PYTHON – Functions & Modules

Dr Sivabalan,

Technical Training Advisor

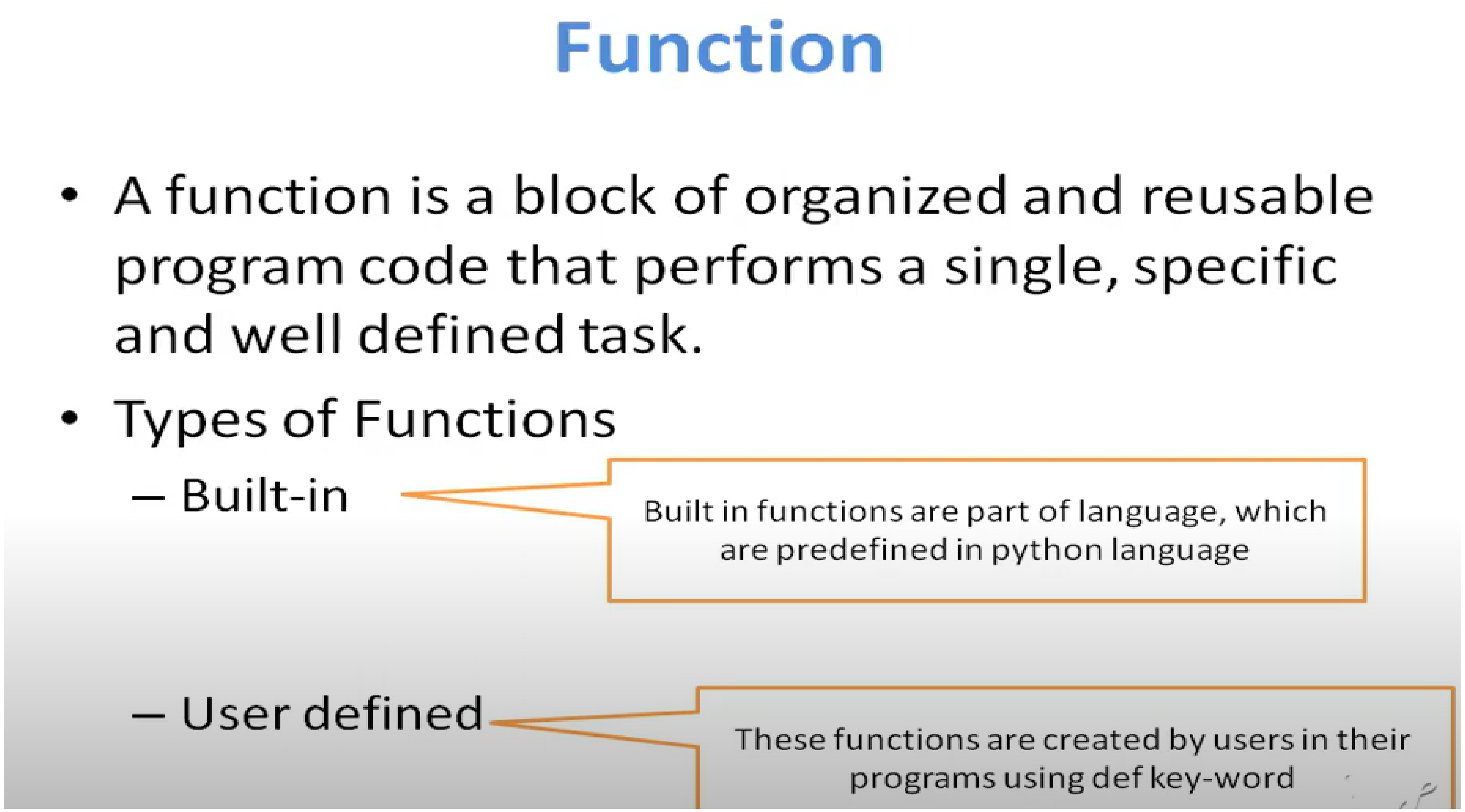
Sivabalan.n@nttdata.com

Agenda- Day 1

1. Need for a function
2. Function Definition
3. Types of Functions
4. Types of user defined functions
5. Return statement
6. Types of Variables
7. Variables scope and Lifetime
8. Types of Arguments
9. Lambda functions
10. Docs strings

Functions & Modules

Functions



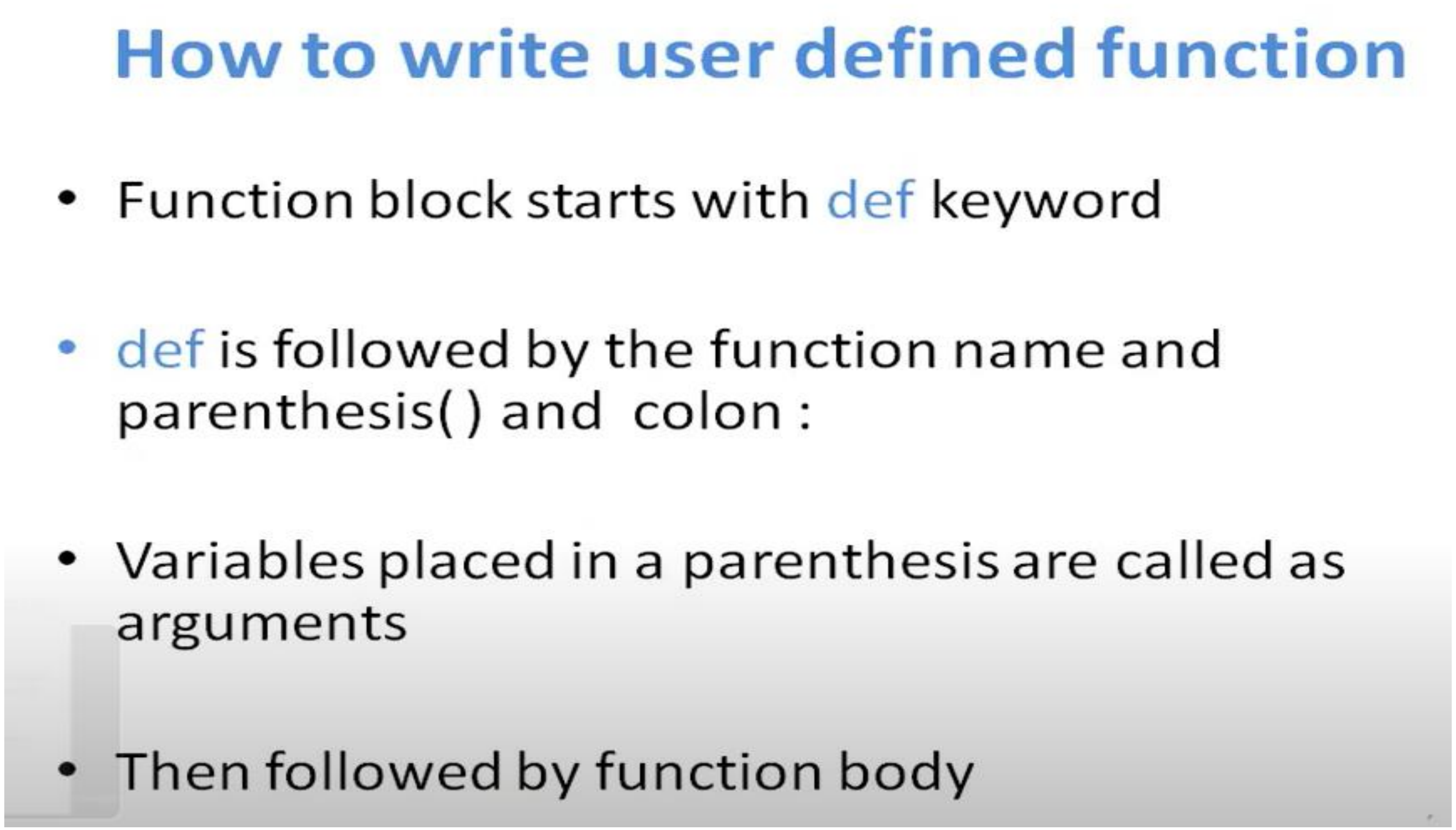
Function

- A function is a block of organized and reusable program code that performs a single, specific and well defined task.
- Types of Functions
  - Built-in
  - User defined

Built in functions are part of language, which are predefined in python language

These functions are created by users in their programs using def key-word

Functions

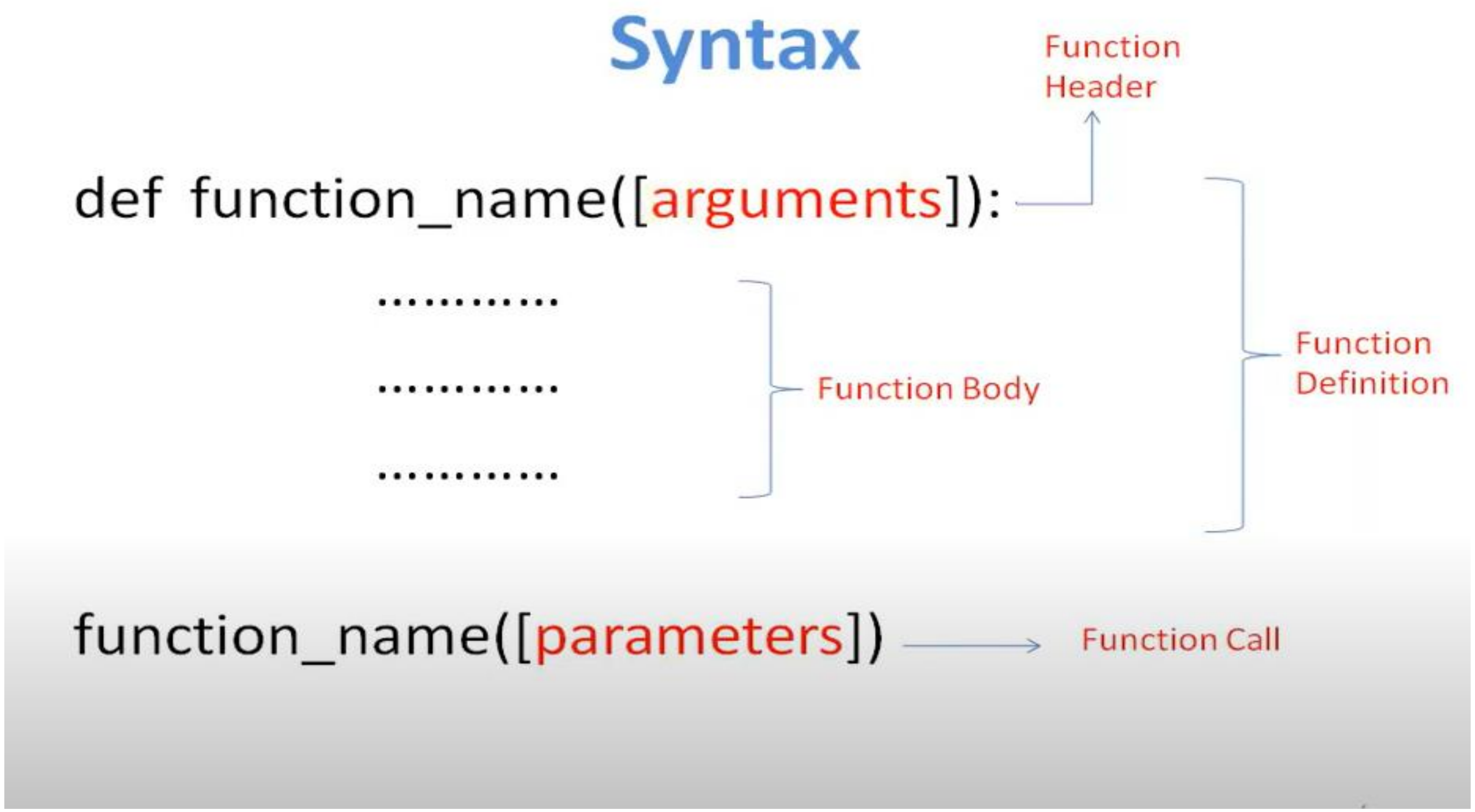


How to write user defined function

- Function block starts with def keyword
- def is followed by the function name and parenthesis( ) and colon :
- Variables placed in a parenthesis are called as arguments
- Then followed by function body

