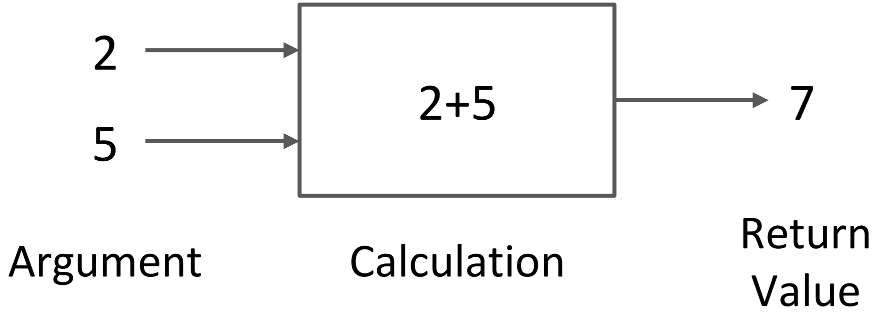
Lecture-3

Python Function

Introduction

- In easy words, function is a code segment that does some specific tasks.
- It takes **argument**, does some **calculation**, and **returns** a result.



Python Function

There are two types of functions-

Built-in functions

- **print( )**, **input( )**, **float( )**, etc are built-in functions
- looking familiar? Yes, we have already used them.
- these functions are already implemented in different libraries
- sometimes we need to **import** those libraries to use these function

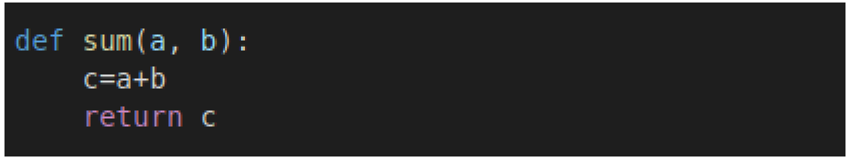
User defined functions

- we can write functions based on our requirement
- we use **def** keyword to define a function
- like as built-in function, we can use user defined functions multiple times

Defining a Function

- A function begins with **def** keyword followed by function name, a set of parentheses and a colon (:)
- Parentheses contains the **parameters** of the function. If the parentheses are empty, that means this function does not contain any **parameters**.
- Function may have **return** statement to return a value. If there is no **return** statement, function does not return anything.
- Statements after colon(:) with same **indentation** are considered as function's block.
- **Parameters** and all the variables within a function block are local variables

Example of Defining Function



```python
def sum(a, b):
    c=a+b
    return c
```

Here, **def** is the keyword

sum is the function name

(a, b) are the parameters

return c is the return statement

c=a+b & ***return c*** both statement have started with same number of spaces. These spaces are known as **indentation**. And for this these two statement are considered as the function's block.

Calling a Function

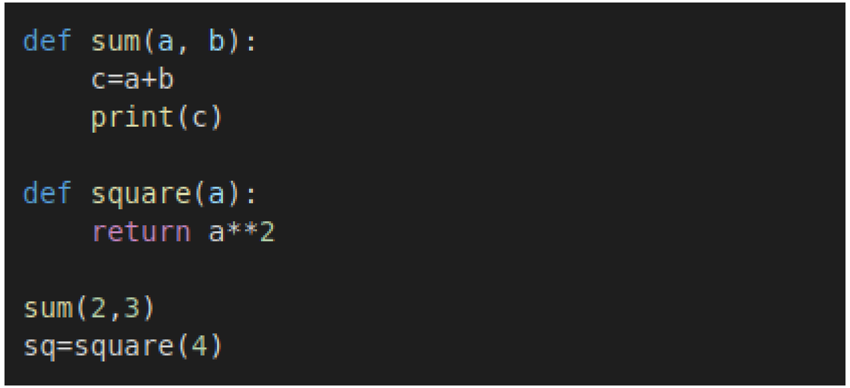
A function can be called by the **function name** followed by a set of parentheses.

Parentheses contains the **parameters**.

