محدوده متغیر

متغیرها در پایتون دارای محدوده (scope) هستند. محدوده یک متغیر به شما می‌گوید که در کجای برنامه می‌توان از متغیر استفاده کرد و یا متغیر قابل دسترسی است. به عنوان مثال متغیری که در داخل یک تابع تعریف می‌شود فقط در داخل بدنه تابع قابل دسترسی است. می‌توان دو متغیر با نام یکسان در دو تابع مختلف تعریف کرد. برنامه زیر این ادعا را اثبات می‌کند :

```
def firstFunction():
    number = 5;
    print("number inside method firstMethod() = {0}".format(number));

def secondFunction():
    number = 10;
    print("number inside method secondMethod() = {0}".format(number));

firstFunction();
secondFunction();
```

```
number inside method firstFunction() = 5
number inside method secondFunction() = 10
```

مشاهده می‌کنید که حتی اگر ما دو متغیر با نام یکسان تعریف کنیم که دارای محدوده‌های متفاوتی هستند، می‌توان به هر کدام از آنها مقادیر مختلفی اختصاص داد. متغیر تعریف شده در داخل تابع firstFunction() هیچ ارتباطی به متغیر داخل تابع secondFunction() ندارد. همانطور که ذکر شد، متغیری که در داخل بدنه یک تابع تعریف شود در خارج از تابع قابل دسترسی نیست. به مثال زیر توجه کنید :

```
def myFunction():
    number = 10

print(number)
```

در تابع بالا یک متغیر به نام number تعریف شده است. اگر بخواهیم در خارج از تابع یعنی خط آخر مقدار این متغیر را چاپ کنیم با پیغام خطا مواجه می شویم. چون این متغیر فقط در داخل تابع قابل دسترسی است. به این متغیرها محلی یا Local گفته می شود. یک نوع دیگر از متغیرها، عمومی یا global هستند. این متغیر ها در خارج از تابع تعریف می شوند و در داخل بدنه تابع قابل دسترسی هستند. به مثال زیر توجه کنید :

```
number = 10
def myFunction():
    print(number)

myFunction();
```

```
10
```

در کد بالا یک متغیر به نام number با مقدار 10 در خارج از تابع تعریف شده است. از تابع myFunction() خواسته ایم که مقدار این متغیر را در هنگام فراخوانی چاپ کند. این اتفاق می افتد، چون متغیر های خارج از تابع در داخل تابع قابل دسترسی هستند. به کد زیر توجه کنید:

```
number = 10

def myFunction():
    number = 5
    print(number)

myFunction()
print(number)
```

```
5
10
```

به دو متغیر number در خطوط 1 و 4 توجه کنید. این دو متغیر جدا از هم هستند. چون یکی از آنها global و دیگری Local است. و چون هیچ ارتباطی به هم ندارند، در نتیجه خروجی نیز متفاوت است. حال اگر بخواهیم مقدار یک متغیر عمومی و خارج از تابع را در داخل تابع مورد استفاده قرار یا تغییر دهیم باید چکار کنیم؟ راهکار، استفاده از کلمه کلیدی global است. به مثال زیر توجه کنید :

```
number = 15

def myFunction():
    global number
    number = 10
    print("The value of the number inside the function is : " , number)

myFunction()
print("The value of the number outside the function is : " , number)
```

```
The value of the number inside the function is :  10
The value of the number outside the function is :  10
```

در کد بالا یک متغیر تعریف کرده ایم و قبل از آن کلمه کلیدی global را نوشته ایم. این کلمه به برنامه می فهماند که قرار است از فلان متغیر در خارج از تابع استفاده شود. همانطور که مشاهده می کنید با وجود کلمه global می توان به متغیر number در خارج از تابع دسترسی داشت و از آن استفاده کرد و یا آن را تغییر داد. ما در اینجا فقط مقدار number را خارج از تابع تغییر داده ایم. ولی شما ممکن است که بخواهید از آن، استفاده های دیگری بکنید. همانطور که مشاهده می کنید با تغییر مقدار متغیر number در داخل تابع، مقدار آن در خارج از تابع هم تغییر می کند. نکته آخر این است که در خطی که کلمه global به کار رفته است نمی توان عمل انتساب را انجام داد. یعنی خط زیر اشتباه است:

```
global number = 10;
```

و در نهایت مطلب بالا در کد زیر خلاصه می شود:

```
number = 1

# Uses global because there is no local 'number'
def function1():
    print ('Inside function1() : ', number)

# Variable 'number' is redefined as a local
def function2():
    number = 2
    print ('Inside function2() : ',number)

# Uses global keyword to modify global 'number'
def function3():
    global number
    number = 3
    print ('Inside function3() : ',number)

# Global scope
print('global : ',number)
function1()
print('global : ',number)
function2()
print('global : ',number)
function3()
print('global : ',number)
```

```
global :  1
Inside function1() :  1
global :  1
Inside function2() :  2
global :  1
Inside function3() :  3
global :  3
```