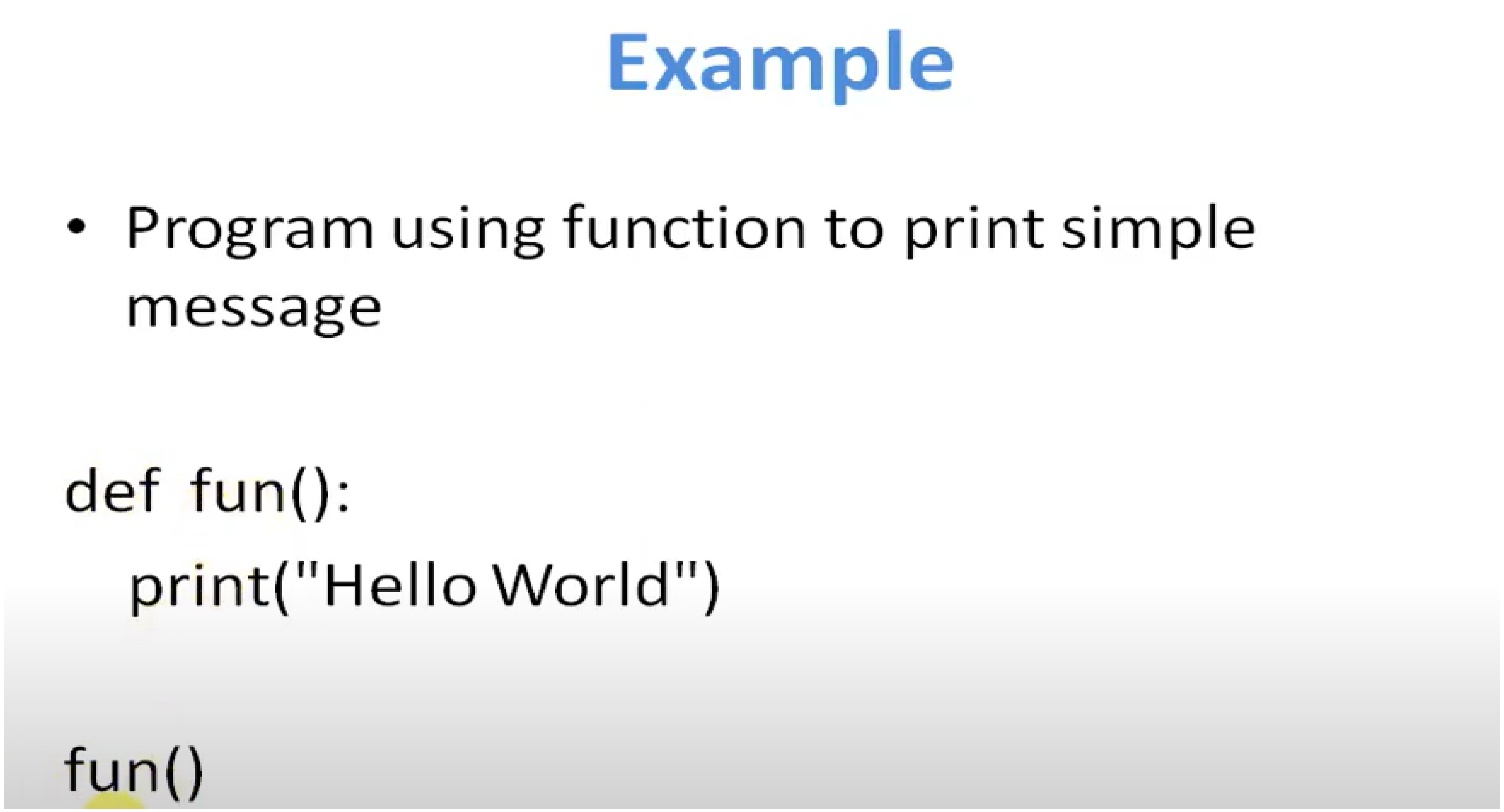
Functions

.


Syntax

```
def function_name([arguments]):
            ............
            ............
            ............

function_name([parameters])
```

Functions



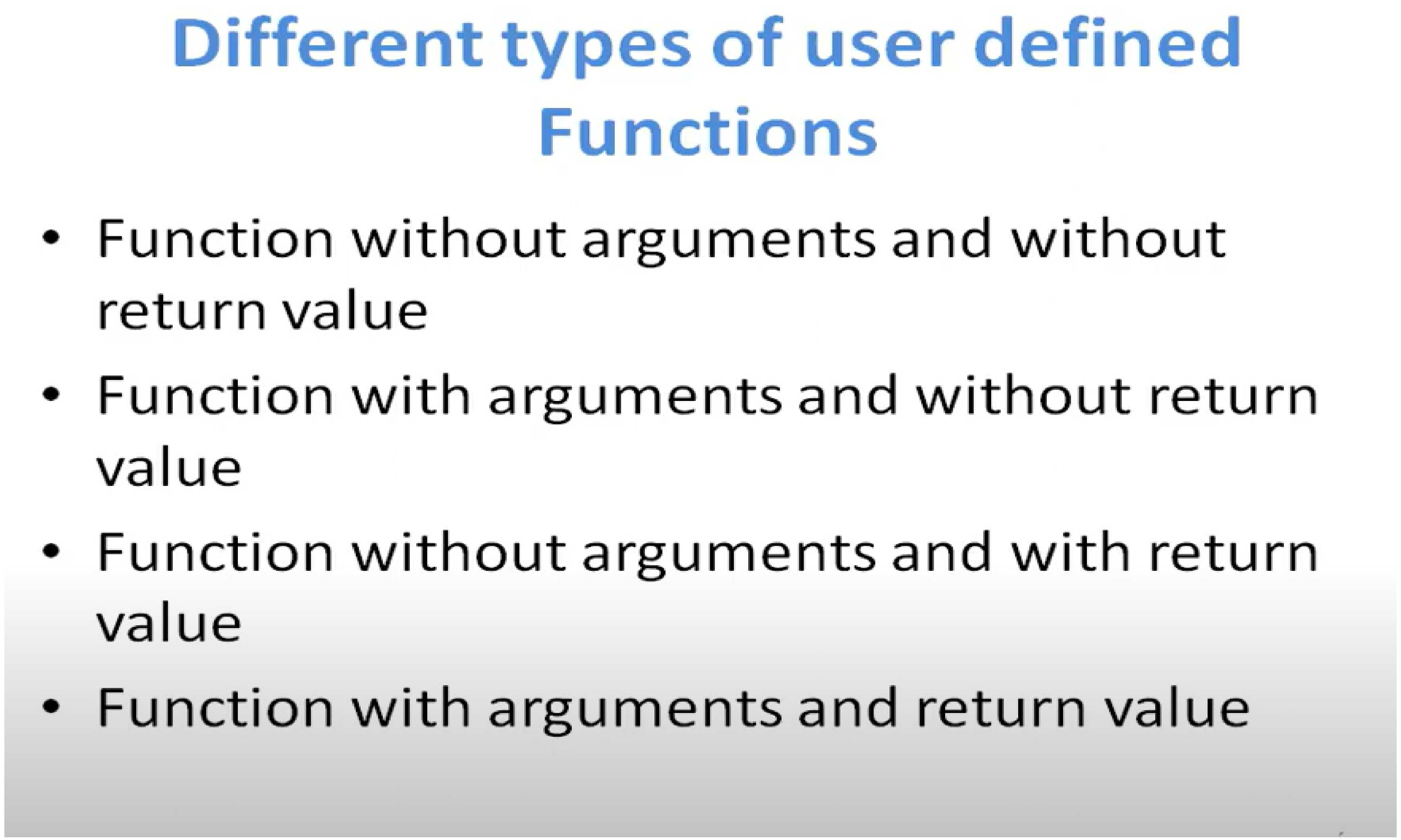
Example

- Program using function to print simple message

```
def  fun():
    print("Hello World")

fun()
```

Functions

.

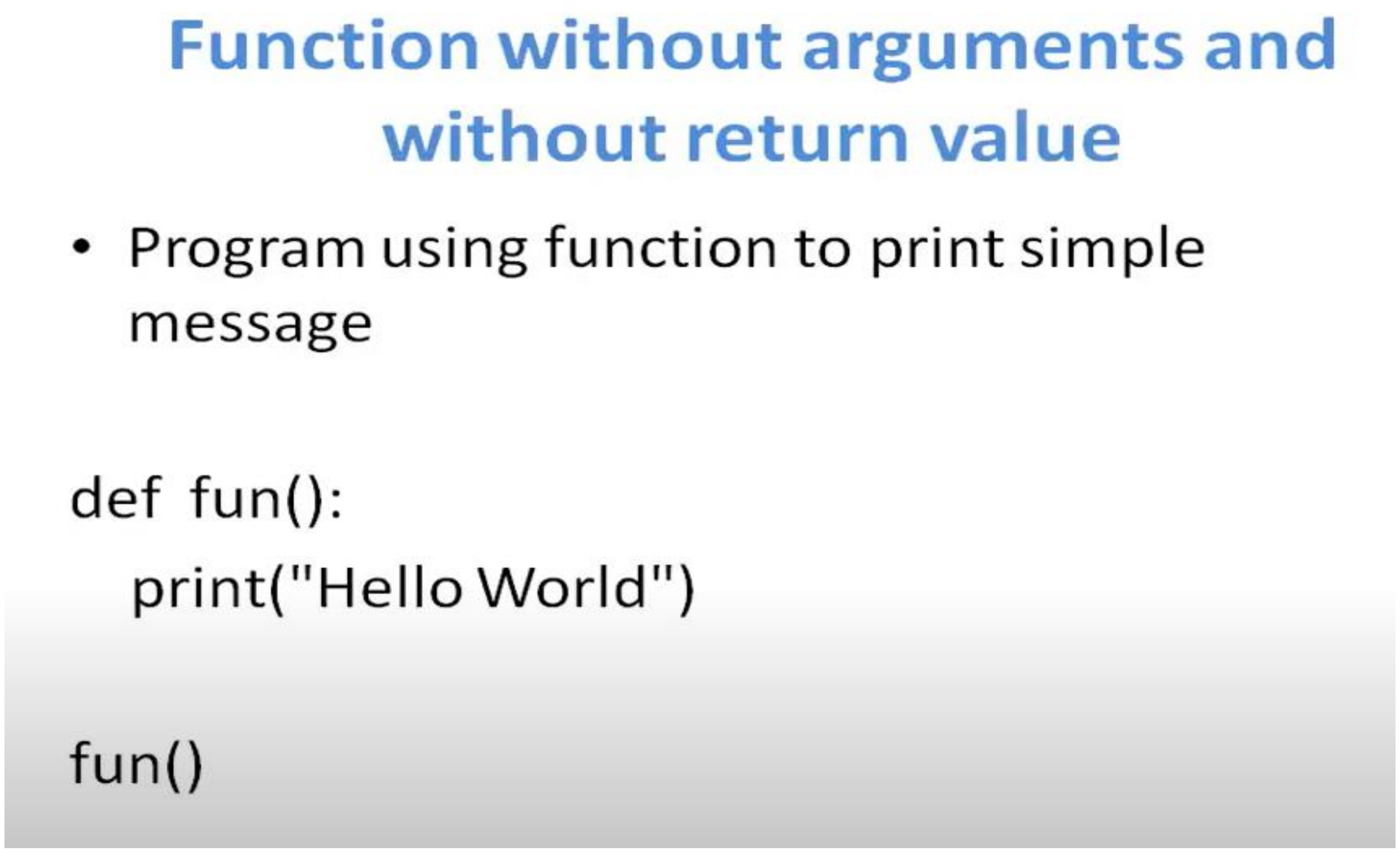


Different types of user defined Functions

- Function without arguments and without return value
- Function with arguments and without return value
- Function without arguments and with return value
- Function with arguments and return value

Functions

.

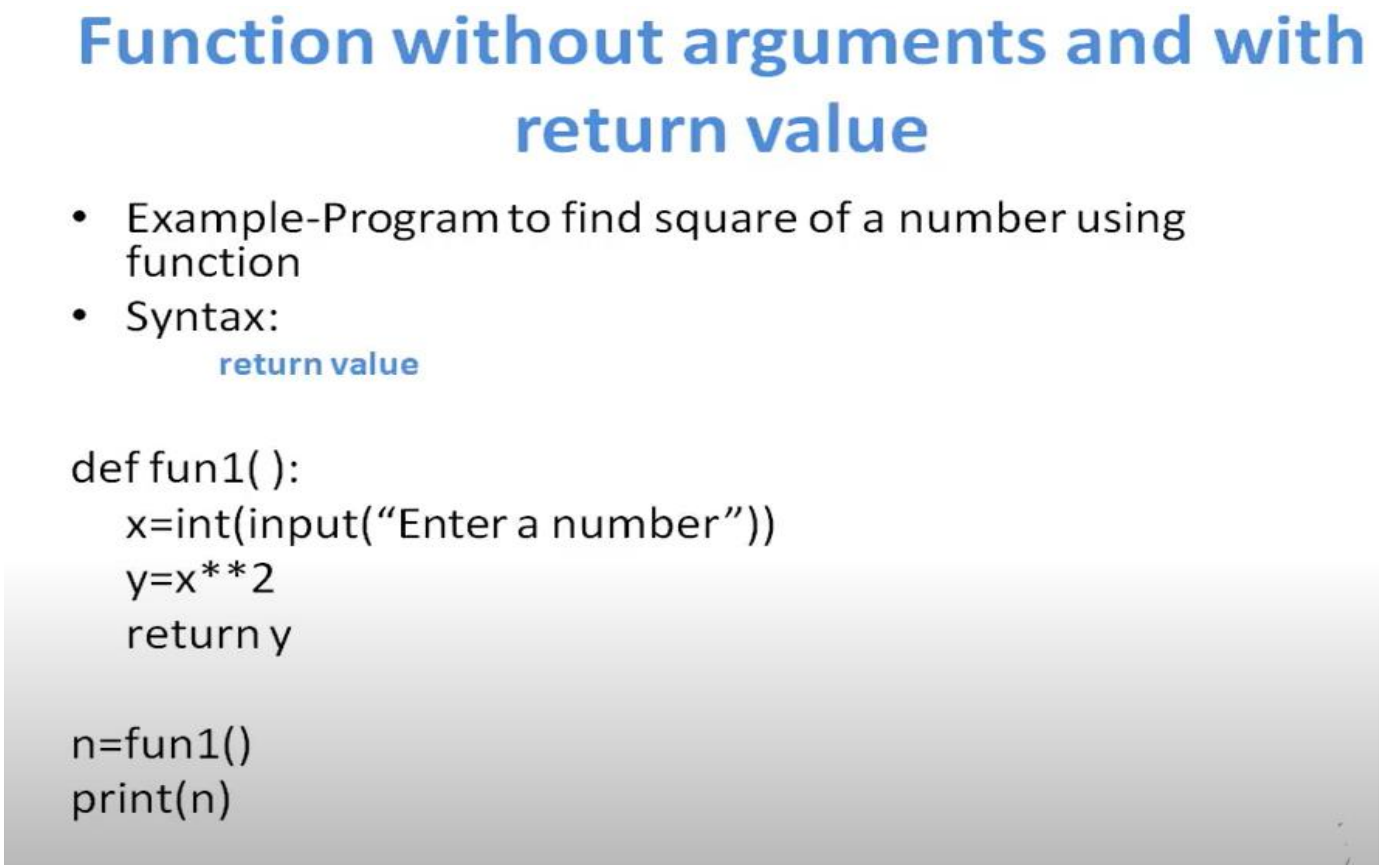


Function without arguments and without return value

- Program using function to print simple message

```
def  fun():
   print("Hello World")

fun()
```

