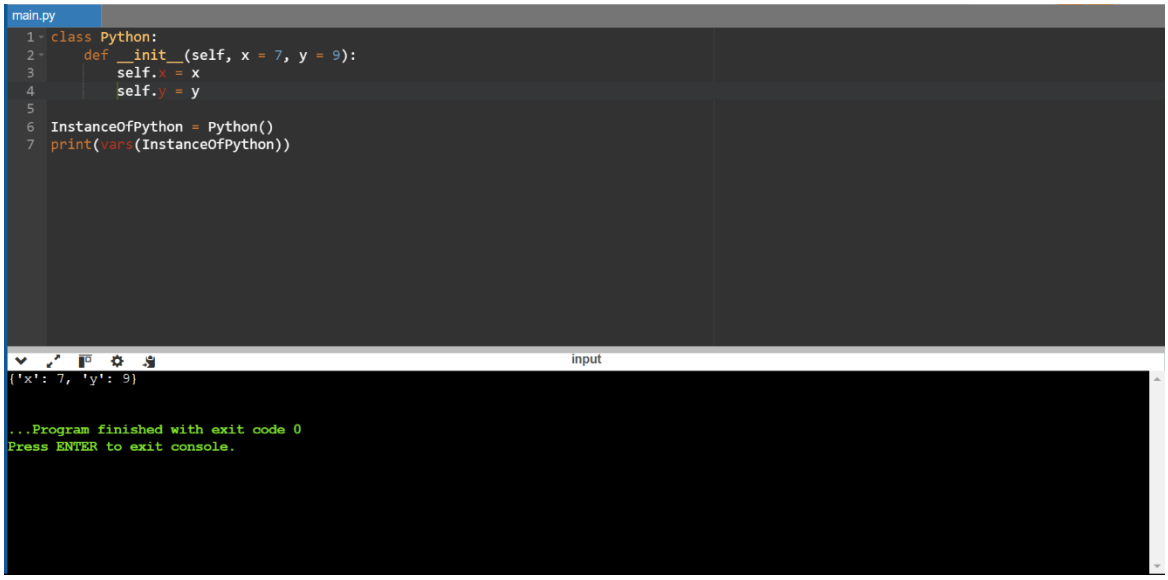


68. Python tuple() Function Example

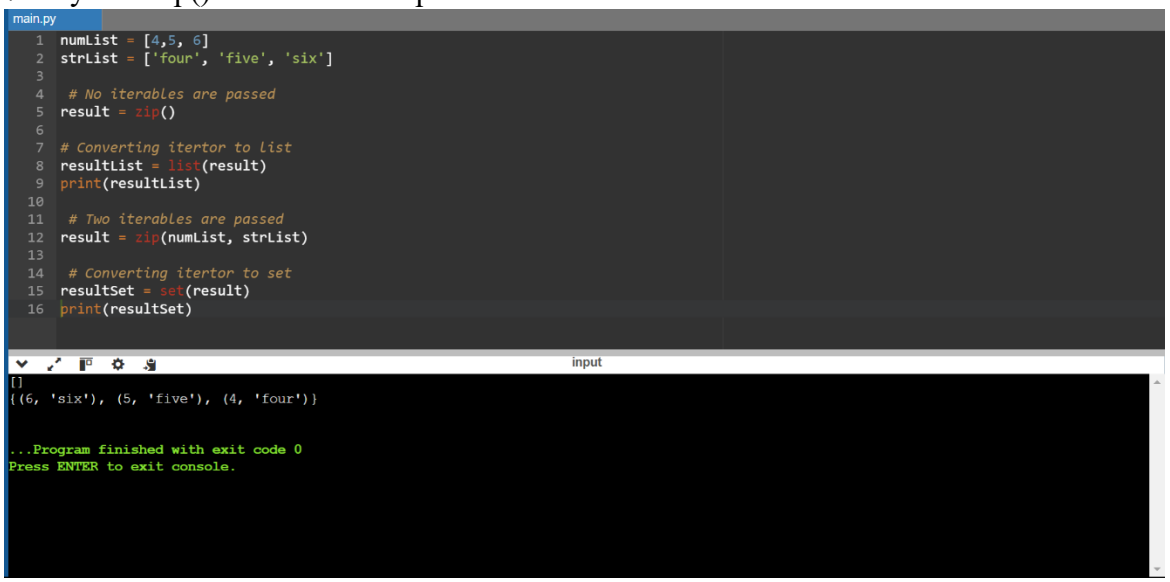


69. Python type() Function Example

70. Python vars() Function Example



71. Python zip() Function Example

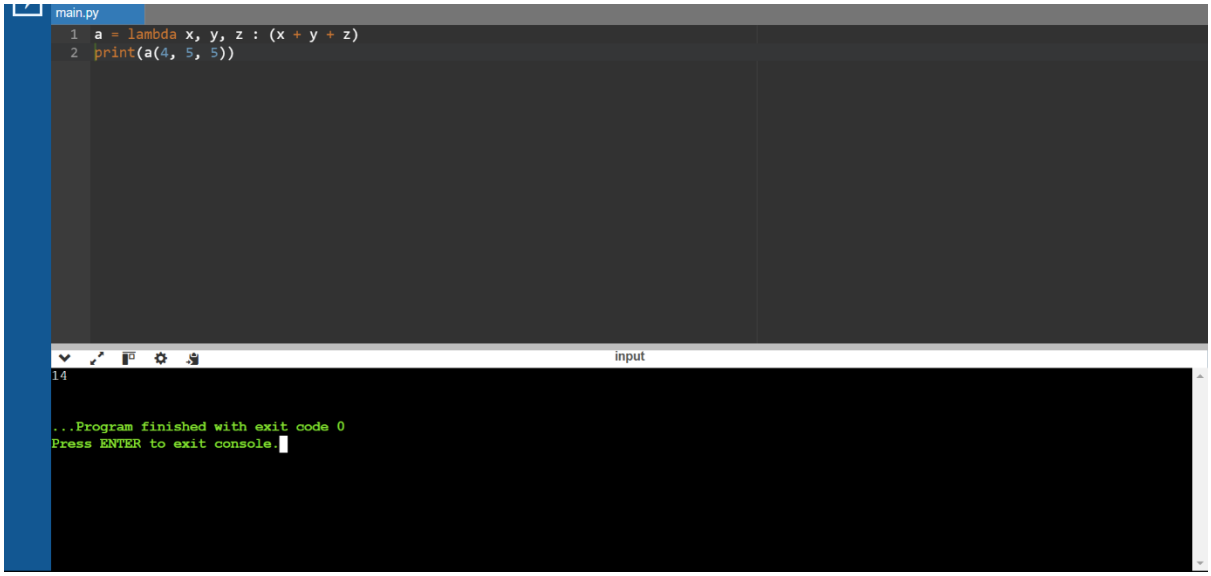


72. Code to demonstrate how we can use a lambda function for adding 4 numbers

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.