If the function has **return** statement, a **variable** is used to store the return value.

A function can be called multiple times in a program. In the same way a single program can have multiple functions.

Function Calling



```
def sum(a, b):
    c=a+b
    print(c)

def square(a):
    return a**2

sum(2,3)
sq=square(4)
```

Here, **sum( )** function does not have any return statement
But **square( )** function have return statement. That why the return value is has been stored in **sq** variable

Different Parameter List

There can be three scenarios-

1. Function **parameter** can have multiple variables. Each variable should be separated by comma.

```
def fun(a, b, c, d):
```

2. Function can have default **parameter** value. But if the function is called with new values, they will override default values.

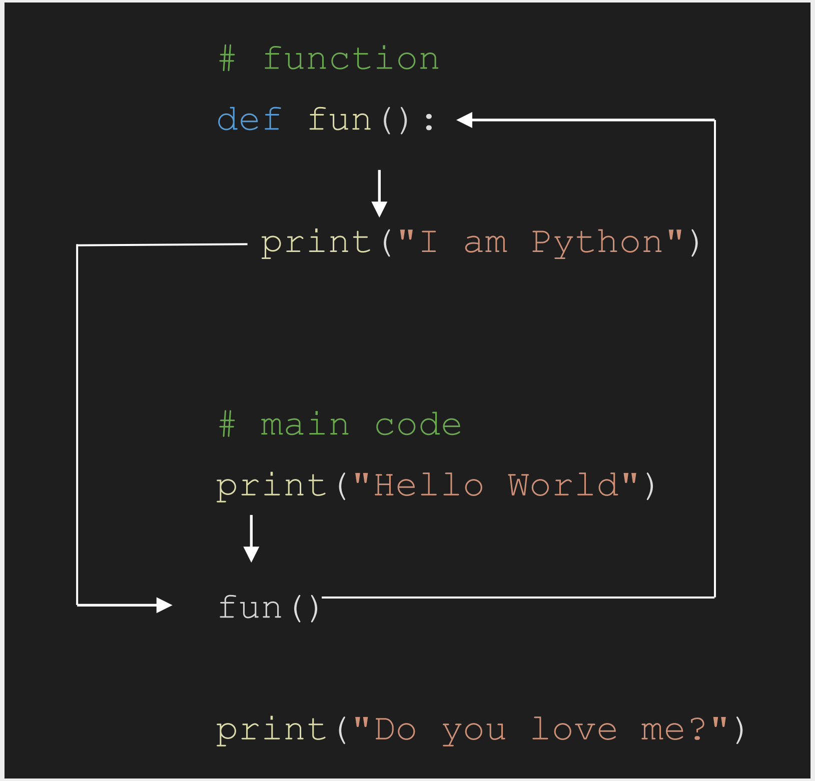
```
def fun(a=2, b="python", c=3.6):
```

3. Function can be defined without **parameter**. In that case, we do not need to pass any value while calling the function.

```
def fun():
```

What Happens When A Function Called

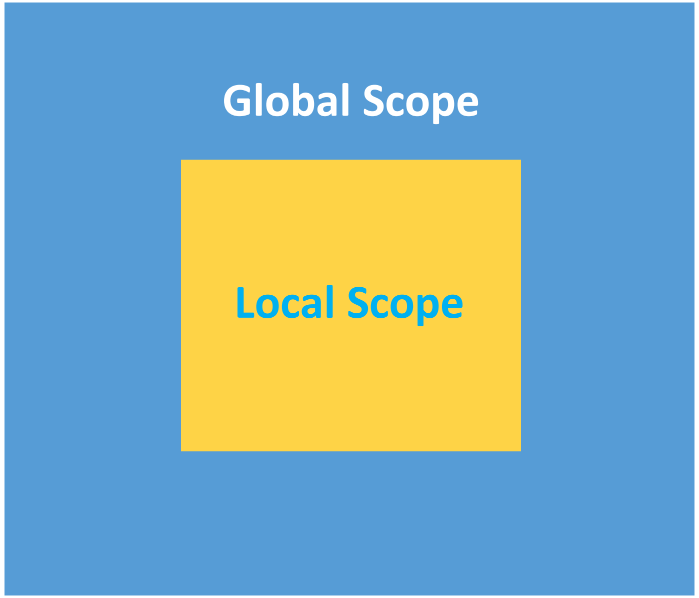
No more text, let’s see what happens internally when we call a function.



```
# function
def fun():
    print("I am Python")


# main code
print("Hello World")

fun()


print("Do you love me?")
```