Functions



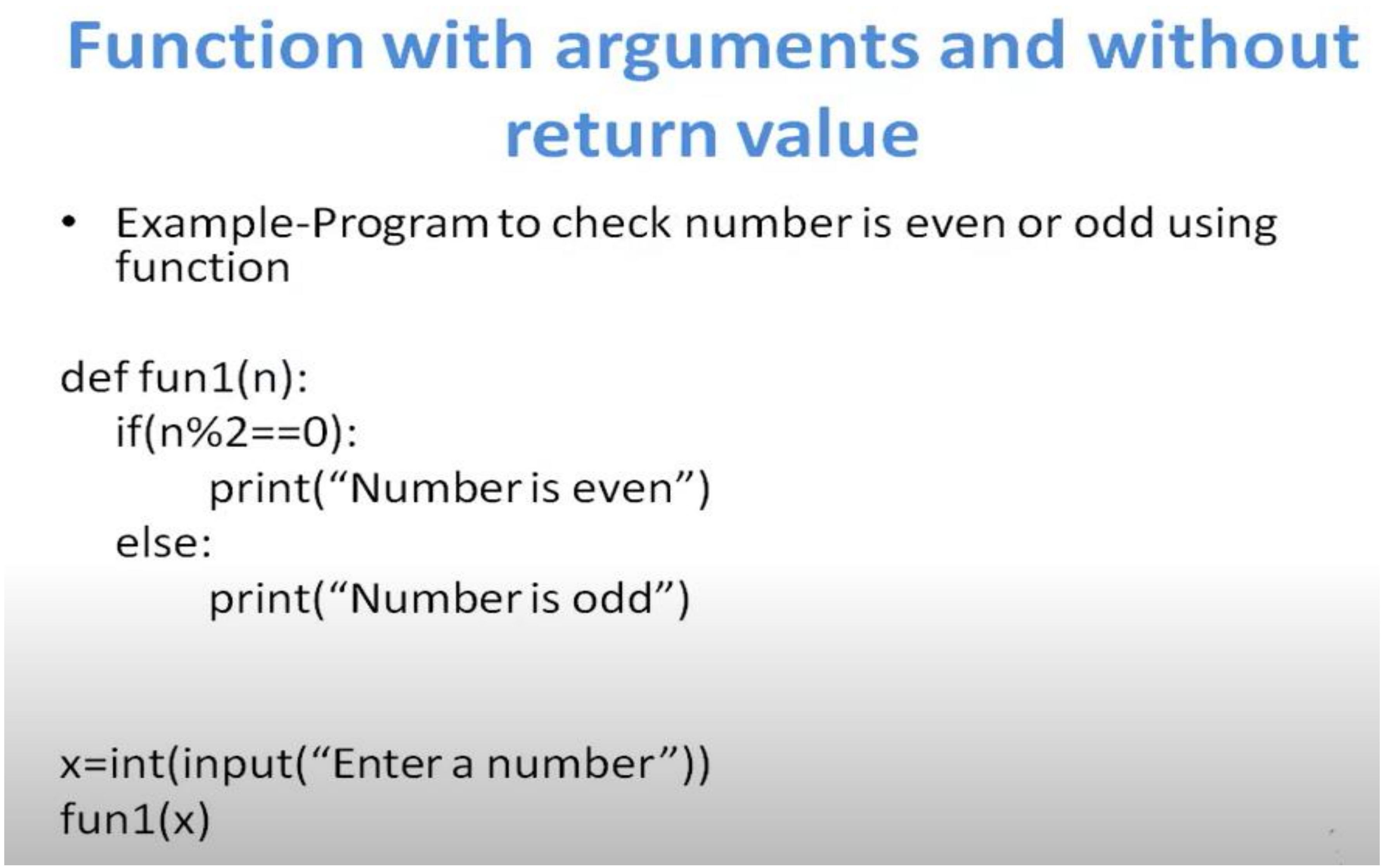
Function without arguments and with return value

- Example-Program to find square of a number using function
- Syntax:

    return value

```
def fun1( ):
    x=int(input("Enter a number"))
    y=x**2
    return y

n=fun1()
print(n)
```

Functions

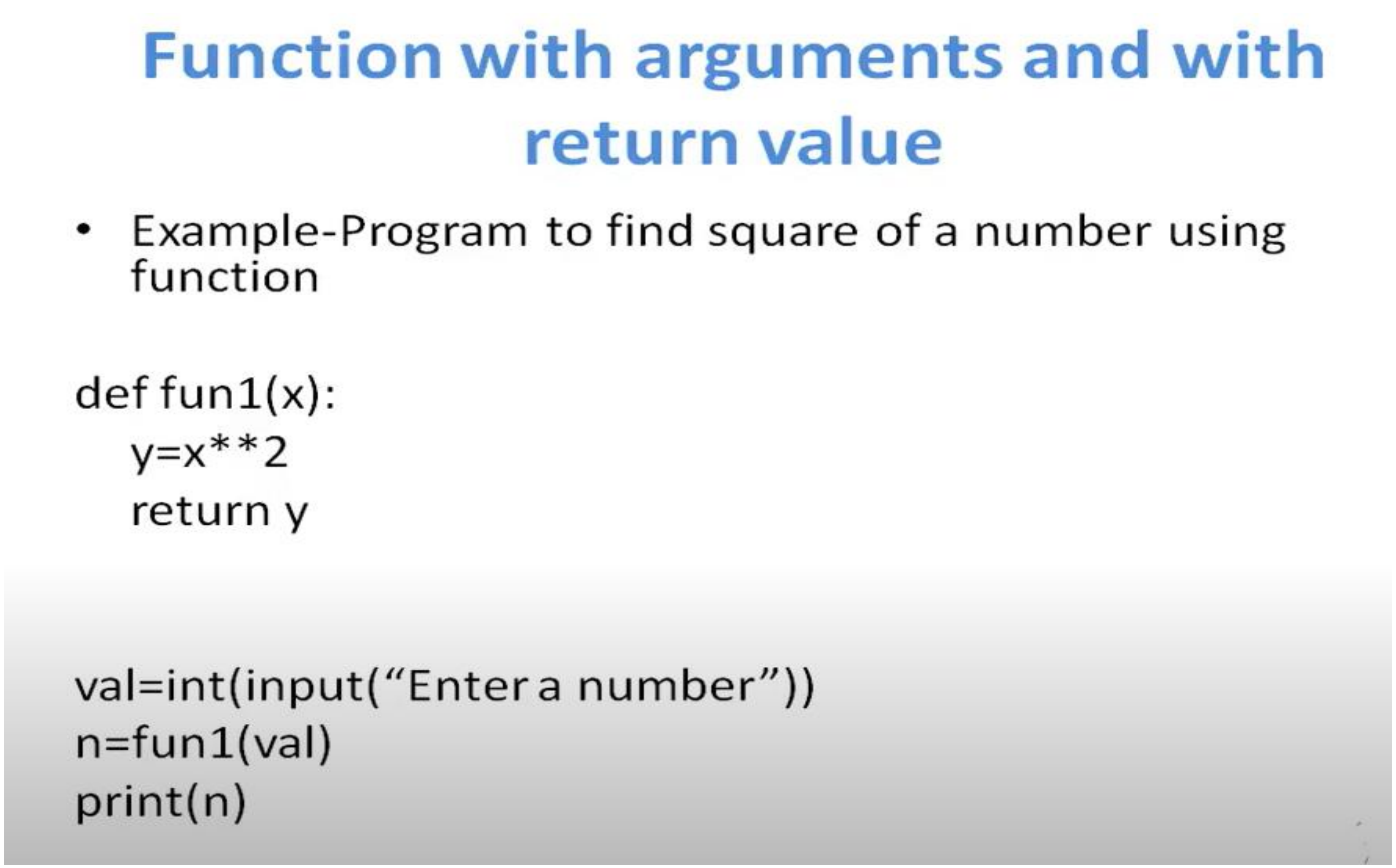


Function with arguments and without return value

- Example-Program to check number is even or odd using function

```
def fun1(n):
    if(n%2==0):
            print(“Number is even”)
    else:
            print(“Number is odd”)


x=int(input(“Enter a number”))
fun1(x)
```

Functions

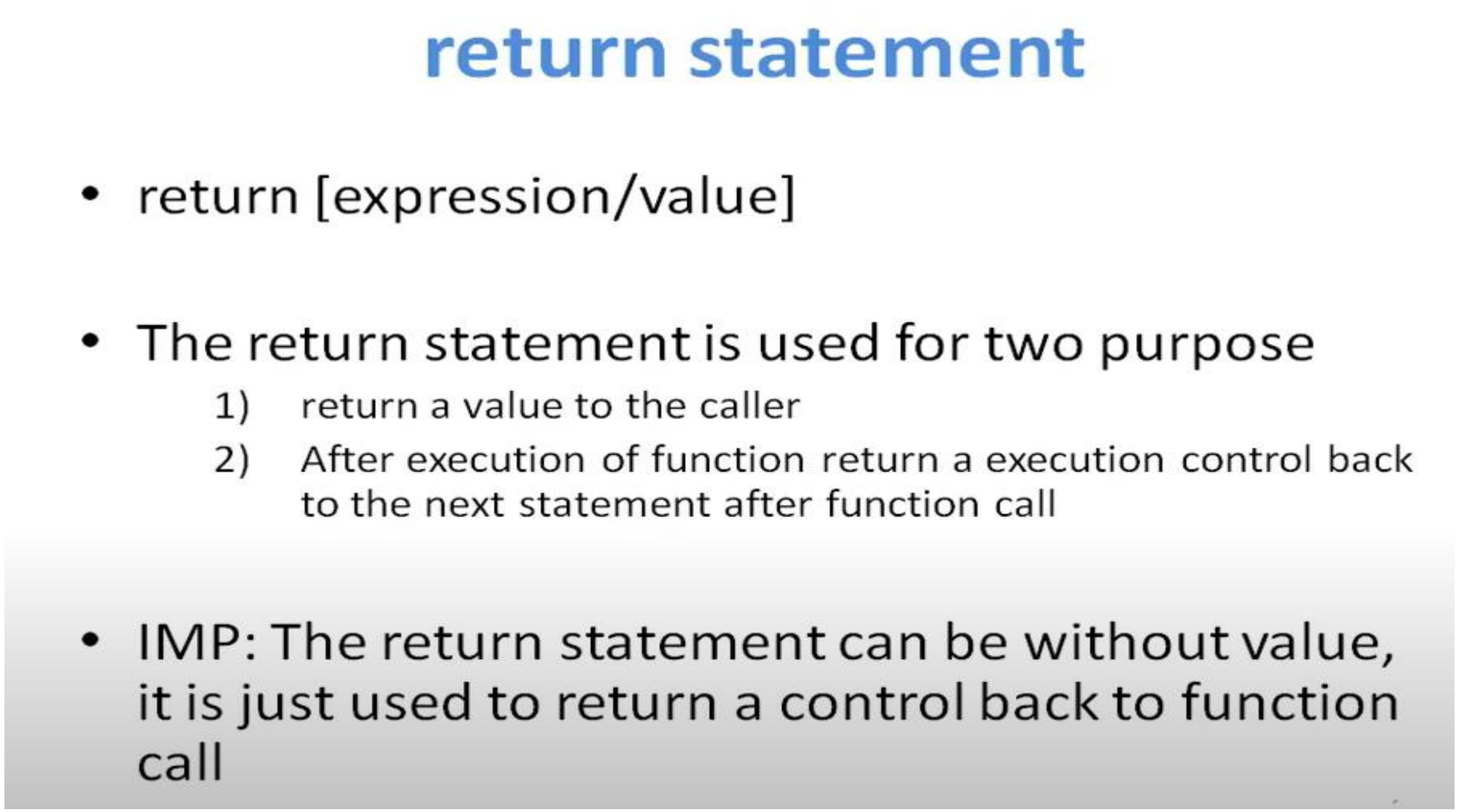


Function with arguments and with return value

- Example-Program to find square of a number using function

```
def fun1(x):
    y=x**2
    return y


val=int(input("Enter a number"))
n=fun1(val)
print(n)
```