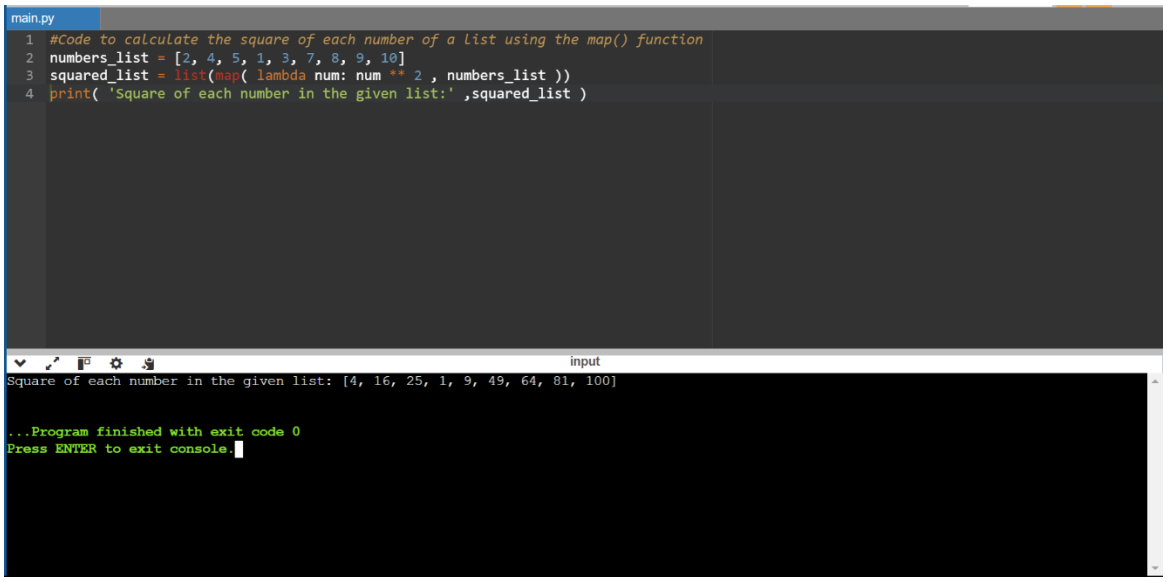
75. another example of a lambda function that adds 2 numbers and return one result.



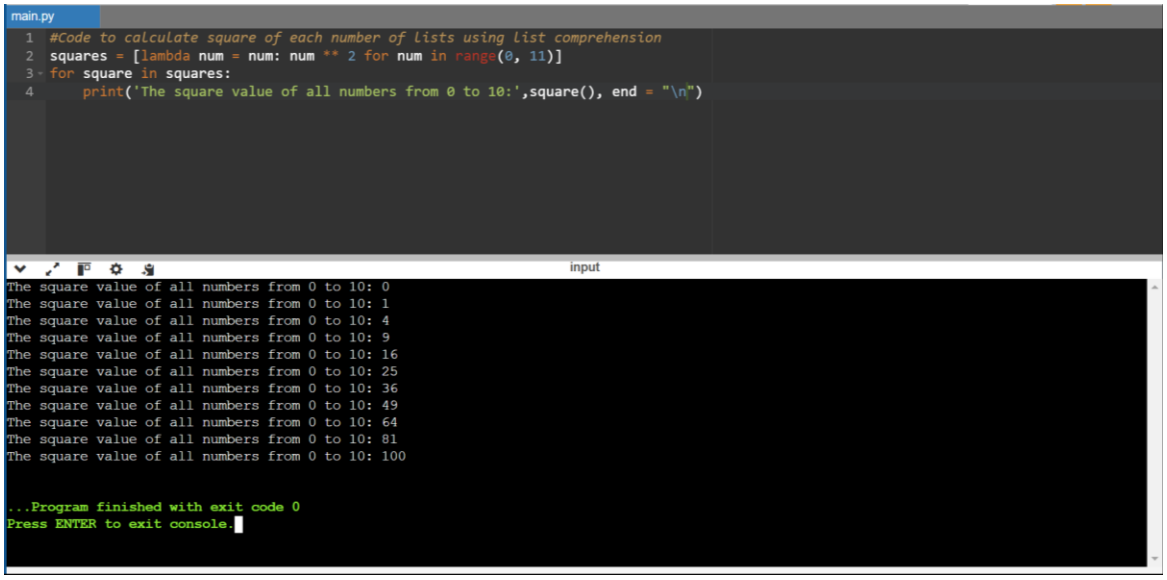
76. Python code to show the reciprocal of the given number to highlight the difference between def() and lambda().

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

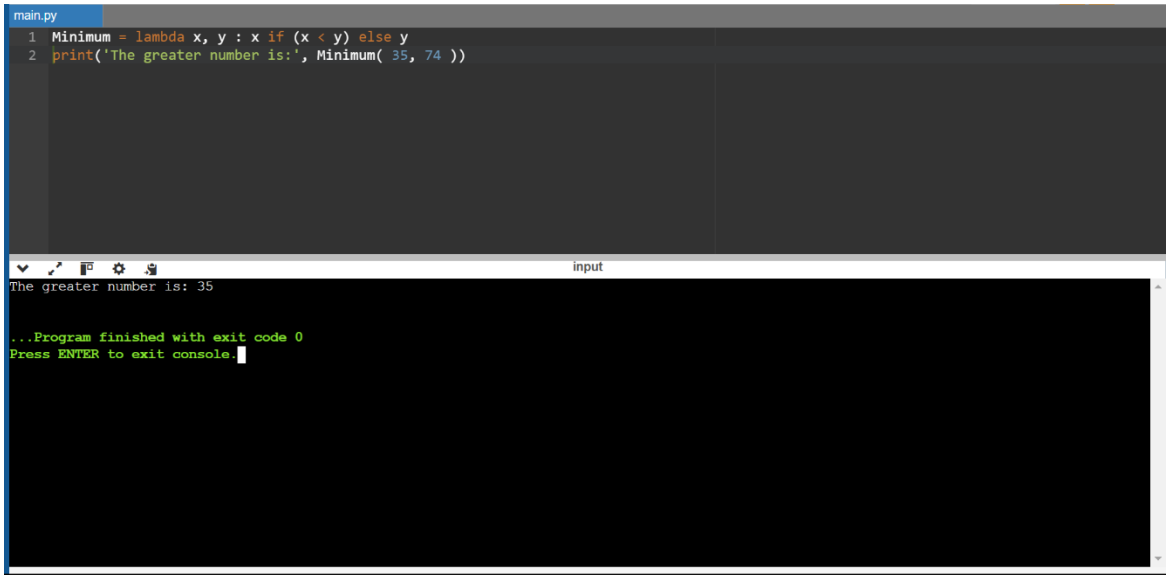
78. Code to calculate the square of each number of a list using the map() function



