
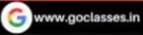
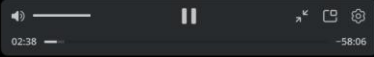



Host




```
graph LR; input --> function; function --> output; output -- "return value" --> output
```









Host

How functions look: syntax

```
def happy():  
    print('Happy birthday to you!')
```

```
def greeting(name):  
    print('Happy birthday, dear ' + name)
```





```
[6]: def happy():  
      print("Happy Birthday to you !!")  
  
      happy()  
  
      Happy Birthday to you !!
```



Hello, Functions!



We define a function using the def keyword:

```
>>> def say_hello():  
...     print('Hello')  
...
```

Once the function is defined, you can call it:


```
>>> say_hello()  
Hello
```







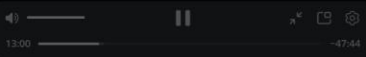
```
def name (parameters) :  
    statements  
    return value      # optionally
```



- **name:** name of function (in **snake_case**)
- **parameters:** information passed into function
- **return:** information given back from the function


Host




 www.goclasses.in



13:00 -47:44




Scope

A variable is only available from inside the region it is created.
This is called scope.


Host



 www.goclasses.in

Scope

- The name that identifies a variable has certain visibility throughout the program
- Three fundamental levels of scope
 - Global ✓
 - Local ✓



```
In [17]: def myfunc():  
         x = 300  
         print(x)
```

```
myfunc()  
print(x)
```

```
300
```



```
-----  
NameError                                Traceback (most recent call last)  
Cell In[17], line 6  
      3 print(x)  
      5 myfunc()  
----> 6 print(x)  
  
NameError: name 'x' is not defined
```





```
In [18]: def myfunc(x):
         print(x)

         myfunc(300)

300
```

```
In [19]: def myfunc(x):
         print(x)

         myfunc(300)
         print(x)


300
```

```
-----
NameError                                Traceback (most recent call last)
Cell In[19], line 5
      2     print(x)
      4     myfunc(300)
--> 5     print(x)

NameError: name 'x' is not defined
```







```
In [18]: def myfunc(x):
         print(x)

         myfunc(300)

300
```


```
In [19]: def myfunc(x):
         print(x)


         myfunc(300)
         print(x)

300
```

```
-----
NameError                                Traceback (most recent call last)
Cell In[19], line 5
      2     print(x)
      4     myfunc(300)
--> 5     print(x)

NameError: name 'x' is not defined
```







```
def mySum(x,y):  
    sum = x + y # Sum is a local variable  
    return sum
```

- x and y are local variables that only exist in the scope of the function mySum



```
def mySum(x,y):  
    sum = x + y # Sum is a local variable  
    return sum  
  
# Global variables  
a = 10  
b = 20  
  
#c is also Global  
c = mySum(a,b)
```



```
In [ ]: x = 300

def myfunc():
    x = 200
    print(x)

myfunc()

print(x)
```

Handwritten annotations: A pink arrow points from the `print(x)` inside the function to the value `200`. Another pink arrow points from the `print(x)` outside the function to the value `300`.

Chrome File Edit View History Bookmarks Profiles Tab Window Help Thu 18 Jul 8:53 PM

Python Programming GO Cl... x +

localhost:8888/notebooks/Documents/GATE%20A/Python%20Programming%20GO%20Classes.ipynb

GO Classes Maint... Inbox (2,710) Sachin GO links GATE GitHub - Develop... GATE DA Google Forms All Bookmar

Read the migration file to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions. Don't show anymore

jupyter Python Programming GO Classes Last Checkpoint: 37 minutes ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [26]: x = 300

```
def myfunc():
    x=x+1
myfunc()
print(x)
```

UnboundLocalError Traceback (most recent call last)

Cell In[26], line 6

```
3 def myfunc():
4     x=x+1
--> 6 myfunc()
8 print(x)
```

Cell In[26], line 4, in myfunc()

```
3 def myfunc():
--> 4     x=x+1
```

UnboundLocalError: cannot access local variable 'x' where it is not associated with a value

32:03 -28:41



```
In [7]: x = 300
```

```
def myfunc():
```

```
    x = 5
```

```
myfunc()
```

```
print(x)
```

Q: how to modify global variable inside the function?



The global keyword



Normally, when you create a variable inside a function, that variable is local, and can only be used inside that function.

