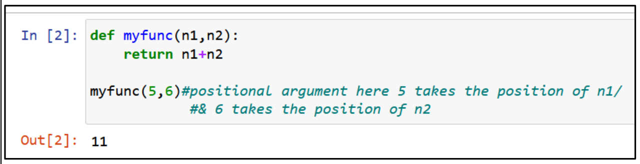
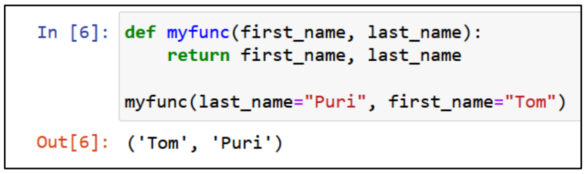
1. In Python, what is the difference between a built-in function and a user-defined function? Provide an example of each.
   ANSWER: Built-in functions: Functions that readily comes with Python are called built-in functions. Example: print(), len(), type() etc.
   User-Defined Functions: Functions defined by the user/programmer are called user defined functions.
   Example: def myfunc( ): etc.

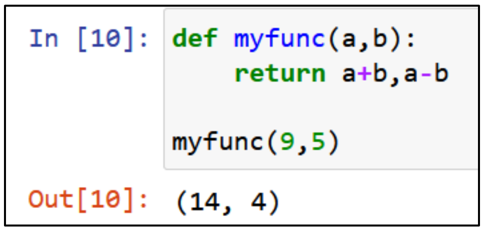
2. How can you pass arguments to a function in Python? Explain the difference between positional arguments and keyword arguments.
   ANSWER: Arguments are specified after the function name, inside the parentheses.
   Example: def myfunc(a,b): #where "a, b ' are arguments.
   return a+b

Positional Arguments	**Keyword Arguments**
The arguments that are passed on their position are called positional arguments.	The arguments that are passed based on th names are called keyword arguments. The order of the argument doesn't matter for Keyword arguments.
	

```
In [2]: def myfunc(n1,n2):
            return n1+n2

        myfunc(5,6)#positional argument here 5 takes the position of n1/
                    #& 6 takes the position of n2

Out[2]: 11
```

```
In [6]: def myfunc(first_name, last_name):
            return first_name, last_name

        myfunc(last_name="Puri", first_name="Tom")

Out[6]: ('Tom', 'Puri')
```

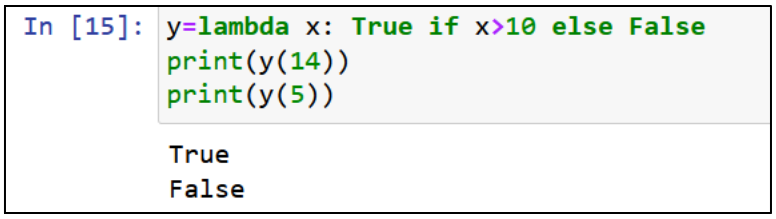
3. What is the purpose of the return statement in a function? Can a function have multiple return statements? Explain with an example.
   ANSWER: The python return statement is used to return the output when the function is called. We **can use** a **return statement** to **return multiple** values from a **function**. To do that, we just **need** to supply several **return** values separated by commas.
   Example:



```
In [10]: def myfunc(a,b):
             return a+b,a-b

         myfunc(9,5)

Out[10]: (14, 4)
```

4. What are lambda functions in Python? How are they different from regular functions? Provide an example where a lambda function can be useful.
   ANSWER: Lambda function is an anonymous function (i.e., defined without a name) that can take any number of arguments but, unlike normal functions, evaluates and returns only one expression. In regular functions we surround the parameters within parenthesis but in Lambda function we do not surround the parameters within parenthesis.

Example:

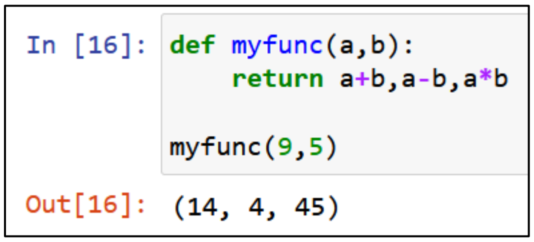


```
In [15]: y=lambda x: True if x>10 else False
         print(y(14))
         print(y(5))

         True
         False
```

5. How does the concept of "scope" apply to functions in Python? Explain the difference between local scope and global scope.
   ANSWER: Generally, a variable is valid only in the block it is created in and is called its **scope**. The variable '**scope**' is a very fundamental **concept** in programming. Function scope in Python means how a particular function is accessible from different components depending on LEGB (Local -> Enclosing -> Global -> Built-in) rule**.**
   Local Scope
   Whenever a variable is defined within a function, its scope lies ONLY within the function. It is accessible from the point at which it is defined until the end of the function.

   Global Scope
   Whenever a variable is declared and defined outside any function, it becomes a global variable, and its scope is anywhere within the program. Which means it can be used and modified by any function.