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

```
main.py
# Code to print the third largest number of the given list using the lambda function

my_List = [ [3, 5, 8, 6], [23, 54, 12, 87], [1, 2, 4, 12, 5] ]
 # sorting every sublist of the above list
sort_List = lambda num : ( sorted(n) for n in num )
 # Getting the third largest number of the sublist
third_Largest = lambda num, func : [ l[ len(l) - 2] for l in func(num)]
result = third_Largest( my_List, sort_List)
print('The third largest number from every sub list is:', result )
```

```
input
The third largest number from every sub list is: [6, 54, 5]


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

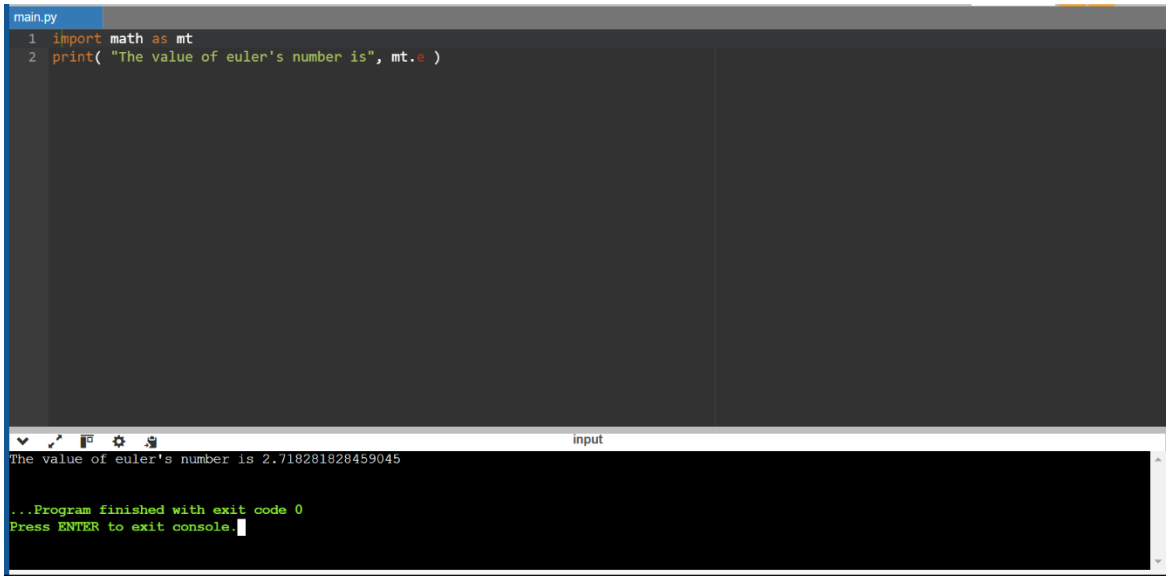
MODULES

82. Python import Statement

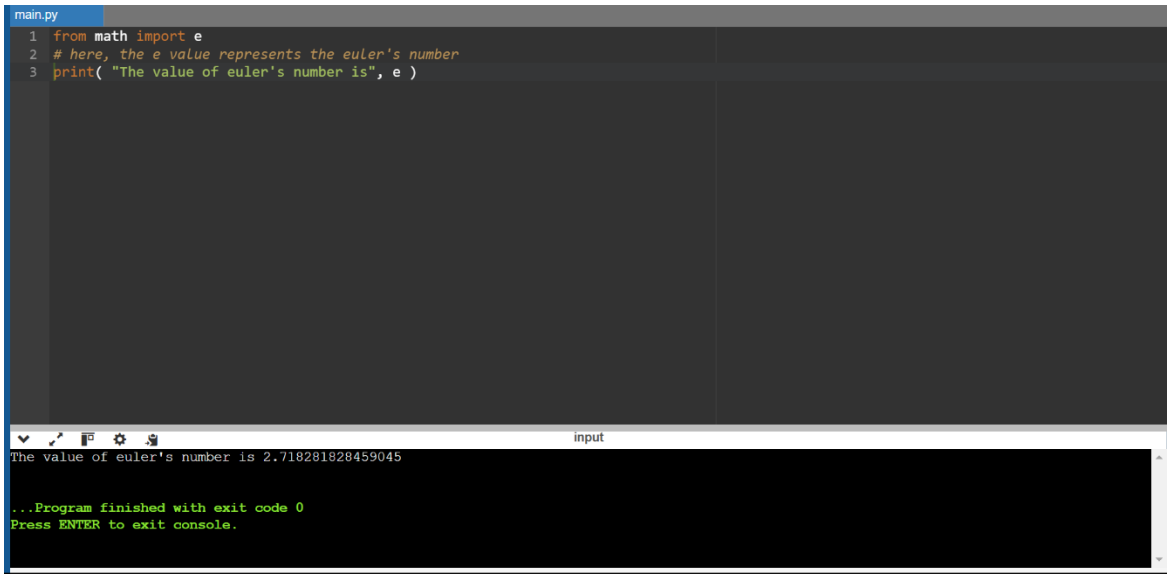
```
main.py
import math
print( "The value of euler's number is", math.e )
```