To create a global variable inside a function, you can use the "global" keyword.

```
def myfunc():
```

```
    global x
```

```
    x = "fantastic"
```

```
myfunc()
```

```
print("Python is " + x)
```




```
In [7]: x = 300
```

```
def myfunc():  
    x = 5  
myfunc()  
print(x)
```

Q: how to modify global variable inside the function?






```
In [1]: # Variable local to the function
```

```
def increment_x(x):  
    x = x + 1  
    print('This is the value of x inside the function:', x)  
    return  
x = 42  
increment_x(x)  
print('This is the value of x outside the function:', x)
```

```
This is the value of x inside the function: 43  
This is the value of x outside the function: 42
```

✚





Host





```
y = 7
def f(x):
    print(x)
    print(y)
f(4)
print(y)
print(x)
```



Output:


```
4
7
7
NameError: name 'x' is not defined
```

Hmmm....there seems to be an error

 www.goclasses.in

   
42:19 ~18:25





Host

```
a=3
b=2
def foo(x):
    return a+x
def bar(x):
    b=1
    return b+x
print(foo(3), bar(3))
```

b is now local

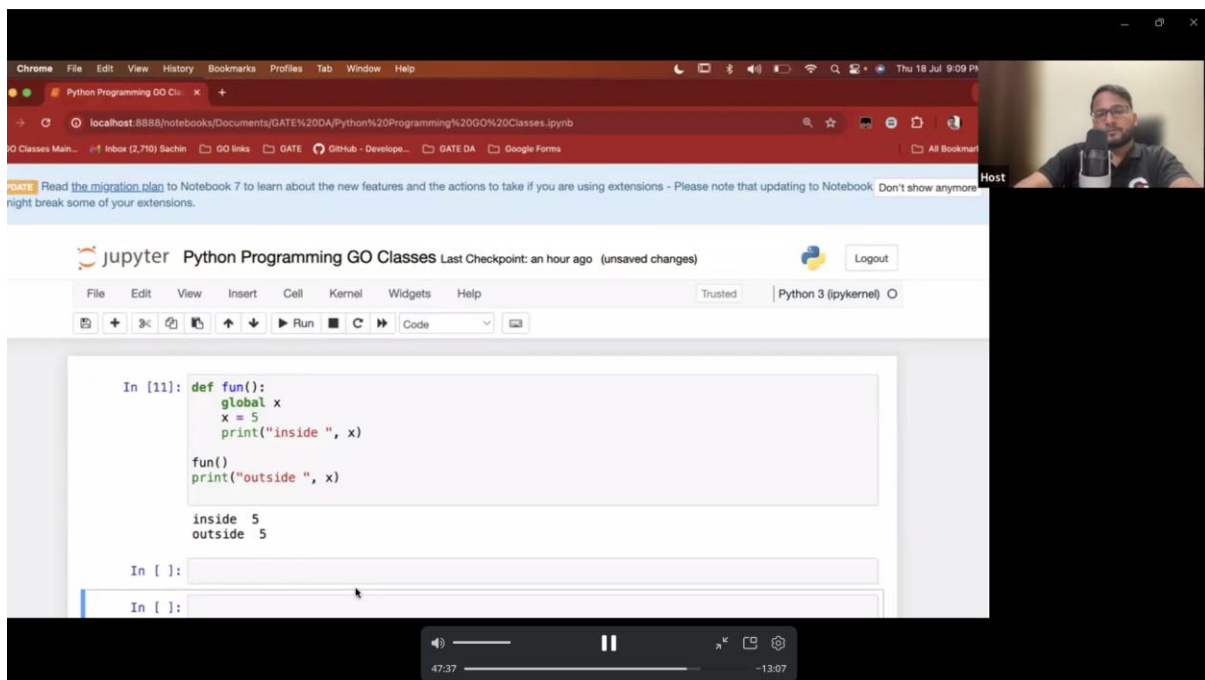
- a and b are two global variables
- In function foo:
 - a is global, its value remains 3
- In function bar:
 - b is local, since it is redefined to be 1

 www.goclasses.in



"global" keyword serve two purposes:

- ① modify existing global variable inside the fun.
- ② there is no existing global variable, we need to define global variable inside the function



Chrome File Edit View History Bookmarks Profiles Tab Window Help

Python Programming GO Cl... x +

localhost:8888/notebooks/Documents/GATE%20DA/Python%20Programming%20GO%20Classes.ipynb

GO Classes Main... Inbox (2,710) Sachin GO links GATE GitHub - Develop... GATE DA Google Forms All Bookmar

NOTE Read the [migration plan](#) to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions. Don't show anymore

jupyter Python Programming GO Classes Last Checkpoint: an hour ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [11]:

```
def fun():
    global x
    x = 5
    print("inside ", x)

fun()
print("outside ", x)
```

inside 5
outside 5

In []:

In []:

