Python - Fechamentos

Neste capítulo, vamos discutir o conceito de encerramentos em Python. Em Python, as funções são consideradas objetos de primeira ordem. Assim como os tipos de dados primários, as funções também podem ser usadas atribuídas a variáveis ou passadas como argumentos.

Funções aninhadas

Você também pode ter uma declaração aninhada de funções, em que uma função é definida dentro do corpo de outra função.

Exemplo

```
def functionA():
   print ("Outer function")
   def functionB():
      print ("Inner function")
   functionB()

functionA()
```

Ele produzirá a seguinte **saída** -

```
Outer function
Inner function
```

No exemplo acima, functionB é definida dentro de functionA. A função interna é então chamada de dentro do escopo da função externa.

Se a função externa receber algum argumento, ele poderá ser passado para a função interna.

```
def functionA(name):
   print ("Outer function")
   def functionB():
      print ("Inner function")
      print ("Hi {}".format(name))
   functionB()

functionA("Python")
```

Ele produzirá a seguinte saída -

```
Outer function
Inner function
Hi Python
```

O que é um fechamento?

Um fechamento é uma função aninhada que tem acesso a uma variável de uma função envolvente que concluiu sua execução. Tal variável não está vinculada ao escopo local. Para usar variáveis imutáveis (número ou string), temos que usar a palavra-chave nonlocal.

A principal vantagem dos encerramentos Python é que podemos ajudar a evitar o uso de valores globais e fornecer alguma forma de ocultação de dados. Eles são usados em decoradores Python.

Exemplo

```
def functionA(name):
   name ="New name"
   def functionB():
      print (name)
   return functionB

myfunction = functionA("My name")
myfunction()
```

Ele produzirá a seguinte **saída** -

```
New name
```

No exemplo acima, temos uma função functionA, que cria e retorna outra função functionB. A função functionB aninhada é o fechamento.

A função functionA externa retorna uma função functionB e a atribui à variável myfunction. Mesmo que tenha finalizado sua execução. Porém, o fechamento da impressora ainda tem acesso à variável name.

Palavra-chave não local

Em Python, a palavra-chave não local permite que uma variável fora do escopo local seja acessada. Isto é usado em um encerramento para modificar uma variável imutável presente no escopo da variável externa.

Exemplo

```
def functionA():
   counter =0
   def functionB():
      nonlocal counter
      counter+=1
      return counter
   return functionB

myfunction = functionA()

retval = myfunction()
print ("Counter:", retval)

retval = myfunction()
print ("Counter:", retval)

retval = myfunction()
print ("Counter:", retval)
```

Ele produzirá a seguinte **saída** -

```
Counter: 1
Counter: 2
Counter: 3
```