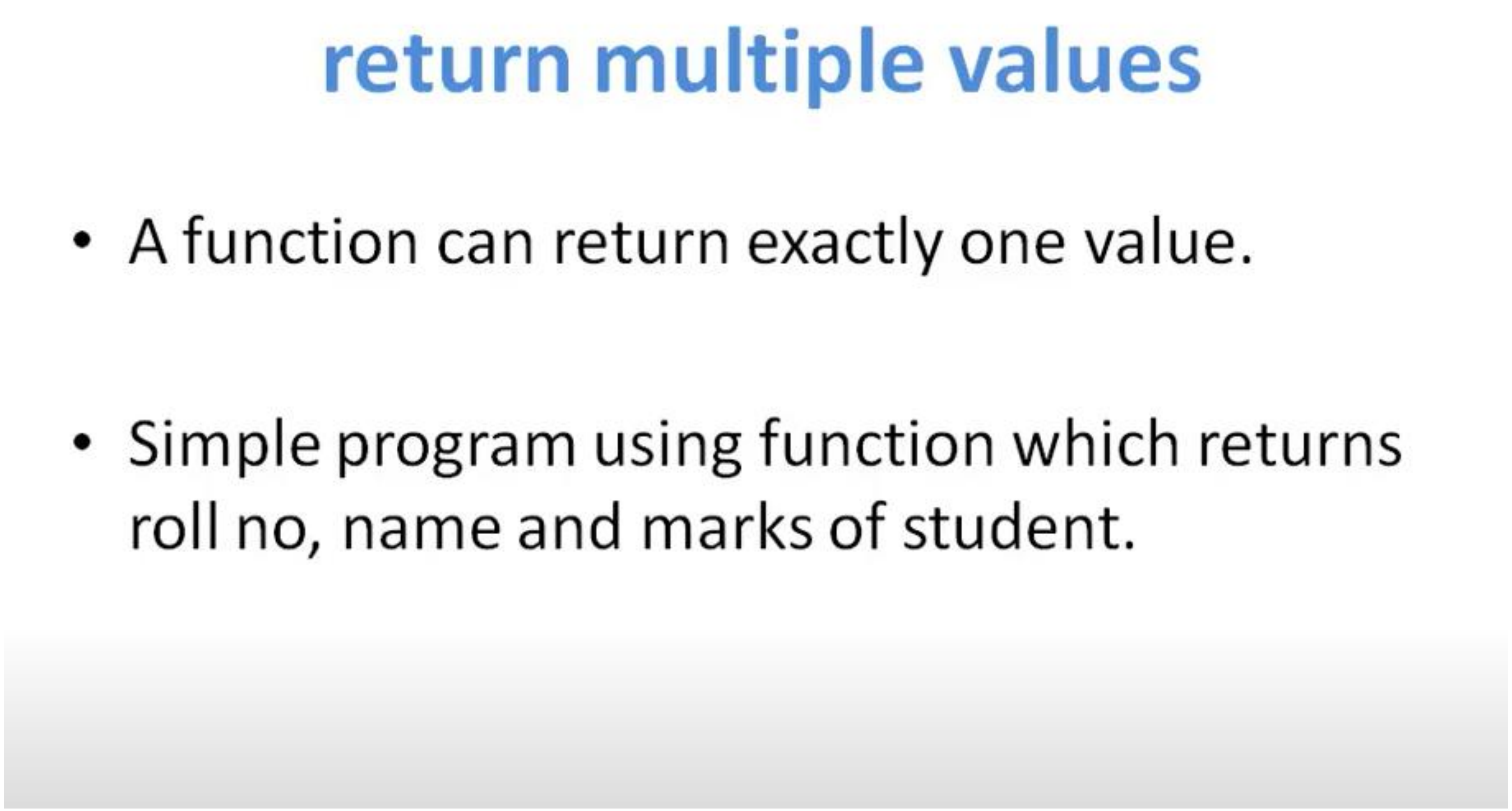
Functions



return statement

- return [expression/value]
- The return statement is used for two purpose
  1) return a value to the caller
  2) After execution of function return a execution control back to the next statement after function call
- IMP: The return statement can be without value, it is just used to return a control back to function call

Functions

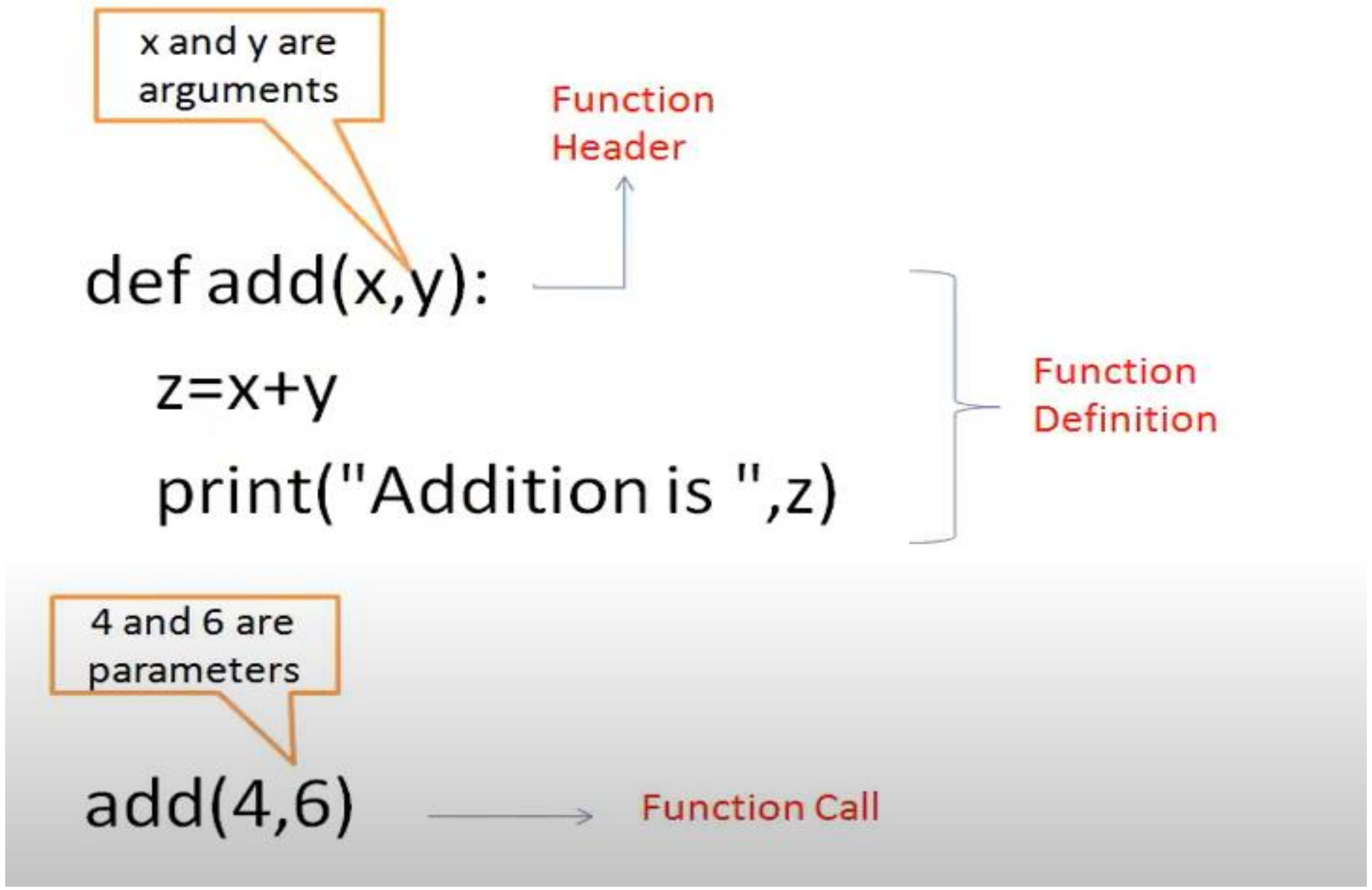


return multiple values

- A function can return exactly one value.
- Simple program using function which returns roll no, name and marks of student.

Functions

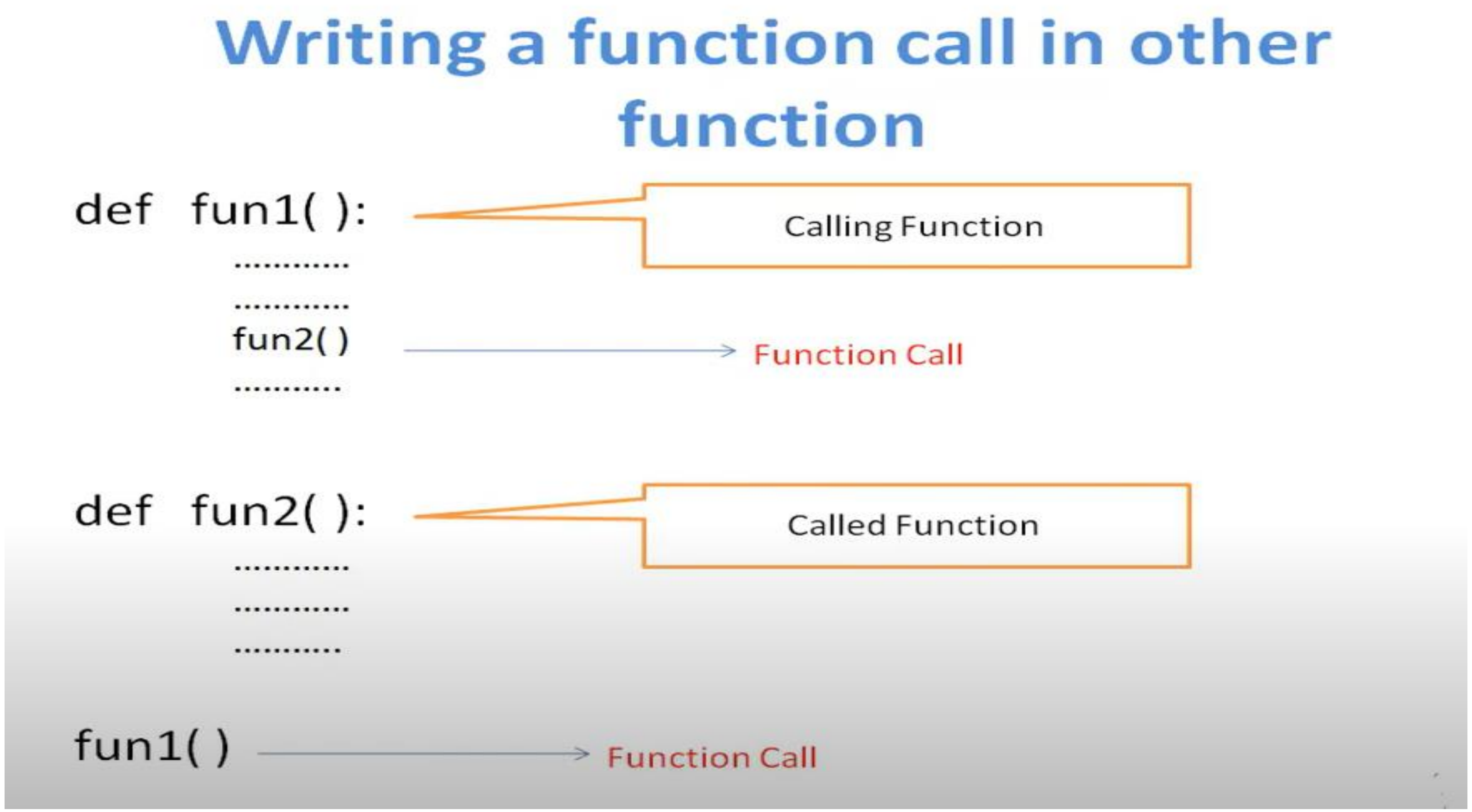
.



```
def add(x,y):
    z=x+y
    print("Addition is ",z)

add(4,6)
```