```
input
The value of euler's number is 2.718281828459045


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

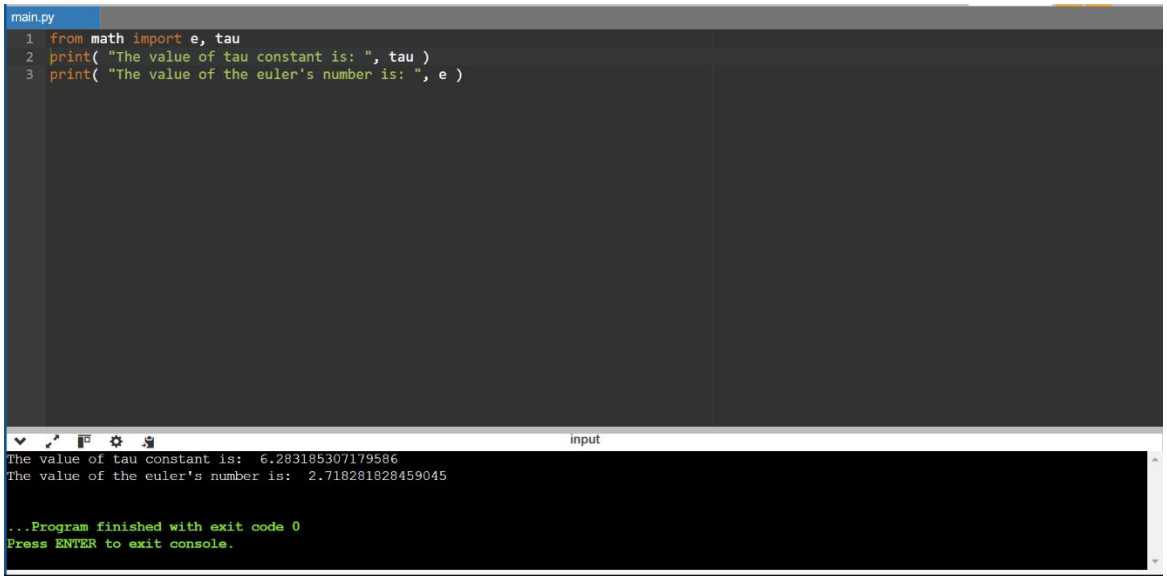
83. Importing and also Renaming



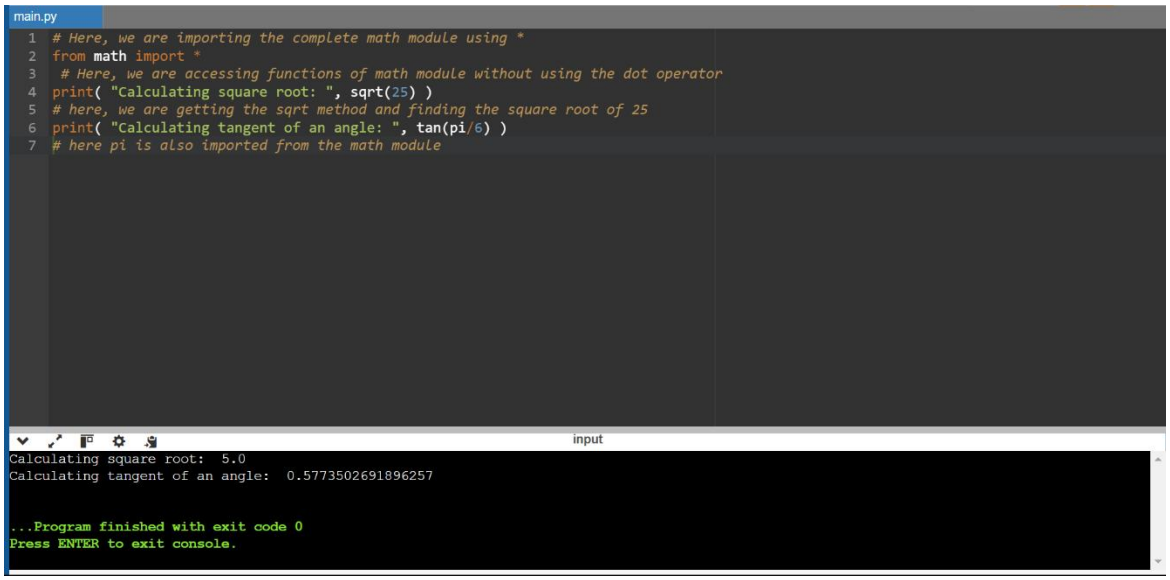
84. Python from...import Statement



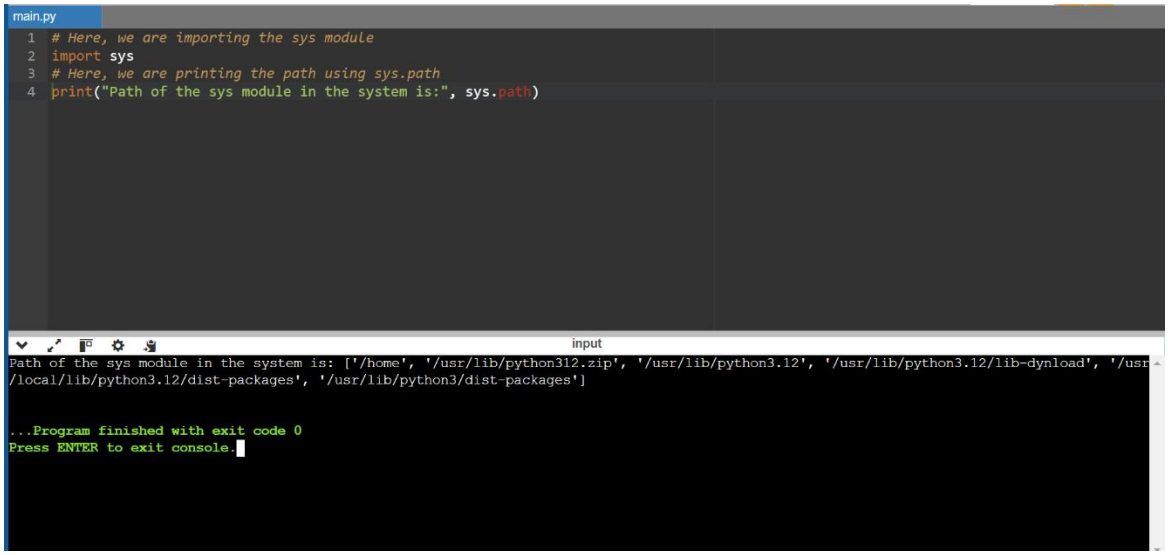
85. we are creating a simple Python program to show how to import multiple