47:37 -13:07

Question: What will be the output of the following code ?

```
a = 1
b = 2
```

```
def foo():
    global a
    a = 2
    b = 3
```

```
    print("In foo:" , "a=", a, " b=", b)
```

```
print("Outside foo: " , "a=", a, " b=", b)
```

```
foo()
```

```
print("Outside foo: " , "a=", a, " b=", b)
```

- In foo:

- A local variable *b*

- A global variable *a*

- The value of *a* changes by executing *foo()*

IES

→ *a = 1 , b = 2*

→ *In foo a = 2, b = 3*

```
a = 1
b = 2
```

```
def foo():
    global a
    a = 2
    b = 3
```

```
    print("In foo:" , "a=", a, " b=", b)
```

```
print("Outside foo: " , "a=", a, " b=", b)
```

```
foo()
```


```
print("Outside foo: " , "a=", a, " b=", b)
```

```
##Outside foo: a= 1 b= 2
```

```
##In foo: a= 2 b= 3
```

```
##Outside foo: a= 2 b= 2
```

Answers



```
In [12]: a = 100

def bar():
    a = a+1

bar()
print(a)
```

UnboundLocalError Traceback (most recent call last)


Cell In[12], line 8

```
4 def bar():
5     a = a+1
--> 8 bar()
9 print(a)
```


Cell In[12], line 5, in bar()

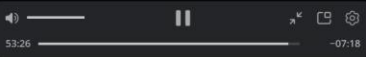
```
4 def bar():
--> 5     a = a+1
```

UnboundLocalError: cannot access local variable 'a' where it is not associated with a value



Host





53:26 -07:18

```
def mystery(a):
    print(a)
    for i in range(1, len(a)):
        a[i] += a[i-1]
        print(a)

mystery([8, 5, 0, -7, 4])
```

[8, 5, 0, -7, 4]


[8, 13, 0, -7, 4]


[8, 13, 13, -7, 4]

[8, 13, 13, 6, 4]


[8, 13, 13, 6, 10]

<https://www.cs.middlebury.edu/~cs101/homework/cs101-midterm-sample-Qs-soln.pdf>

 www.goclasses.in



Host



06:42 -1:12:45

```
In [3]: l = ['a','b']

def fun(p,q):
    p.append(q)

fun(l,'c')

print(l)

['a', 'b', 'c']
```

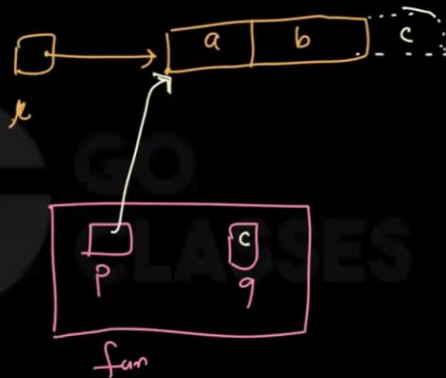


`l = ['a', 'b']` ✓

`def func(p, q):`
`p.append(q)`

`func(l, 'c')`

`print(l)`



Question: What will be the output of the following code ?

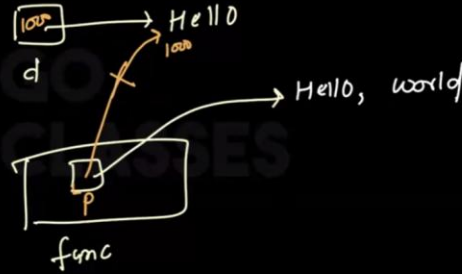
```
d = "Hello"
```

```
def func(p):  
    p = p + ", world!"
```

```
func(d)
```

```
print(d)
```

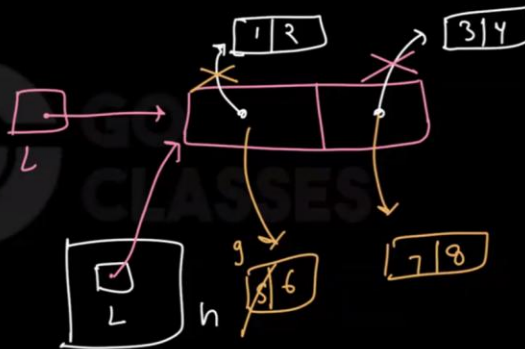
?? hello




```
def h(L):  
    L[1] = [7, 8]  
    L[0] = [5, 6]  
    L[0][0] = 9  
    print(L)
```

```
L = [[1, 2], [3, 4]]  
h(L)  
print(L)
```

↳ [[9, 6], [7, 8]]