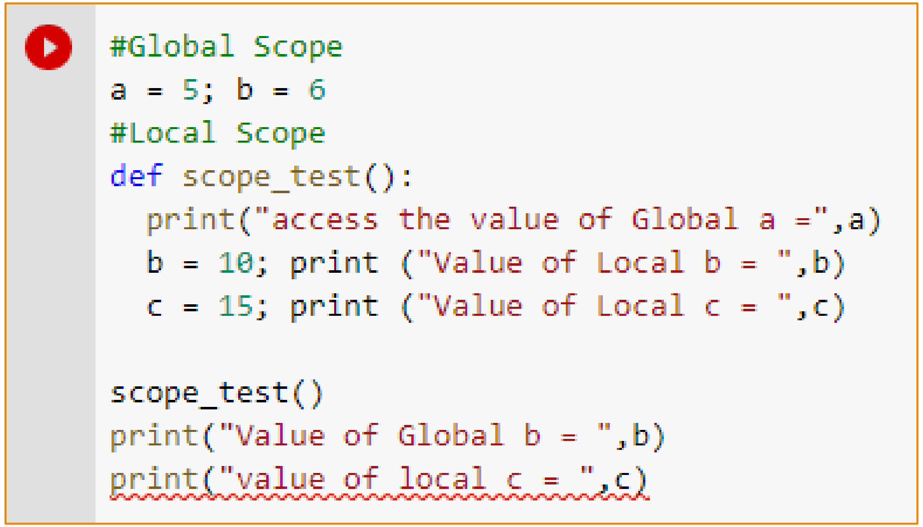
Scope of Variables

- Each identifier or variable is restricted by some regions according to its declaration. This region is known as scope.
- That variable is totally unknown to program outside of its scope.
- Basically there are two scopes; Global Scope, Local Scope When a variable is declared inside a function or class, it’s considered as in local scope.
- When a variable is declared outside a function or class, it’s considered as in global scope.



Scope of Variables

If same identifier is used in global scope (outside of a function) and local scope (inside of a function), program treats them as different variable. Variable of global scope can be accessed from anywhere. Variable of local scope are discarded when the function or class block ends. A variable declared inside a local scope can be treated as global by adding **global** keyword.



```
#Global Scope
a = 5; b = 6
#Local Scope
def scope_test():
  print("access the value of Global a =",a)
  b = 10; print ("Value of Local b = ",b)
  c = 15; print ("Value of Local c = ",c)

scope_test()
print("Value of Global b = ",b)
print("value of local c = ",c)
```

a is declared in global scope and accessible inside the function

b is declared in both scope and the program treated them as different.

d is declared in local scope but with **global** keyword. So, it is treated as global variable and accessible outside of function.

c is declared in local scope and when it was tried to access from outside of function, it shows an error.

Python Standard Library

- A module in python is a file that contains the group of related functions, data and classes.
- Python standard library is a collection of such module that contains the core contents of python language.
- Its packages and modules contain capabilities for a wide variety of everyday programming tasks.

Some important modules

collections —Data structures beyond lists, tuples, dictionaries and sets

csv —Processing comma-separated value files (like those in Excel).

datetime —Date and time manipulations. Also modules time and calendar .

decimal —Fixed-point and floating-point arith-metic, including monetary calculations.

math —Common math constants and operations.

os —Interacting with the operating system.

random —Pseudorandom numbers.

string —String processing.

Thank You