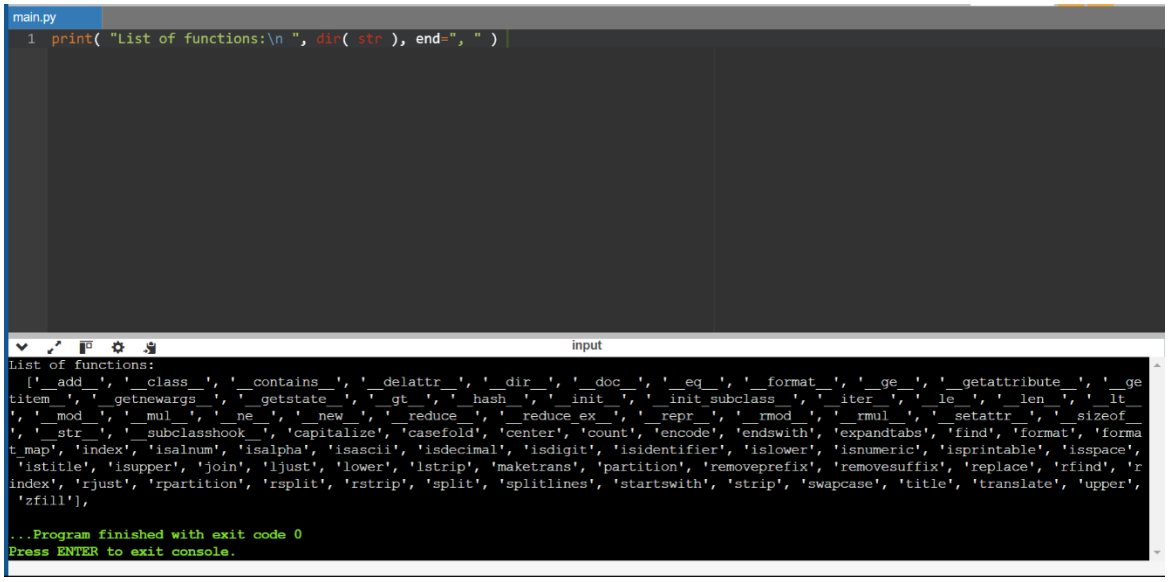
86. we are importing the complete math module using *



87. Example to print the path.



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



89. Namespaces and scoping.

```
main.py
1  Number = 204
2  def AddNumber(): # here, we are defining a function with the name Add Number
3  # Here, we are accessing the global namespace
4      global Number
5      Number = Number + 200
6  print("The number is:", Number)
7  # here, we are printing the number after performing the addition
8  AddNumber() # here, we are calling the function
9  print("The number is:", Number)
```

```
input
The number is: 204
The number is: 404


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

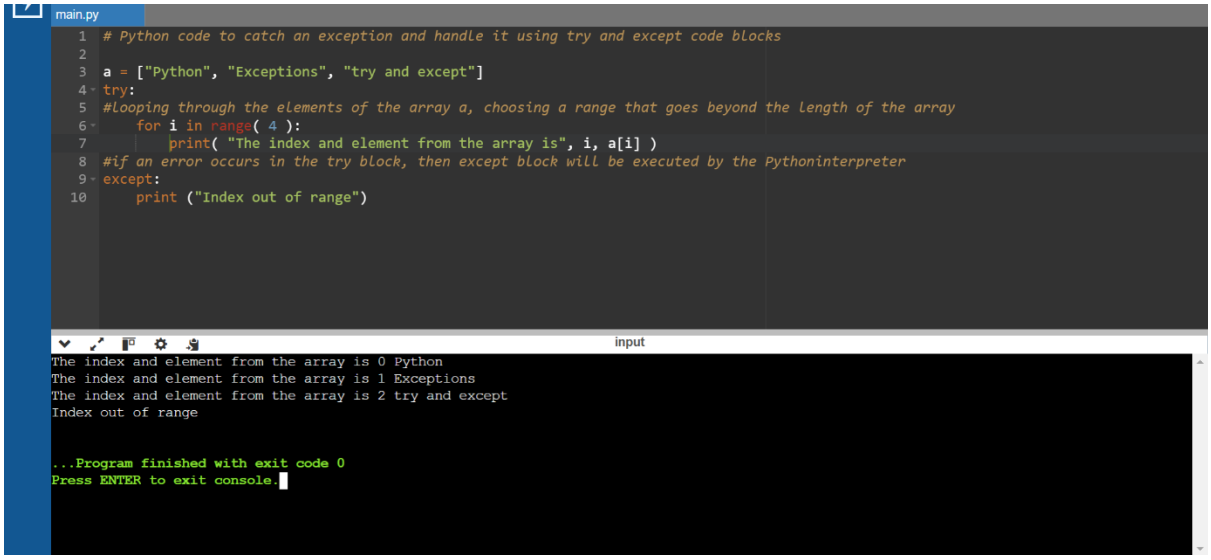
EXCEPTION

90. Exceptions versus Syntax Errors.

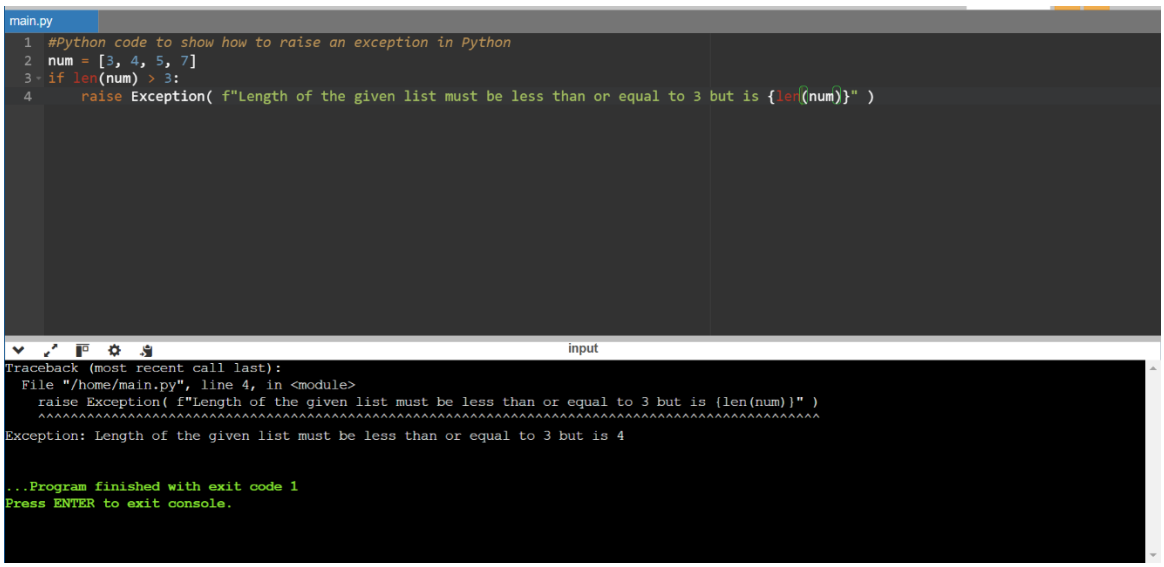
```
main.py
1   #Python code after removing the syntax error
2  string = "Python Exceptions"
3
4  for s in string:
5      if (s != o:
6          print( s )
```