Functions

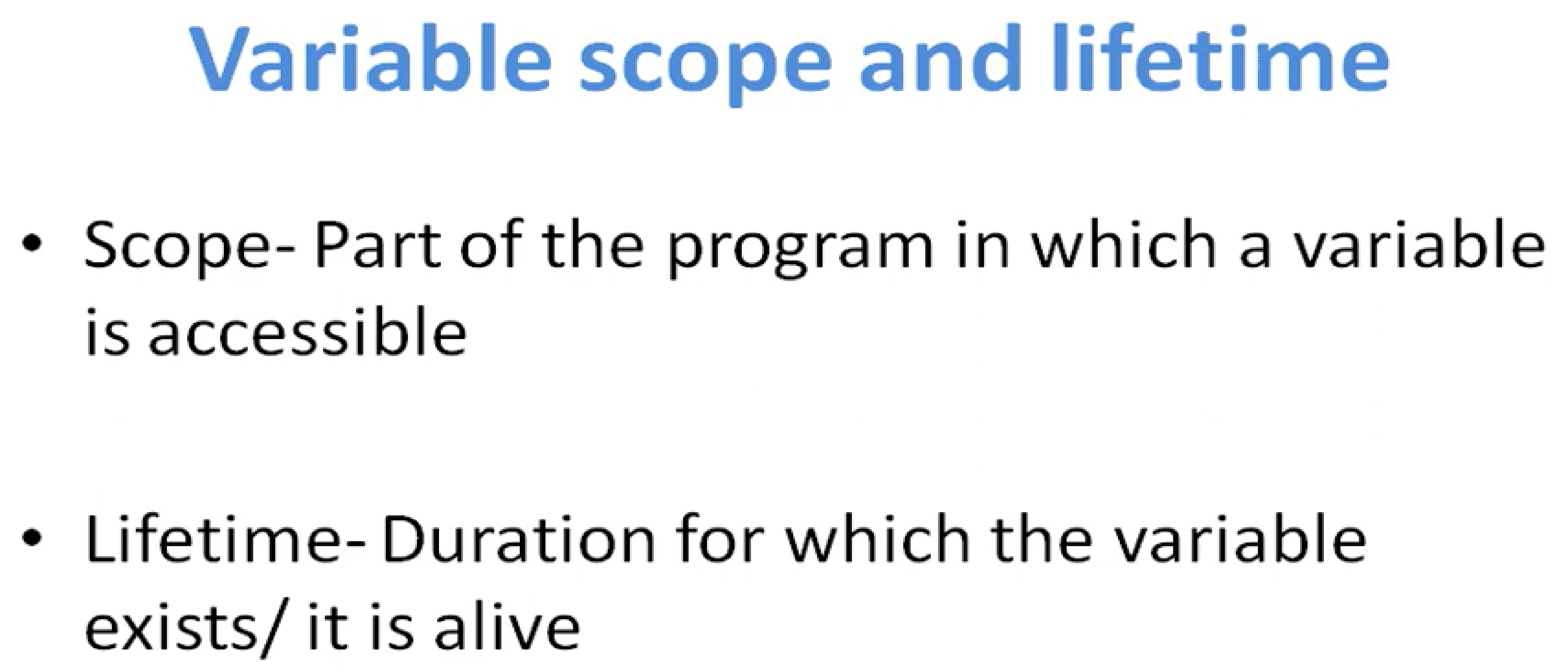
.



Writing a function call in other function

```
def  fun1( ):
        ............
        ............
        fun2( )
        ...........

def  fun2( ):
        ............
        ............
        ...........

fun1( )
```

Functions



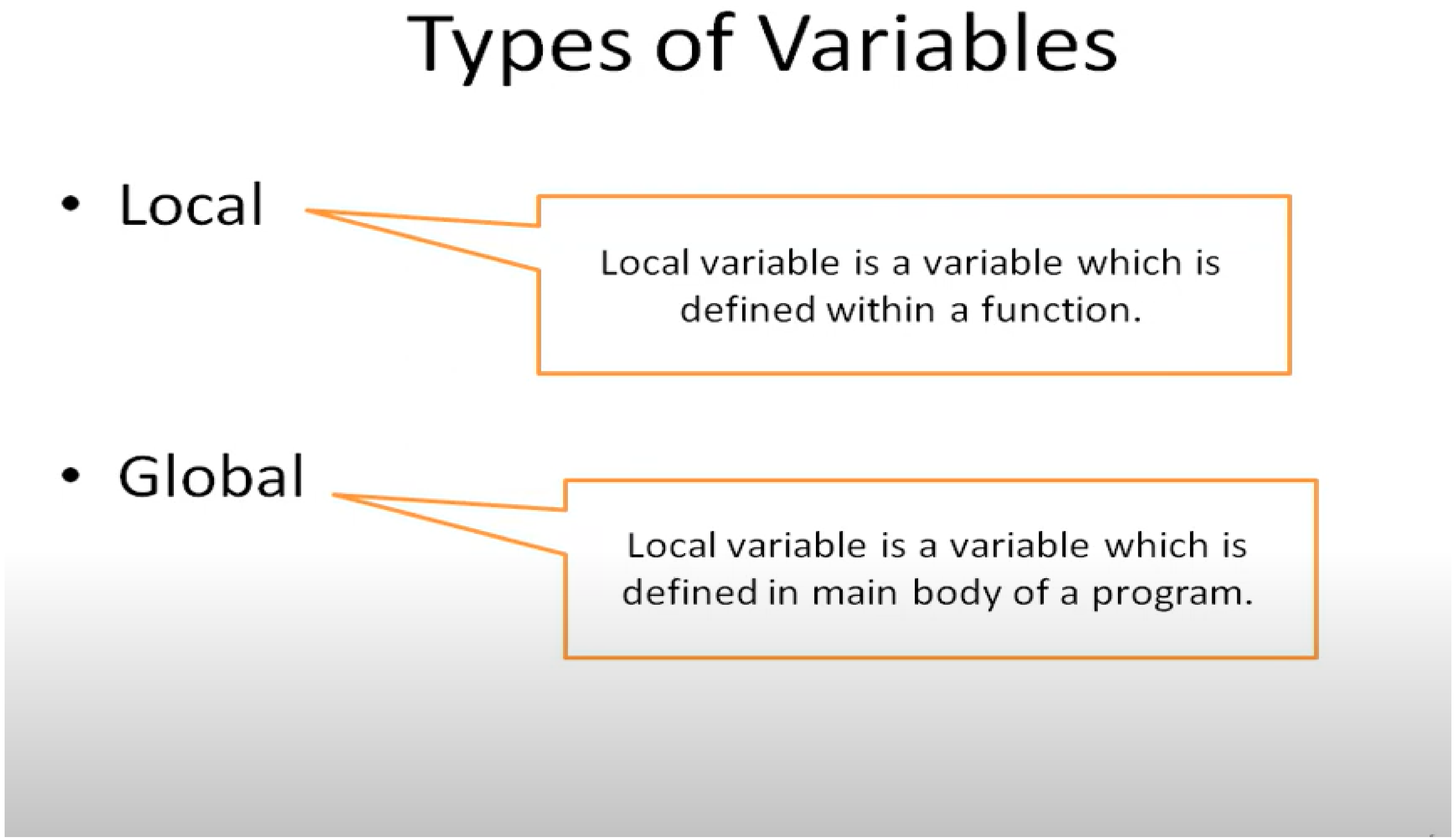
Variable scope and lifetime

- Scope- Part of the program in which a variable is accessible

- Lifetime- Duration for which the variable exists/ it is alive

Functions

.


Types of Variables

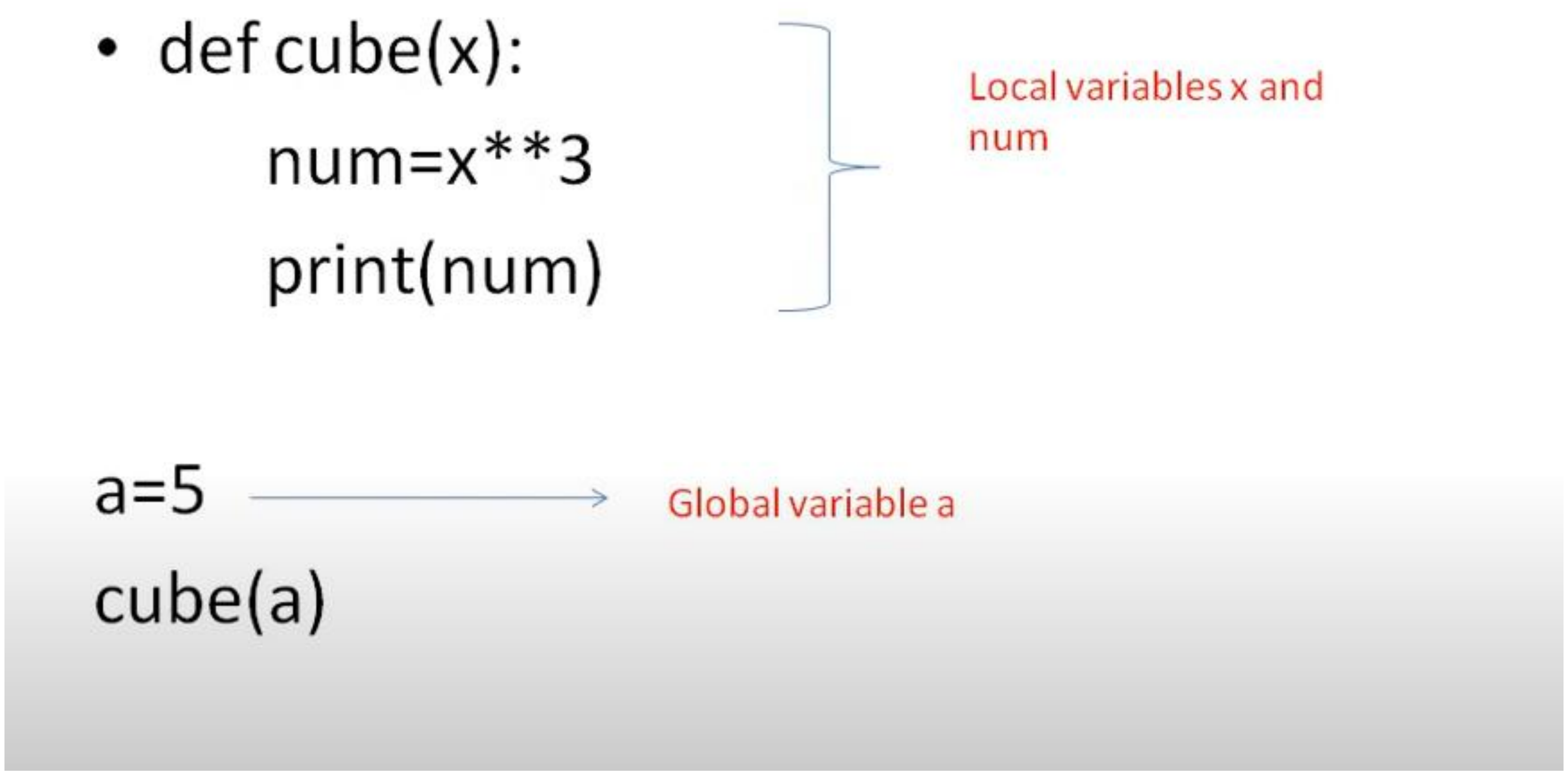
- Local

Local variable is a variable which is defined within a function.

- Global

Local variable is a variable which is defined in main body of a program.