[[9, 6], [7, 8]]



```
In [1]: def greeting(name = "friend"):
        print("Hello "+name)

        greeting()
        greeting("Mike ")


Hello friend
Hello Mike
```

```
In [2]: def greeting(name ):
        print("Hello "+name)

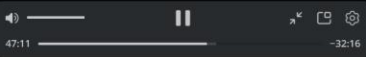
        greeting()
        greeting("Mike ")

TypeError                                 Traceback (most recent call last)
Cell In[2], line 4
      1 def greeting(name ):
      2     print("Hello "+name)
--> 4 greeting()
      5 greeting("Mike ")

TypeError: greeting() missing 1 required positional argument: 'name'
```



Host



47:11 -32:16

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Python Programming GO Cl...

localhost:8888/notebooks/Documents/GATE%20DA/Python%20Programming%20GO%20Classes.ipynb

GO Classes Main... Inbox (2,710) Sachin GO links GATE GitHub - Develop... GATE DA Google Forms

Read the [migration plan](#) to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions.

jupyter Python Programming GO Classes Last Checkpoint: an hour ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (pykernel)

Run Code


```
In [3]: def foo(x, y=10):
        print(x+y)

        foo(5)
        foo(5,100)



15
105
```


In []:

In []:



Host





Host


```
# Demonstrating default argument values

def say(s, times=1):
    print(s * times)

say('Hello')
say('World', 3)
```

Handwritten notes:
An arrow points from `say('Hello')` to `Hello`.
An arrow points from `say('World', 3)` to `World World World`.




49:53 -29:34




Host

Order of Default Arguments:

Consider the following function definition:

```
def fun(x=1, y, z=2):
    print(x,y,z)
```

Suppose we call the function with `fun(5,6)` then what are the values of `x,y,x` inside function?




50:03 -29:24

Using Default Argument Values

Only those parameters which are at the end of the parameter list can be given default argument values.

- ▶ We cannot have a parameter with a default argument value before a parameter without a default argument value, in the order of parameters declared, in the function parameter list.
- ▶ This is because values are assigned to the parameters by position.

Example

- ▶ `def func(a, b=5)` is valid
- ▶ `def func(a=5, b)` is not valid

53:39 -25:48

The screenshot shows a Jupyter Notebook interface with a Chrome browser window at the top. The browser address bar shows the URL: `localhost:8888/notebooks/Documents/GATE%20DA/Python%20Programming%20GO%20Classes.ipynb`. The Jupyter Notebook interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running, and saving. The notebook content shows a Python function definition:

```
In [4]: def fun(x=1, y, z=2):  
        print("x = ", x)  
        print("y = ", y)  
        print("z = ", z)  
  
        fun(5,6)
```

The output of the cell shows a `SyntaxError` message:

```
Cell In[4], line 1  
    def fun(x=1, y, z=2):  
                ^  
SyntaxError: non-default argument follows default argument
```

The error message indicates that the parameter `y` does not have a default value but is positioned after parameters `x` and `z` which do have default values. This is an invalid syntax in Python.

54:13 -25:14



```
In [3]: def fun(x, y=1, z=2):
        print("x = ", x)
        print("y = ", y)
        print("z = ", z)

        fun(5)

x = 5
y = 1
z = 2
```

```
In [4]: def fun(x, y=1, z=2):
        print("x = ", x)
        print("y = ", y)
        print("z = ", z)

        fun(5,9)

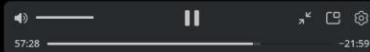
x = 5
y = 9
z = 2
```

```
In [5]: def fun(x, y=1, z=2):
        print("x = ", x)
        print("y = ", y)
        print("z = ", z)

        fun(5, 9, 84)

x = 5
y = 9
z = 84
```

put all default arguments at the end.



```
In [5]: def func(a, b=5, c=10, d=20):
        print("a =", a)
        print("b =", b)
        print("c =", c)
        print("d =", d)

        # Example usage:
        func(1)
        func(1, 2)
        func(1, 2, 3)
        func(1, 2, 3, 4)
```



In [5]: `def func(a, b=5, c=10, d=20):`

```
    print("a =", a)
    print("b =", b)
    print("c =", c)
    print("d =", d)
```

`# Example usage:`

```
func(1)           # This will use the default values for b, c, and d
func(1, 2)        # This will use the default values for c and d
func(1, 2, 3)     # This will use the default value for d
func(1, 2, 3, 4)  # This will use the provided values for all parameters
```

`:-`

```
a = 1
b = 5
c = 10
d = 20
```

```
a = 1
b = 2
c = 10
d = 20
```

```
a = 1
b = 2
c = 3
d = 20
```

```
a = 1
b = 2
c = 3
d = 4
```



Host