```
input
  File "/home/main.py", line 5
    if (s != o:
            ^
SyntaxError: invalid syntax


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

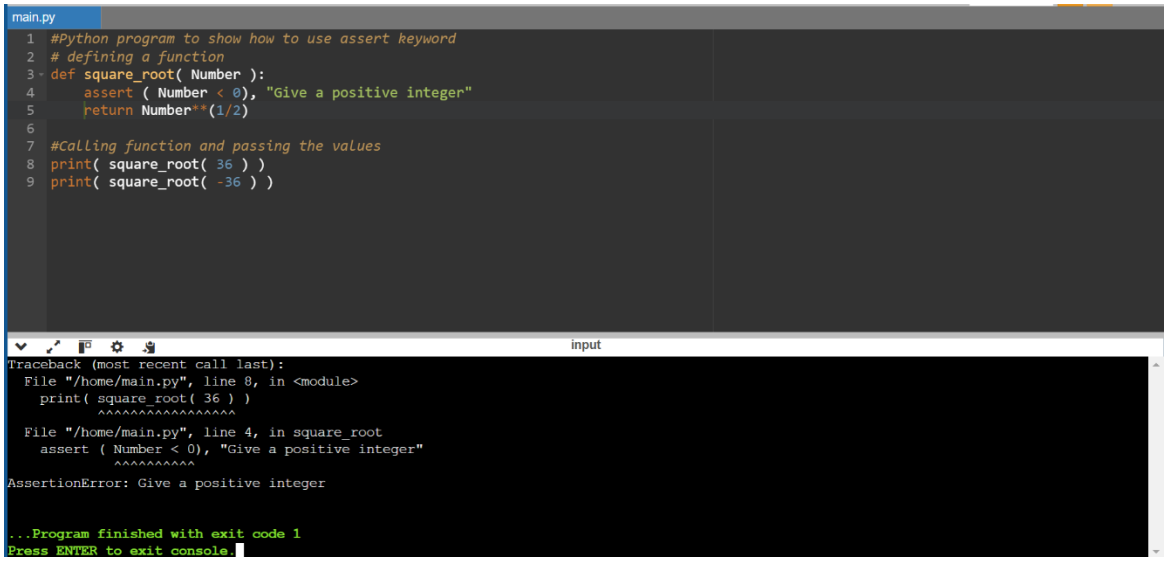
91. Python code to catch an exception and handle it using try and except code blocks.



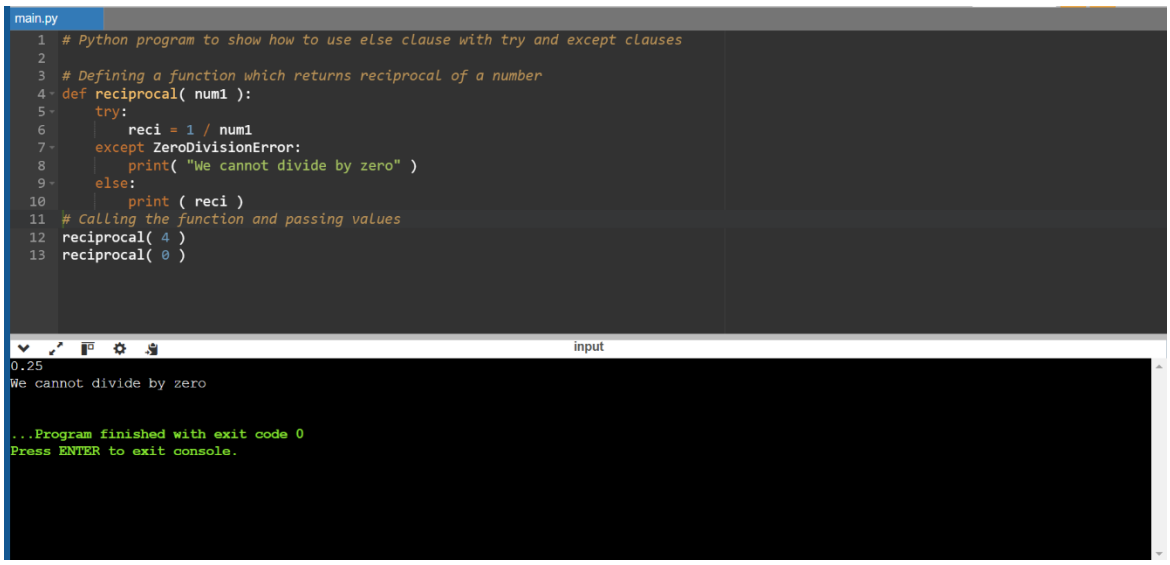
92. Python code to show how to raise an exception in Python



93. The assert Statement



94. Try with Else Clause



95. Finally Keyword in Python

main.py

```
# Python code to show the use of finally clause

# Raising an exception in try block
try:
    div = 4 // 0
    print( div )
# this block will handle the exception raised
except ZeroDivisionError:
    print( "Atepting to divide by zero" )
# this will always be executed no matter exception is raised or not
finally:
    print( 'This is code of finally clause' )
```

input

```
Atepting to divide by zero
This is code of finally clause


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

96. User-Defined Exceptions

main.py