Functions

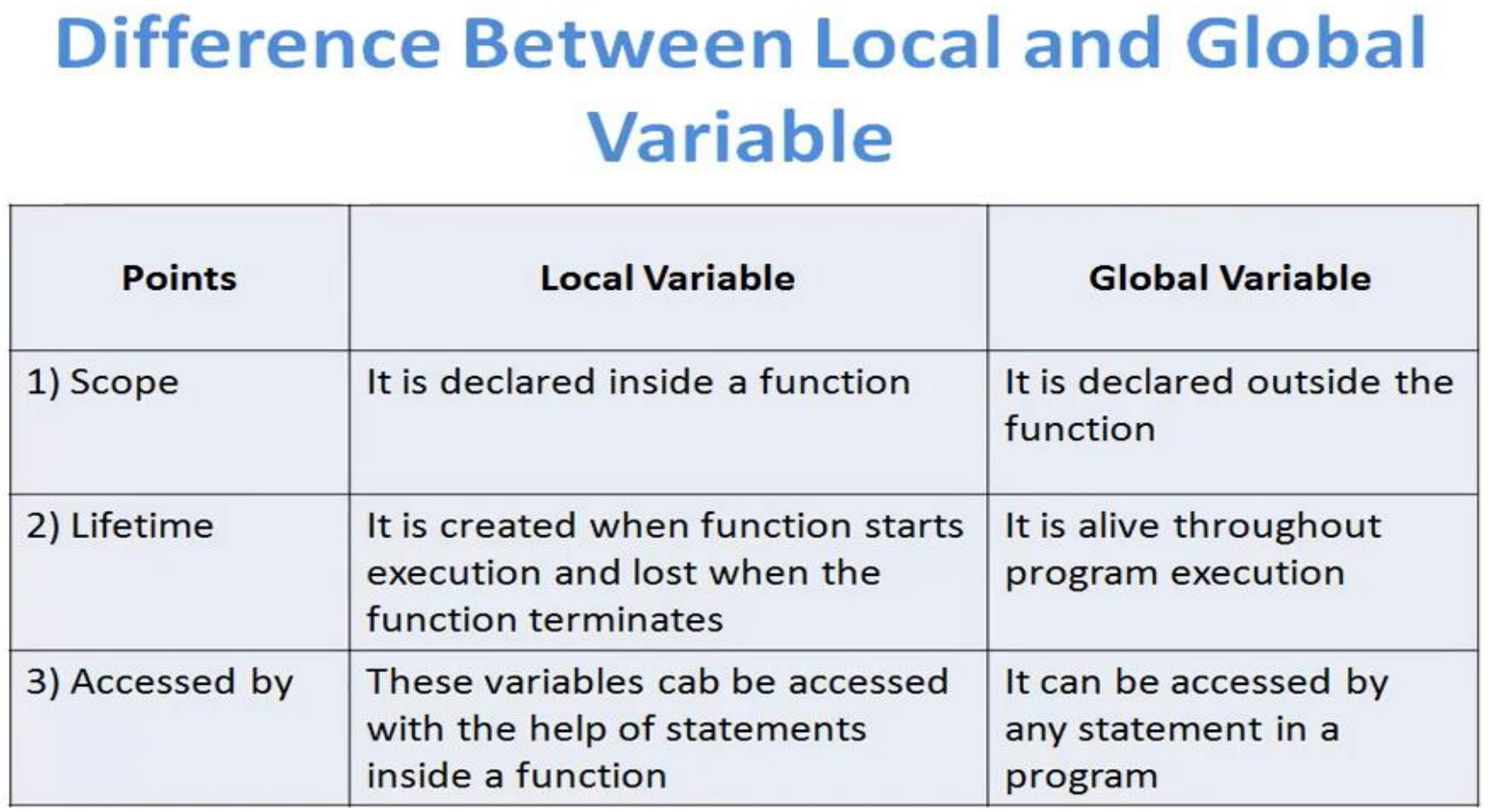


```
• def cube(x):
        num=x**3
        print(num)

a=5
cube(a)
```

Local variables x and num

Global variable a

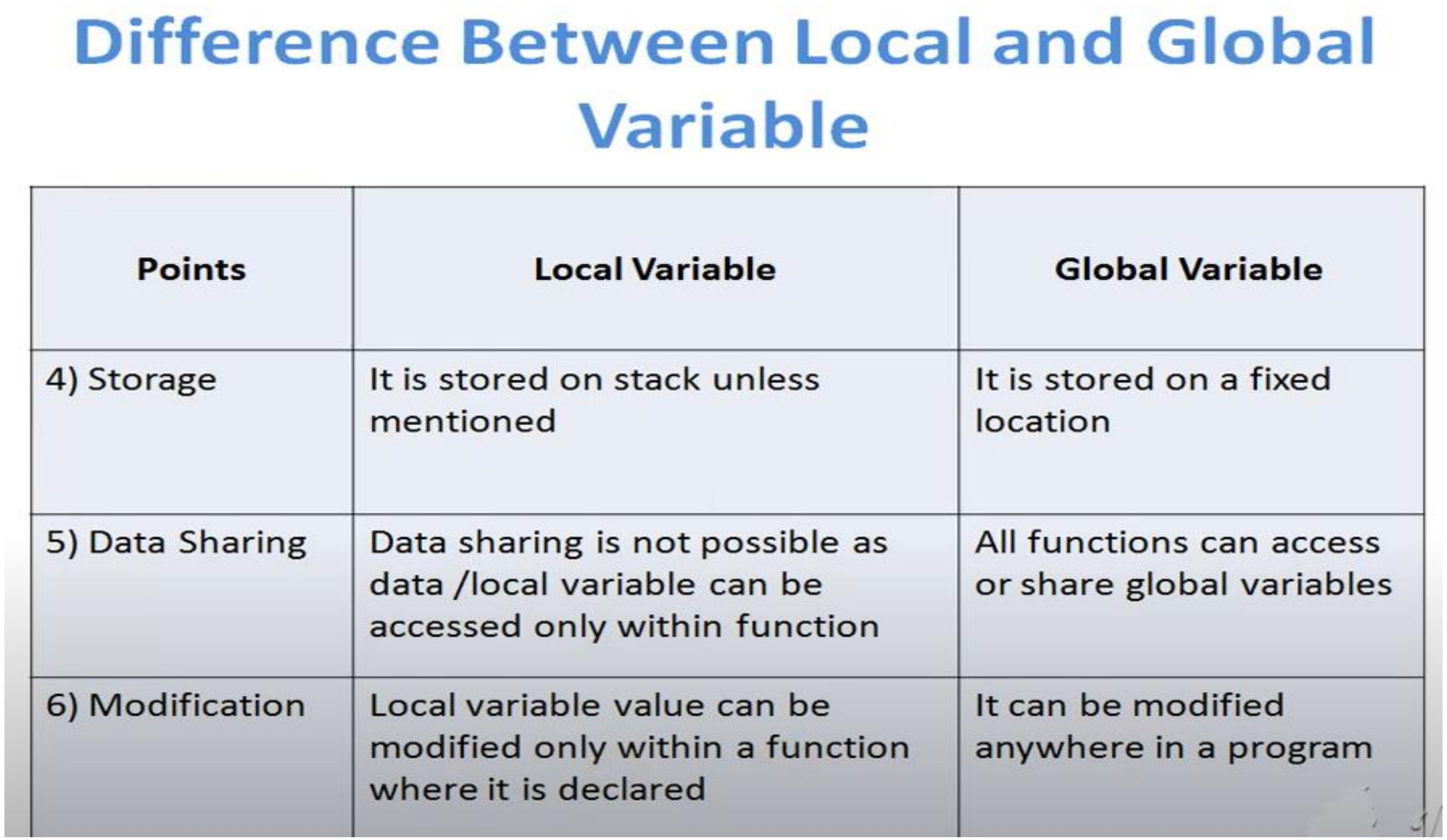
Functions



Difference Between Local and Global Variable

Points	Local Variable	Global Variable
1) Scope	It is declared inside a function	It is declared outside the function
2) Lifetime	It is created when function starts execution and lost when the function terminates	It is alive throughout program execution
3) Accessed by	These variables cab be accessed with the help of statements inside a function	It can be accessed by any statement in a program

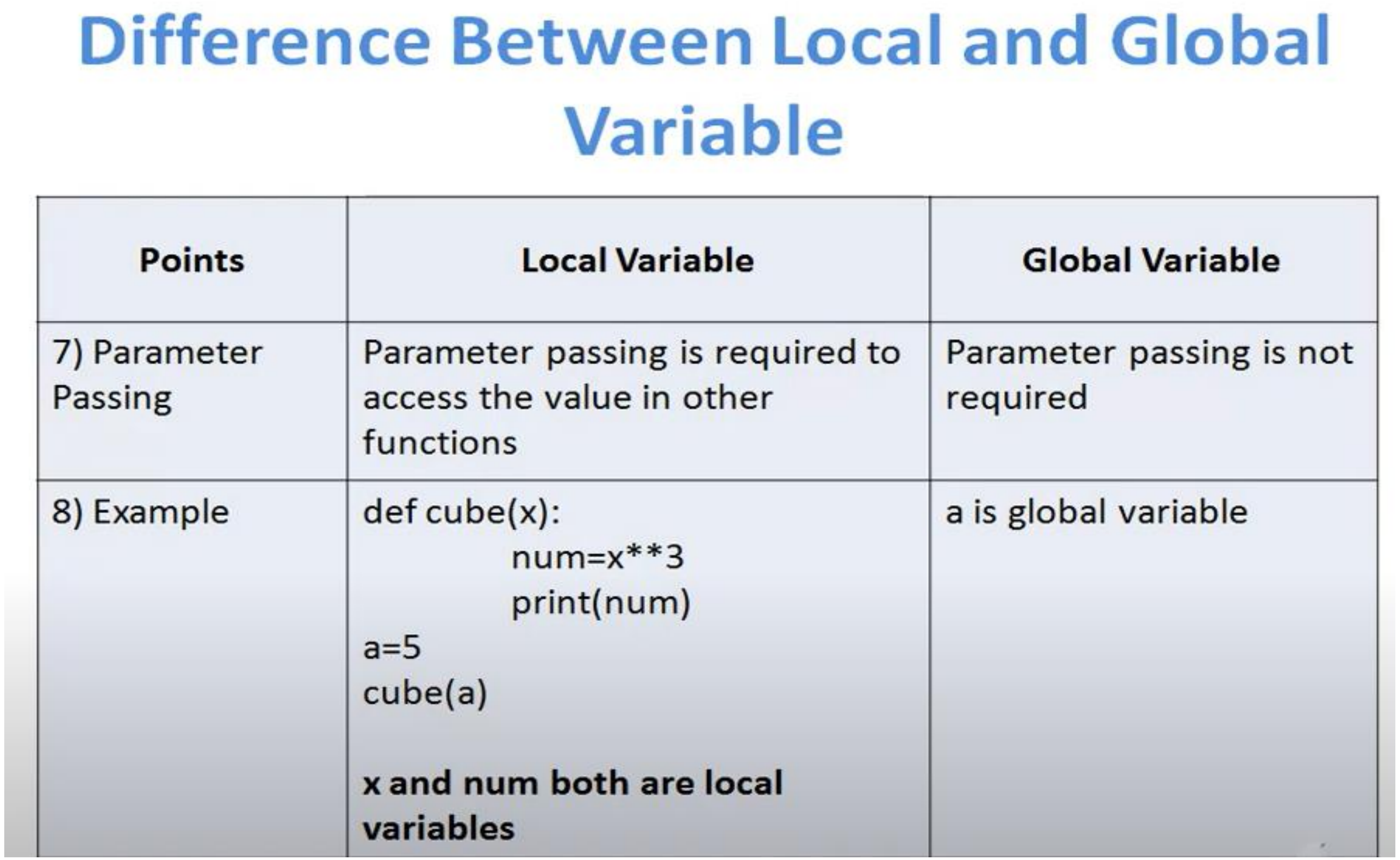
Functions



Difference Between Local and Global Variable

Points	Local Variable	Global Variable
4) Storage	It is stored on stack unless mentioned	It is stored on a fixed location
5) Data Sharing	Data sharing is not possible as data /local variable can be accessed only within function	All functions can access or share global variables
6) Modification	Local variable value can be modified only within a function where it is declared	It can be modified anywhere in a program

Functions

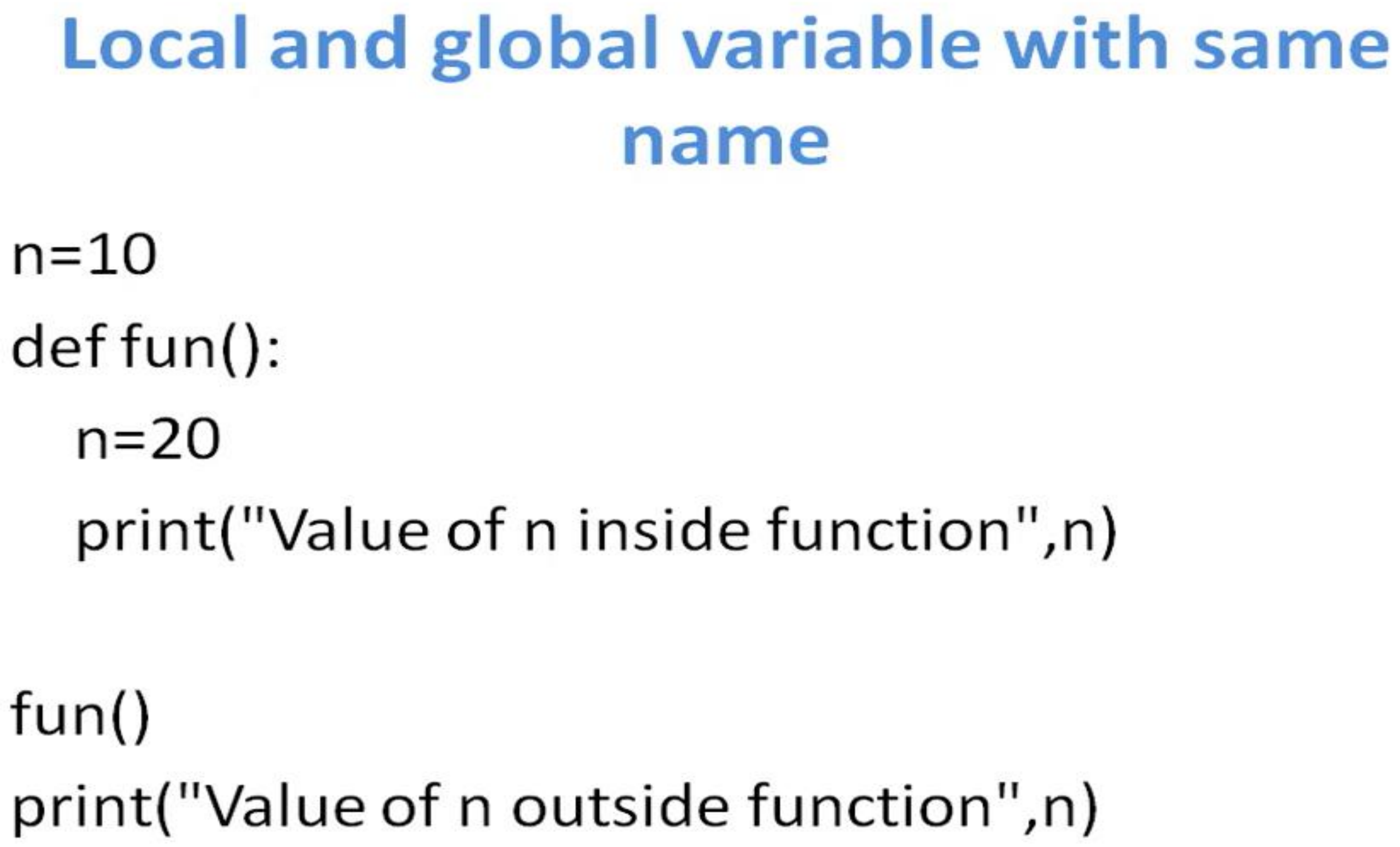


Difference Between Local and Global Variable

Points	Local Variable	Global Variable
7) Parameter Passing	Parameter passing is required to access the value in other functions	Parameter passing is not required

