LAB03-03_Functions (ASSIGN03)

October 24, 2021

Practice with Functions in Python

Scope of a Variable

Look a bit more closely at the issue of scope.

The scope of a variable is the part of that program where that variable is accessible. Variables that are declared outside of all function definitions, such as the myFavouriteBand variable in the code shown here, are accessible from anywhere within the program. As a result, such variables are said to have global scope, and are known as global variables. myFavouriteBand is a global variable, so it is accessible from within the getBandRating function, and we can use it to determine a band's rating. We can also use it outside of the function, such as when we pass it to the print function to display it:

[1]:
```
# Example of global variable

myFavouriteBand = "AC/DC"

def getBandRating(bandname):
    if bandname == myFavouriteBand:
        return 10.0
    else:
        return 0.0

print("AC/DC's rating is:", getBandRating("AC/DC"))
print("Deep Purple's rating is:",getBandRating("Deep Purple"))
print("My favourite band is:", myFavouriteBand)
```

```
AC/DC's rating is: 10.0
Deep Purple's rating is: 0.0
My favourite band is: AC/DC
```

Take a look at this modified version of our code. Now the myFavouriteBand variable is defined within the getBandRating function. A variable that is defined within a function is said to be a local variable of that function. That means that it is only accessible from within the function in which it is defined. Our getBandRating function will still work, because myFavouriteBand is still defined within the function. However, we can no longer print myFavouriteBand outside our function, because it is a local variable of our getBandRating function; it is only defined within the getBandRating function:

[2]:
```
# Example of local variable

def getBandRating(bandname):
    myFavouriteBand = "AC/DC"
    if bandname == myFavouriteBand:
        return 10.0
    else:
        return 0.0

print("AC/DC's rating is: ", getBandRating("AC/DC"))
print("Deep Purple's rating is: ", getBandRating("Deep Purple"))
print("My favourite band is", myFavouriteBand)
```

```
AC/DC's rating is:  10.0
Deep Purple's rating is:  0.0
My favourite band is AC/DC
```

Finally, take a look at this example. We now have two myFavouriteBand variable definitions. The first one of these has a global scope, and the second of them is a local variable within the getBandRating function. Within the getBandRating function, the local variable takes precedence. **Deep Purple** will receive a rating of 10.0 when passed to the getBandRating function. However, outside of the getBandRating function, the getBandRating s local variable is not defined, so the myFavouriteBand variable we print is the global variable, which has a value of **AC/DC**:

[3]:
```
# Example of global variable and local variable with the same name

myFavouriteBand = "AC/DC"

def getBandRating(bandname):
    myFavouriteBand = "Deep Purple"
    if bandname == myFavouriteBand:
        return 10.0
    else:
        return 0.0

print("AC/DC's rating is:",getBandRating("AC/DC"))
print("Deep Purple's rating is: ",getBandRating("Deep Purple"))
print("My favourite band is:",myFavouriteBand)
```

```
AC/DC's rating is: 0.0
Deep Purple's rating is:  10.0
My favourite band is: AC/DC
```

1 Start with some simple exercises

Come up with a function that divides the first input by the second input:

[4]:
```
first_input = int(input('enter the  first number :'))
second_input = int(input(' enter the second number :'))

def divide (a, b):


     return a/b

print('the result of devision is :', divide(first_input,second_input))
```

```
enter the  first number : 250
 enter the second number : 25
```

```
the result of devision is : 10.0
```

Use the function con for the following question.

[5]:
```
# Use the con function for the following question

def con(a, b):
    return(a + b)
```

Can the con function we defined before be used to add to integers or strings?

[6]:
```
# Write your code below and press Shift+Enter to execute
```

Can the con function we defined before be used to concatenate a list or tuple?

[7]:
```
# Write your code below and press Shift+Enter to execute
```

2 More exercises

Favorite Colors

- Write a function that takes two arguments, a person's name and their favorite color. The function should print out a statement such as "Hillary's favorite color is blue."
- Call your function three times, with a different person and color each time.

[8]:
```
def MyColor (name, color):
    print(name,"'s color is:", color)

for i in range (3) :

    name =str(input('Enter a name:'))
    color =str(input(' Enter a color :'))
    MyColor(name, color)
```

```
Enter a name: sara
 Enter a color : blue

sara 's color is: blue

Enter a name: jack
 Enter a color : red

jack 's color is: red

Enter a name: tom
 Enter a color : brown

tom 's color is: brown
```

Phones

- Write a function that takes two arguments, a brand of phone and a model name. The function should print out a phrase such as "iPhone 6 Plus".
- Call your function three times, with a different combination of brand and model each time.

[9]:
```
def introduction (a, b):
     print(a, b)

for i in range (3) :

    brand =str(input('Enter a brand:'))
    model =str(input(' Enter a model :'))
    introduction (brand, model)
```

```
Enter a brand: Samsung
 Enter a model : Galaxy Note20

Samsung Galaxy Note20

Enter a brand: Iphone
 Enter a model : 12PRO

Iphone 12PRO

Enter a brand: Iphone
 Enter a model : 13

Iphone 13
```

[ ]:
```

```