As others have noted, you need to declare a variable global in a function when you want that function to be able to modify the global variable. If you only want to access it, then you don't need global.

def a():

global foo

foo = 'A'

def b():

global foo

foo = 'B'

b()

a()

print foo

# prints 'A' because a() was the last function to modify 'foo'.

Note that global is only required to modify global objects. You can still access them from within a function without declaring global. Thus, we have:

x = 5

def access\_only():

return x

# This returns whatever the global value of 'x' is

def modify():

global x

x = 'modified'

return x

# This function makes the global 'x' equal to 'modified', and then returns that value

def create\_locally():

x = 'local!'

return x

# This function creates a new local variable named 'x', and sets it as 'local',

# and returns that. The global 'x' is untouched.

Note the difference between create\_locally and access\_only -- access\_only is accessing the global x despite not calling global, and even though create\_locally doesn't use global either, it creates a local copy since it's assigning a value.