```
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 )
```

input

```
  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

main.py

```
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])
```

input

```
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?

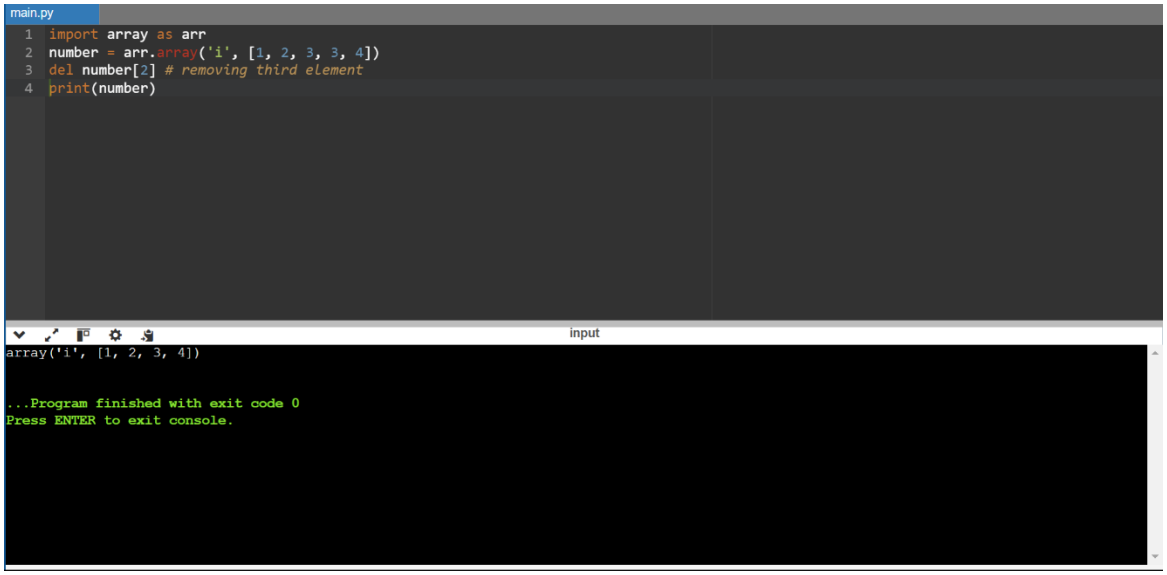
main.py

```
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)
```

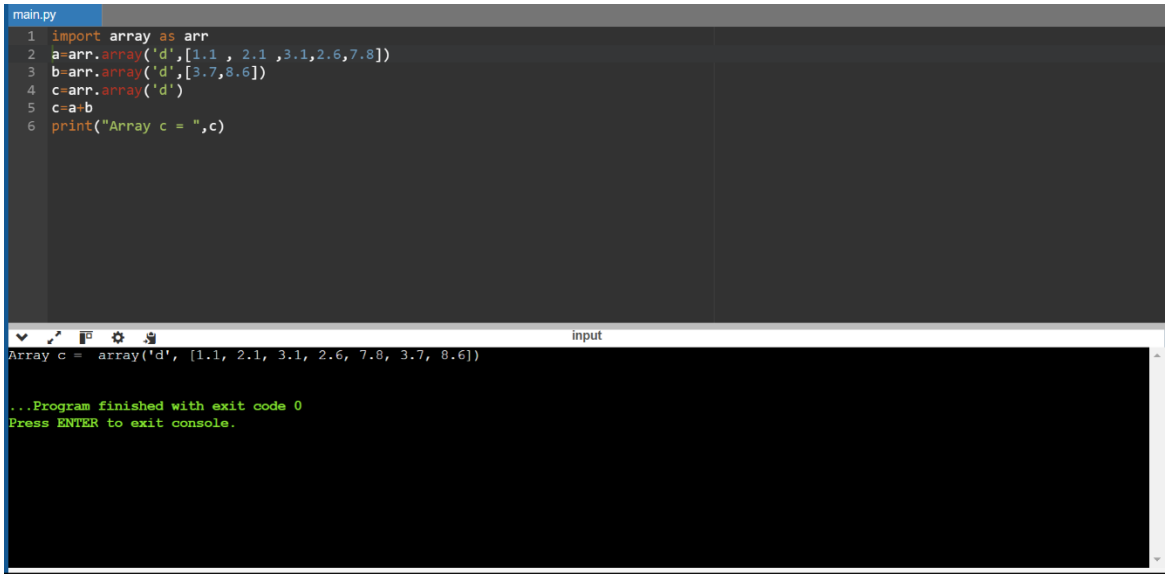
input

```
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

```
main.py
1  import array as arr
2  x = arr.array('i', [4, 7, 19, 22]) # Initialize the array elements
3  print("First element:", x[0])
4  print("Second element:", x[1])
5  print("Second last element:", x[-1])
```

input

```
First element: 4
Second element: 7
Second last element: 22


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

DECORATORS

102. Sample example

```
main.py
1  def func1(msg):
2      print(msg)
3  func1("Hii, welcome to function ")
4  func2 = func1
5  func2("Hii, welcome to function ")
```

input

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


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

103. Inner Function

main.py