6. How can you use the "return" statement in a Python function to return multiple values?
   ANSWER:



```
In [16]: def myfunc(a,b):
             return a+b,a-b,a*b

         myfunc(9,5)

Out[16]: (14, 4, 45)
```

7. What is the difference between the "pass by value" and "pass by reference" concepts when it comes to function arguments in Python?

ANSWER: Pass by reference – It is used in some programming languages, where values to the argument of the function are passed by reference which means that the address of the variable is passed and then the operation is done on the value stored at these addresses.

Pass by value – It means that the value is directly passed as the value to the argument of the function. Here, the operation is done on the value and then the value is stored at the address. Pass by value is used for a copy of the variable.

8. Create a function that can intake integer or decimal value and do following operations:

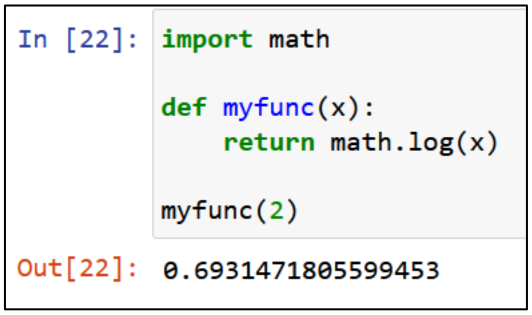
a. Logarithmic function (log x)

b. Exponential function (exp(x))

c. Power function with base 2 (2^x)

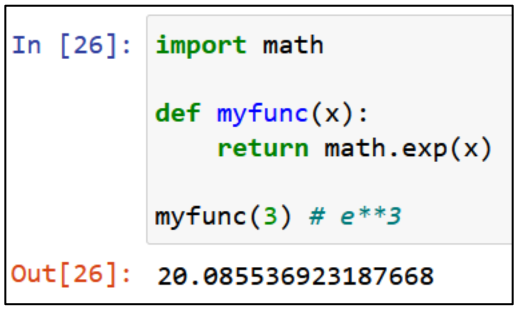
d. Square root

ANSWER: a: Logarithmic function (log x):



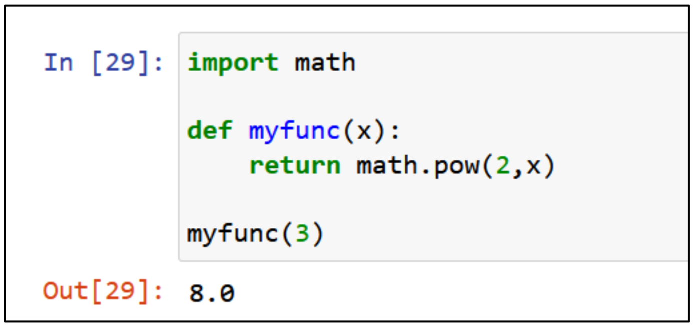
```
In [22]: import math

         def myfunc(x):
             return math.log(x)

         myfunc(2)

Out[22]: 0.6931471805599453
```

b. Exponential function (exp(x))



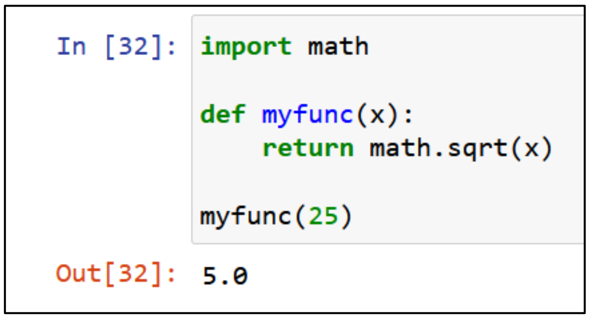
```
In [26]: import math

         def myfunc(x):
             return math.exp(x)

         myfunc(3) # e**3

Out[26]: 20.085536923187668
```

c. Power function with base 2 (2^x):



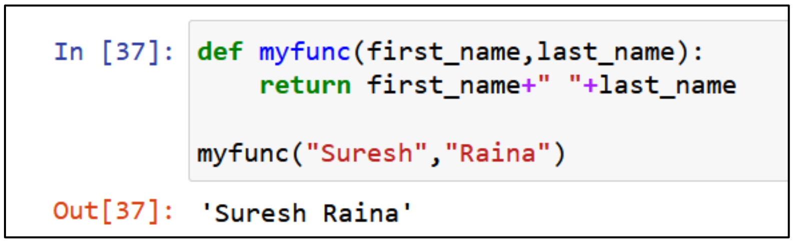
```
In [29]: import math

         def myfunc(x):
             return math.pow(2,x)

         myfunc(3)

Out[29]: 8.0
```

d. Square root:



```
In [32]: import math

         def myfunc(x):
             return math.sqrt(x)

         myfunc(25)

Out[32]: 5.0
```

9. Create a function that takes a full name as an argument and returns first name and last name.
   ANSWER:



```
In [37]: def myfunc(first_name,last_name):
             return first_name+" "+last_name

         myfunc("Suresh","Raina")

Out[37]: 'Suresh Raina'
```