8) Example	def cube(x):
	        num=x**3
	        print(num)
	a=5
	cube(a)
	
	x and num both are local variables
		a is global variable

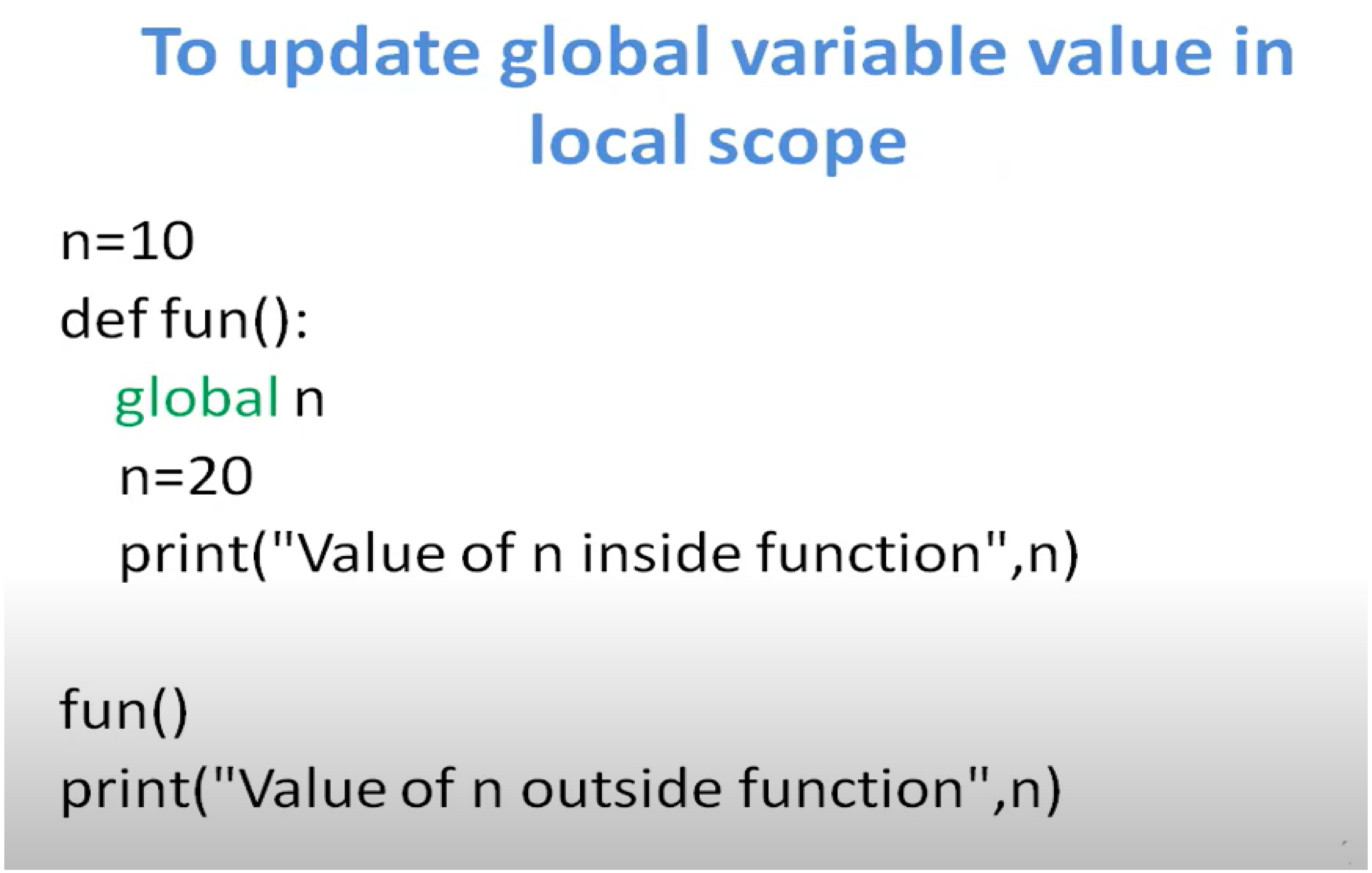
Functions



Local and global variable with same name

```
n=10
def fun():
    n=20
    print("Value of n inside function",n)

fun()
print("Value of n outside function",n)
```

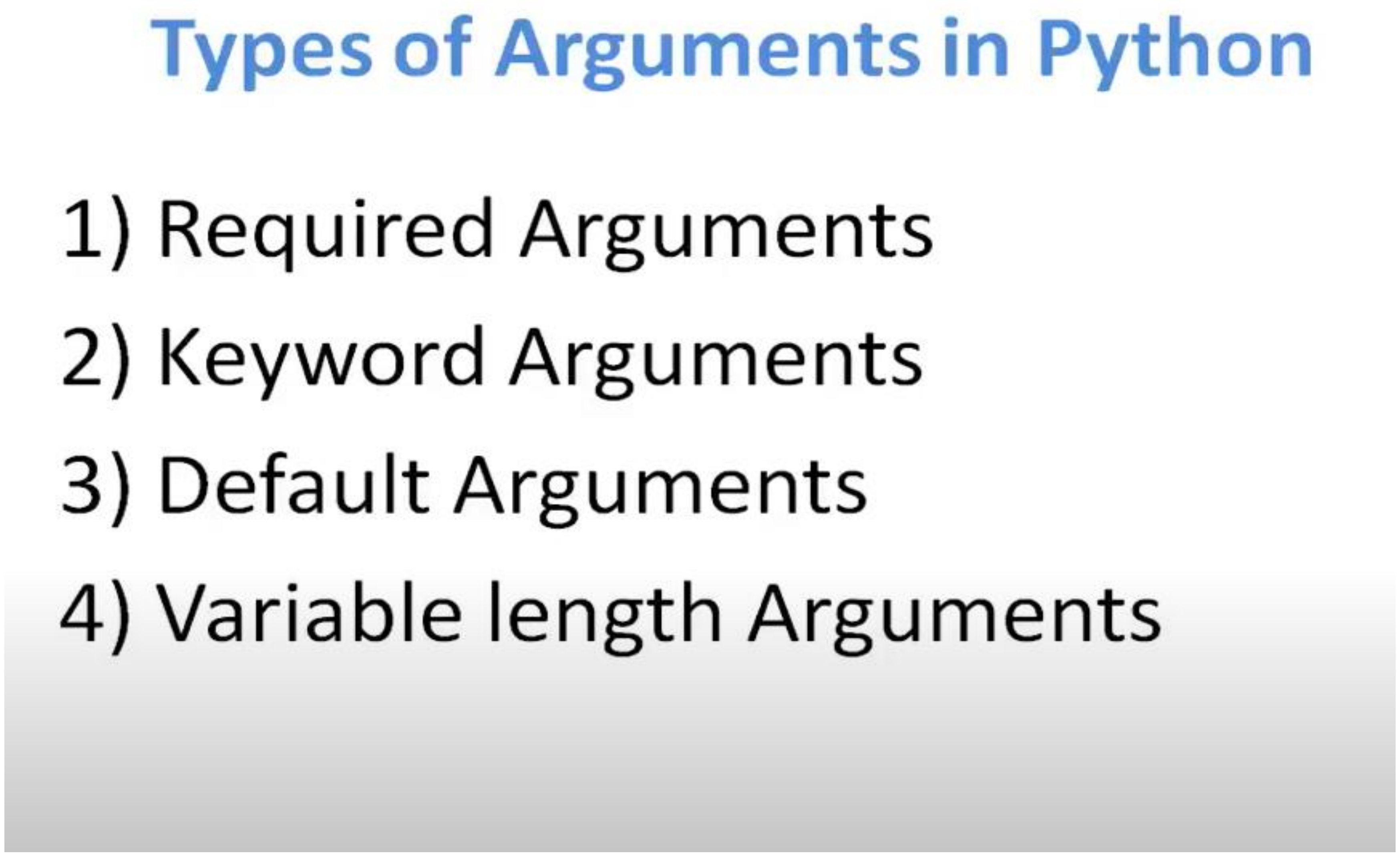
Functions



To update global variable value in local scope

```
n=10
def fun():
    global n
    n=20
    print("Value of n inside function",n)

fun()
print("Value of n outside function",n)
```

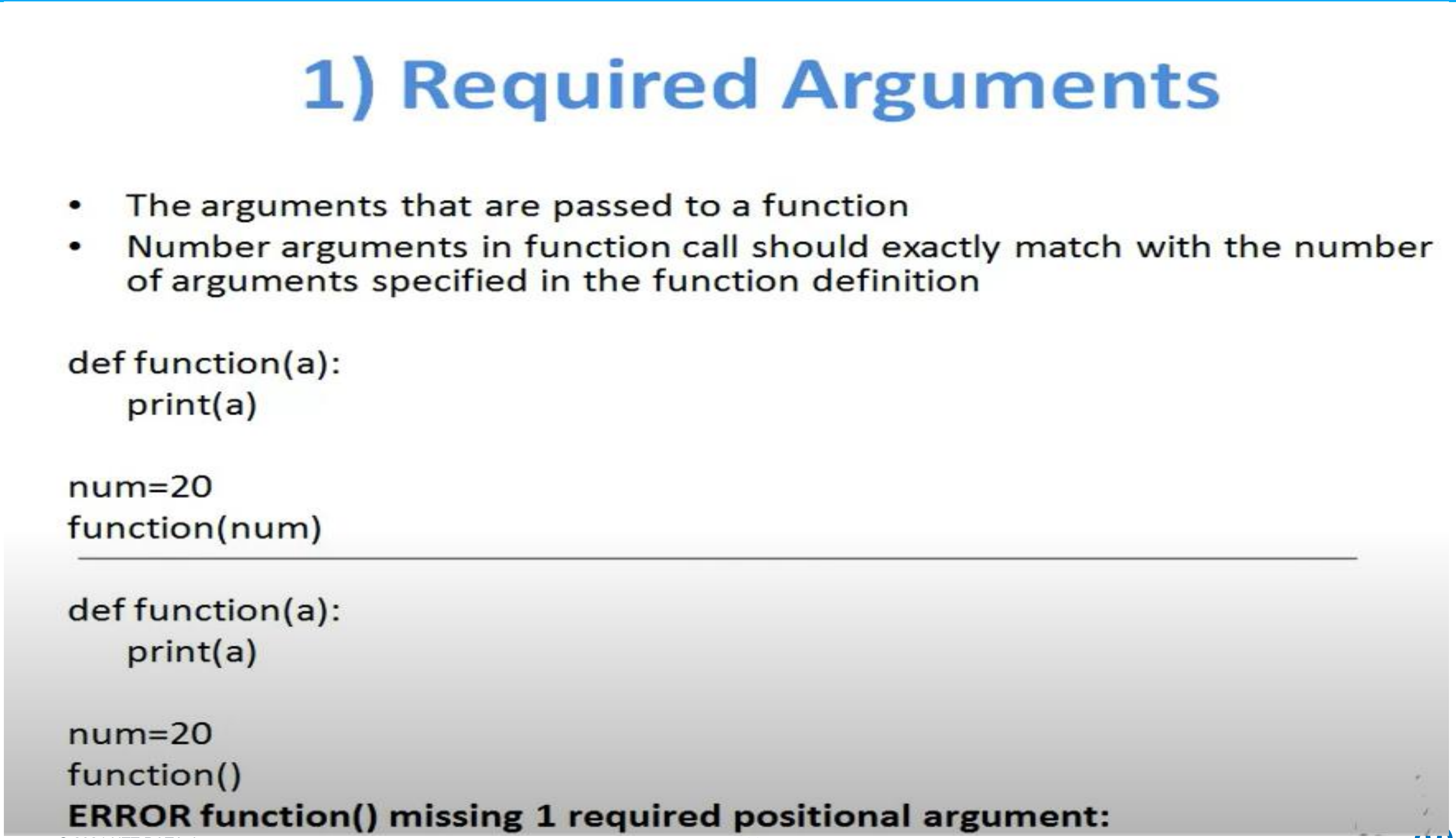
Functions



Types of Arguments in Python

1) Required Arguments
2) Keyword Arguments
3) Default Arguments
4) Variable length Arguments

