Global variables (global), local variables (local), nonlocal variables in Python

In this Python lesson you will learn about global variables, local variables, nonlocal variables in Python and the use of these variables.

In this Python lesson you will learn about global variables, local variables, nonlocal variables in Python and the use of these variables.

Global variables in Python

In the Python programming language, a variable declared outside a function or in a global scope is called a global variable or a global variable. Global variables can be accessed from inside or outside the function.

Take a look at an example of how to create global variables in Python.

```
x = "Bi?n toàn c?c" #khai báo bi?n x
#G?i x t? trong hàm vidu()
def vidu():
 print("x trong hàm vidu() :", x)

vidu()
#G?i x ngoài hàm vidu()
print("x ngoài hàm vidu():", x)
```

In the above example, we declare the variable x as a global variable, and define the function `vidu()` to print the variable x. Finally, we call `vidu()` to print the value of variable `x` . Running the above code will result in:

```
 x in vidu (): Global variable
 x outside vidu (): Global variable
```

What happens if you change the value of x in the function?

```
x = 2
def vidu():
  x=x*2
 print(x)

vidu()
```

If you run this code you will get an error message:

```
  Không k?t n?iLocalError: ??a ch? n?i b? x x tham chi?u tr??c khi chuy?n
??i
```

This error occurs because Python treats `x` as a local variable and `x` not defined in `vidu()`.

To change global variables in a function you will have to use the global keyword. We will talk more in the article about global keywords.

Local variable in Python

Variables declared inside a function or in a local scope are called local or local variables.

```
  def vidu ():
  y = "Local variable"
  vidu ()
  print (y)
```

When you run the above code, you will receive an error message:

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

This error occurs because we tried to access the local variable y in the global scope, but he only works in the `vidu()` function or the local scope.

Normally, to create a local variable, we will declare it in a function like the example below:

```
  def vidu ():
  y = "Local variable"
  print (y)
  vidu ()
```

Running the above code will result

```
  Local variable
```

We go back to looking at the previous problem, where `x` is a global variable and we want to change `x` in `vidu()`.

Local variables and global variables

Here, we will learn how to use local and global variables in the same code.

```
  x = 2

  def vidu ():
  global x
  y = "Local variable"
  x = x * 2
  print (x)
  print (y)
```

```
# Written by TipsMake.com
vidu ()
```

Running the above code we will have output:

```
4
Local variable
```

In the above code, we declare `x` as a global variable and `y` is a local variable in `vidu()` and use the * operator to change global variables and print both the values ??of x and y. After calling `vidu()` the value of x will be 4 because it is duplicated.

Examples using global variables and localized names:

```
x = 5

def vidu ():
x = 10
print ("Local variable:", x)

vidu ()
print ("Global variable x:", x)
```

After running the above code, we have the output:

```
Local variable: 10
Global variable x: 5
```

In the above code, we use the same x name for both local and global variables. When printing the same variable x, we get two different results because the variable is declared in both the scope and the local (inside the `vidu()` function) and the global (outside the function `vidu()`).

When we print the variable in the function `vidu()` it will output `Bi?n x c?c b?: 10` , this is called the local scope of the variable. Similarly when we print variables outside the function `vidu()` will produce `Bi?n x toàn c?c: 5` , this is the global scope of the variable.

Nonlocal variable in Python

From this nonlocal I don't know how to translate it into Vietnamese so it is standard. In Python, the nonlocal variable is used in nested functions where local scope is not defined. Understandably, the nonlocal variable is not a local variable, not a global variable, you declare a variable as nonlocal when you want to use it at a wider range than local, but not to a global level.

To declare the variable nonlocal we need to use the nonlocal keyword.

For example:

```
def ham_ngoai():
 x = "Bi?n c?c b?"

 def ham_trong():
```

```
    nonlocal x
    x = "Bi?n nonlocal"
    print("Bên trong:", x)

    ham_trong()
    print("Bên ngoài:", x)

hamngoai()
```

Running the code above you will have the output:

```
    Inside: Variable nonlocal
    Outside: Variable nonlocal
```

In the above code there is a nested function called `ham_trong()` , we use the `nonlocal` keyword to create a nonlocal variable. `ham_trong()` function is defined in the scope of `ham_ngoai()` .

Note: If we change the value of the nonlocal variable, the change will appear in the local variable.

Don't forget to do your Python homework.

Next article: Global keyword in Python

Previous article: Anonymous function, Lambda in Python