```
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()
```

input

```
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

main.py

```
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))
```

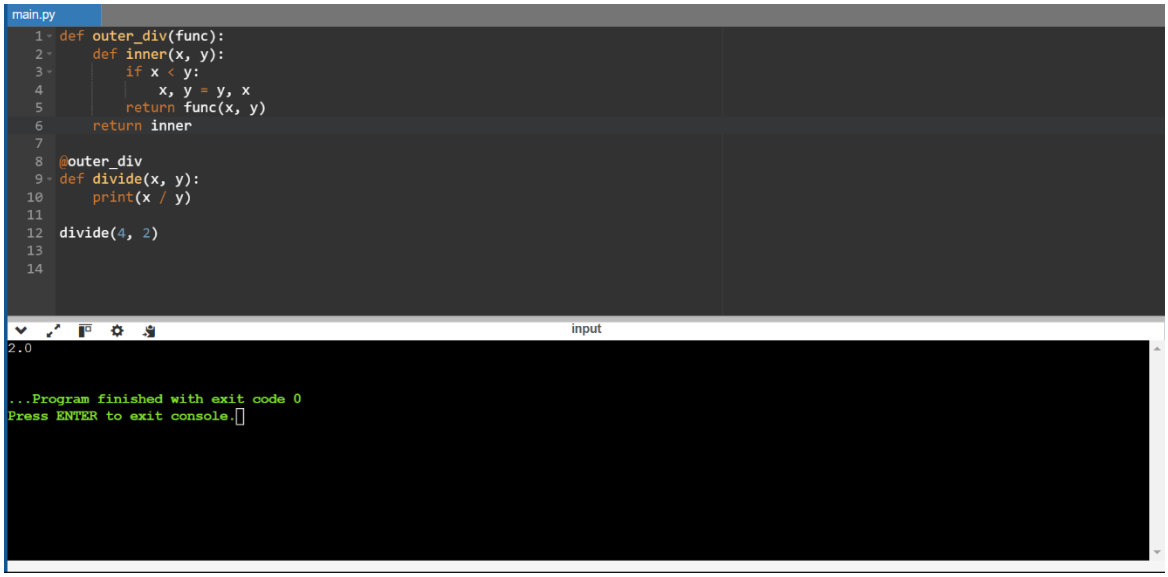
input

```
9
21


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

105. Inner Function

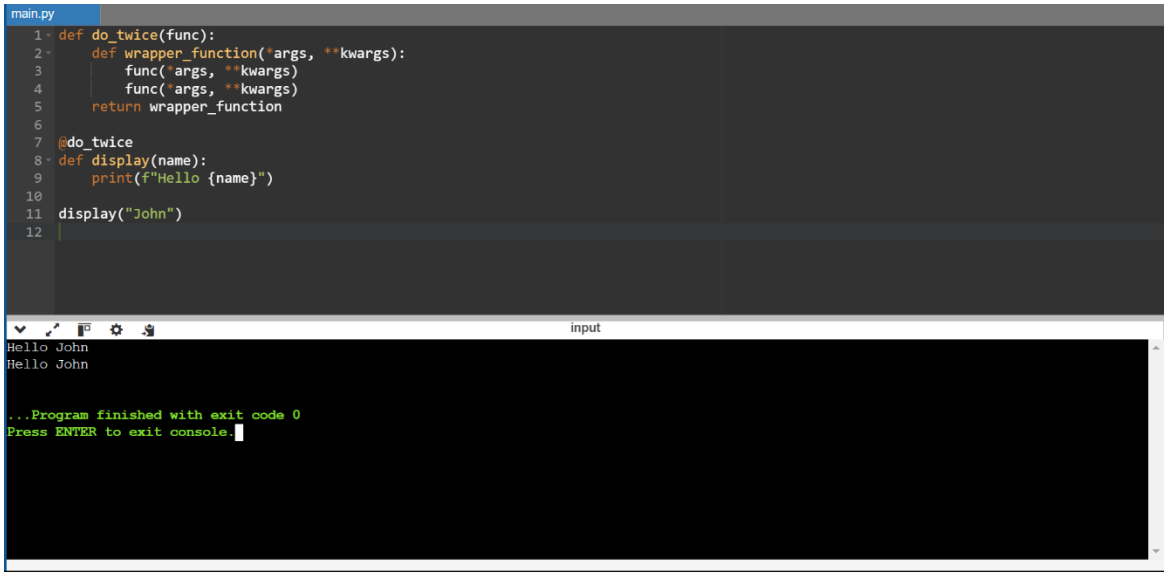
106. Decorating functions with parameters

107. Syntactic Decorator

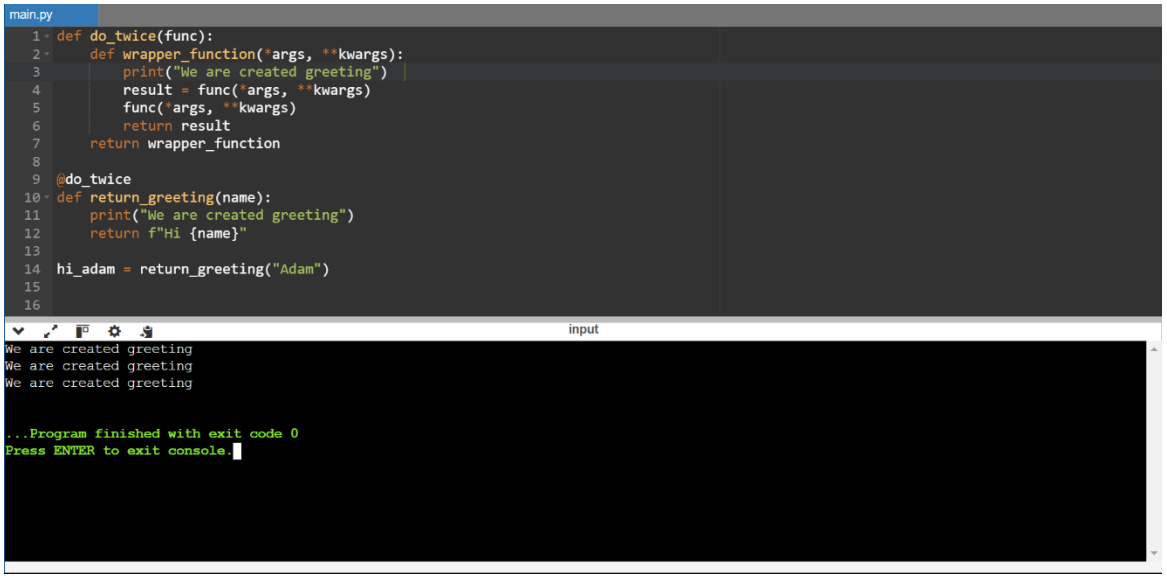


108. Reusing Decorator

109. Python Decorator with Argument



110. Returning Values from Decorated Functions



111. Class Decorators - @property decorator

main.py

```
class Student: # here, we are creating a class with the name Student
    def __init__(self,name,grade):
        self.name = name
        self.grade = grade
    @property
    def display(self):
        return self.name + " got grade " + self.grade

stu = Student("John","B")
print("Name of the student: ", stu.name)
print("Grade of the student: ", stu.grade)
print(stu.display)
```

input

```
Name of the student:  John
Grade of the student:  B
John got grade B


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

112. Class Decorators - @staticmethod decorator

main.py

```
class Person: # here, we are creating a class with the name Student
    @staticmethod
    def hello(): # here, we are defining a function hello
        print("Hello Peter")
per = Person()
per.hello()
Person.hello()
```

input

```
Hello Peter
Hello Peter


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

113. Decorator with Arguments

main.py

```python
import functools
def repeat(num):
    def decorator_repeat(func):
        @functools.wraps(func)
        def wrapper(*args,**kwargs):
            for _ in range(num):
                value = func(*args,**kwargs)
            return value
        return wrapper
    return decorator_repeat

@repeat(num=5)
def function1(name):
    print(f"{name}")
function1("JavatPoint")
```

input

```
JavatPoint
JavatPoint
JavatPoint
JavatPoint
JavatPoint

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

114. Stateful Decorators

main.py

```python
import functools

def count_function(func):
    @functools.wraps(func)
    def wrapper_count_calls(*args, **kwargs):
        wrapper_count_calls.num_calls += 1
        print(f"Call {wrapper_count_calls.num_calls} of {func.__name__!r}")
        return func(*args, **kwargs)
    wrapper_count_calls.num_calls = 0
    return wrapper_count_calls

@count_function
def say_hello():
    print("Say Hello")

say_hello()
say_hello()
```

input

```
Call 1 of 'say_hello'
Say Hello
Call 2 of 'say_hello'
Say Hello


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

115. Classes as Decorators