Functions



1) Required Arguments

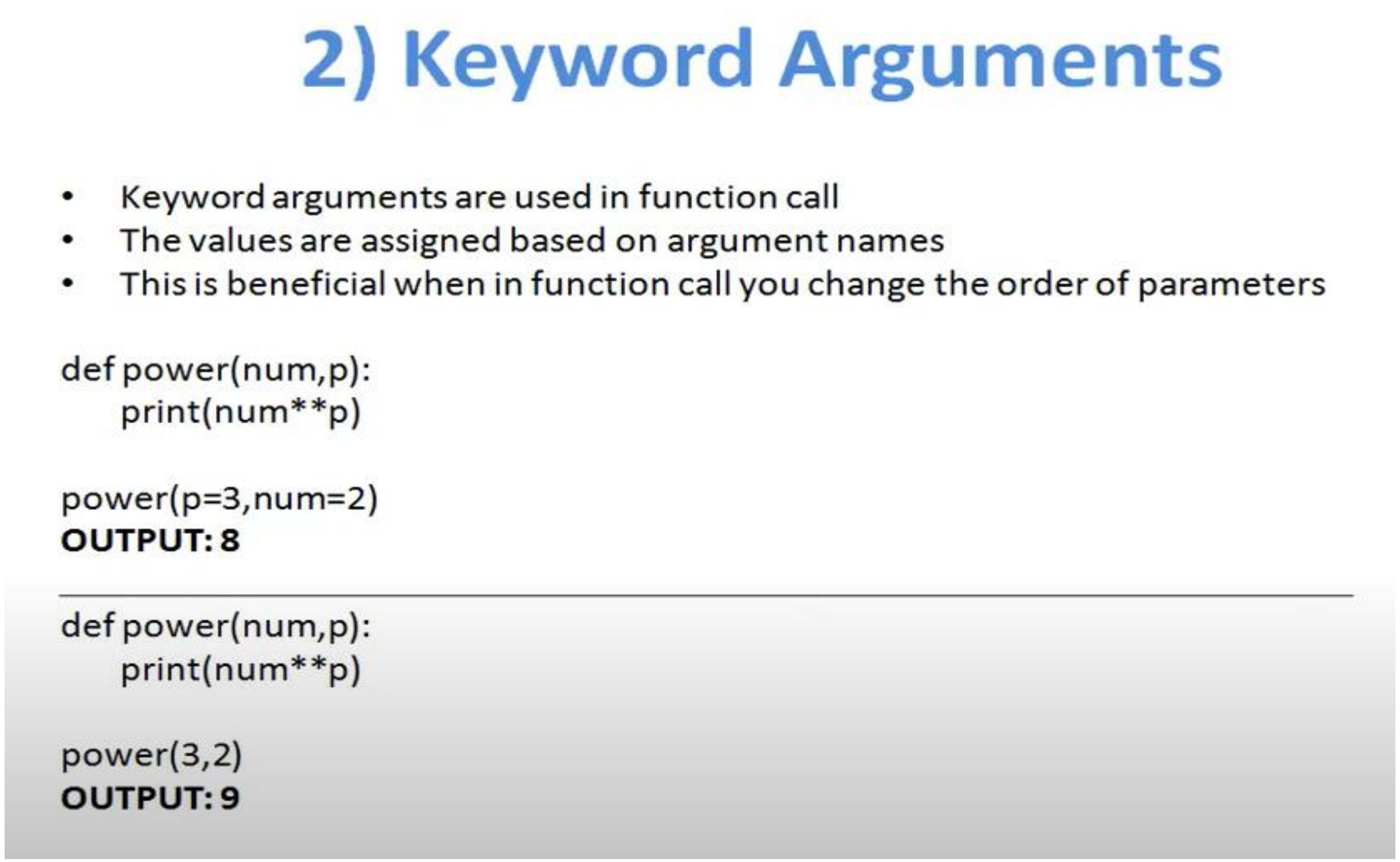
- The arguments that are passed to a function
- Number arguments in function call should exactly match with the number of arguments specified in the function definition

```
def function(a):
    print(a)

num=20
function(num)
```

```
def function(a):
    print(a)

num=20
function()
```

ERROR function() missing 1 required positional argument:

Functions



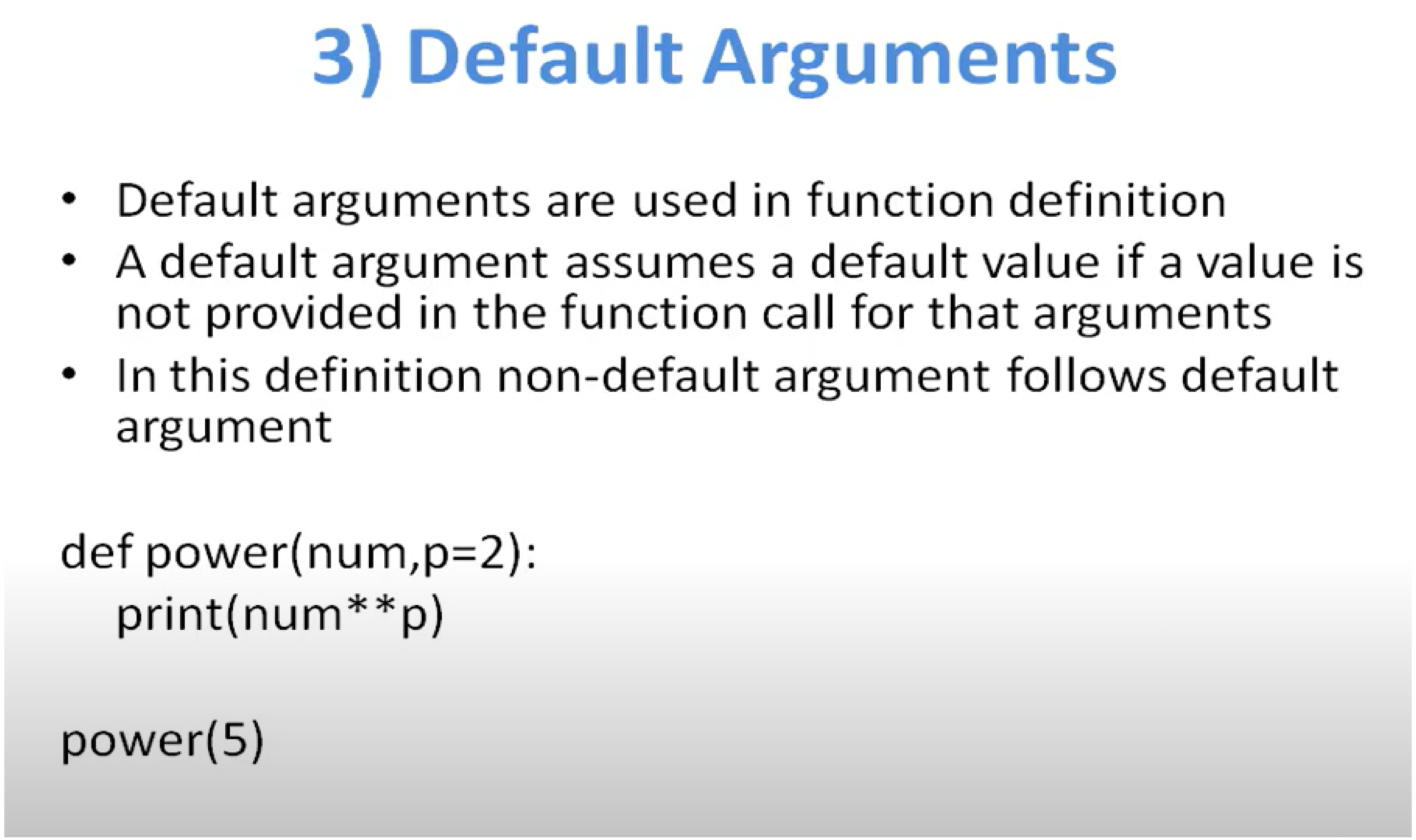
2) Keyword Arguments

- Keyword arguments are used in function call
- The values are assigned based on argument names
- This is beneficial when in function call you change the order of parameters

```
def power(num,p):
    print(num**p)

power(p=3,num=2)
```
OUTPUT: 8

```
def power(num,p):
    print(num**p)

power(3,2)
```
OUTPUT: 9

Functions

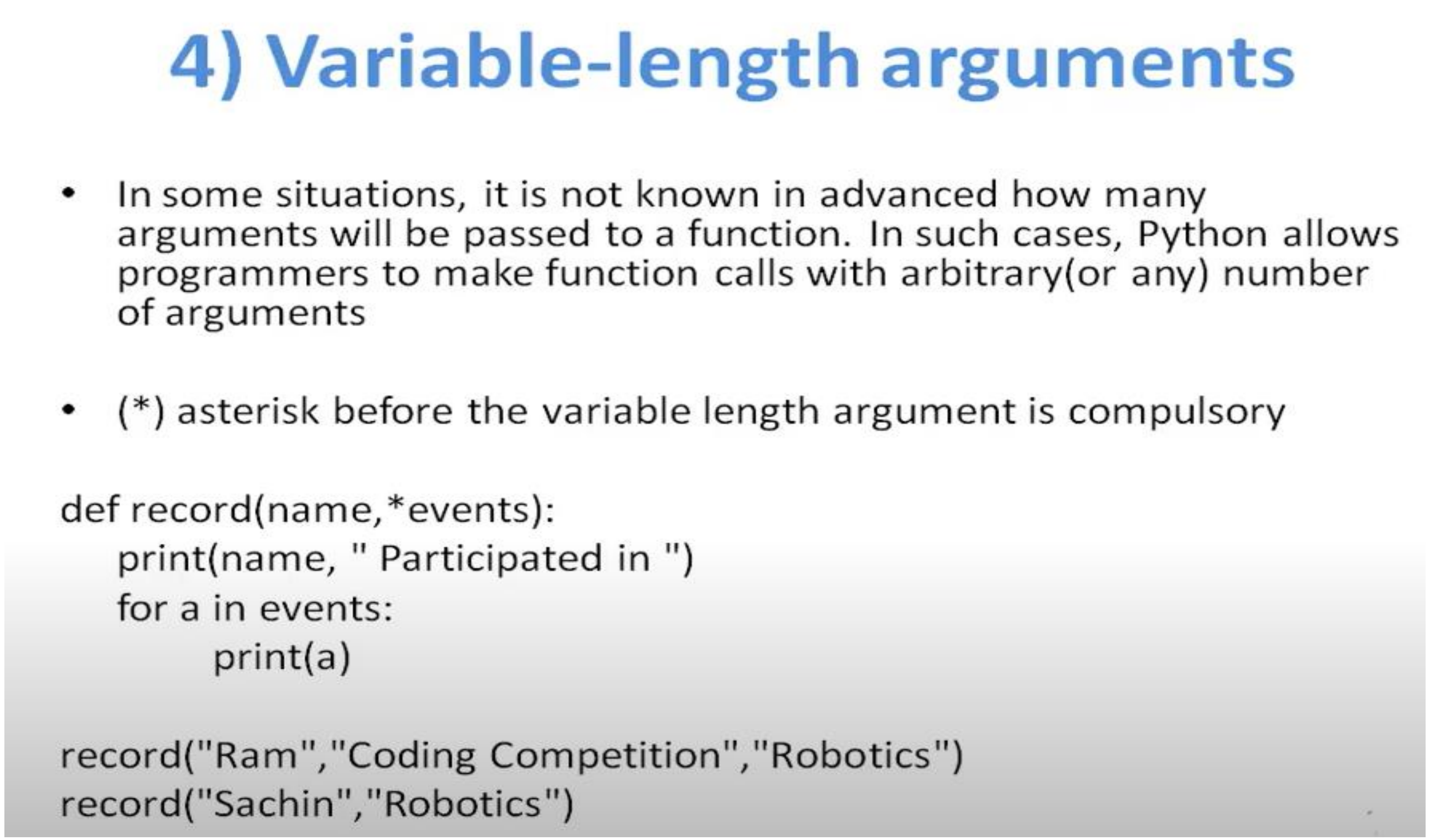


3) Default Arguments

- Default arguments are used in function definition
- A default argument assumes a default value if a value is not provided in the function call for that arguments
- In this definition non-default argument follows default argument

```
def power(num,p=2):
    print(num**p)

power(5)
```

Functions



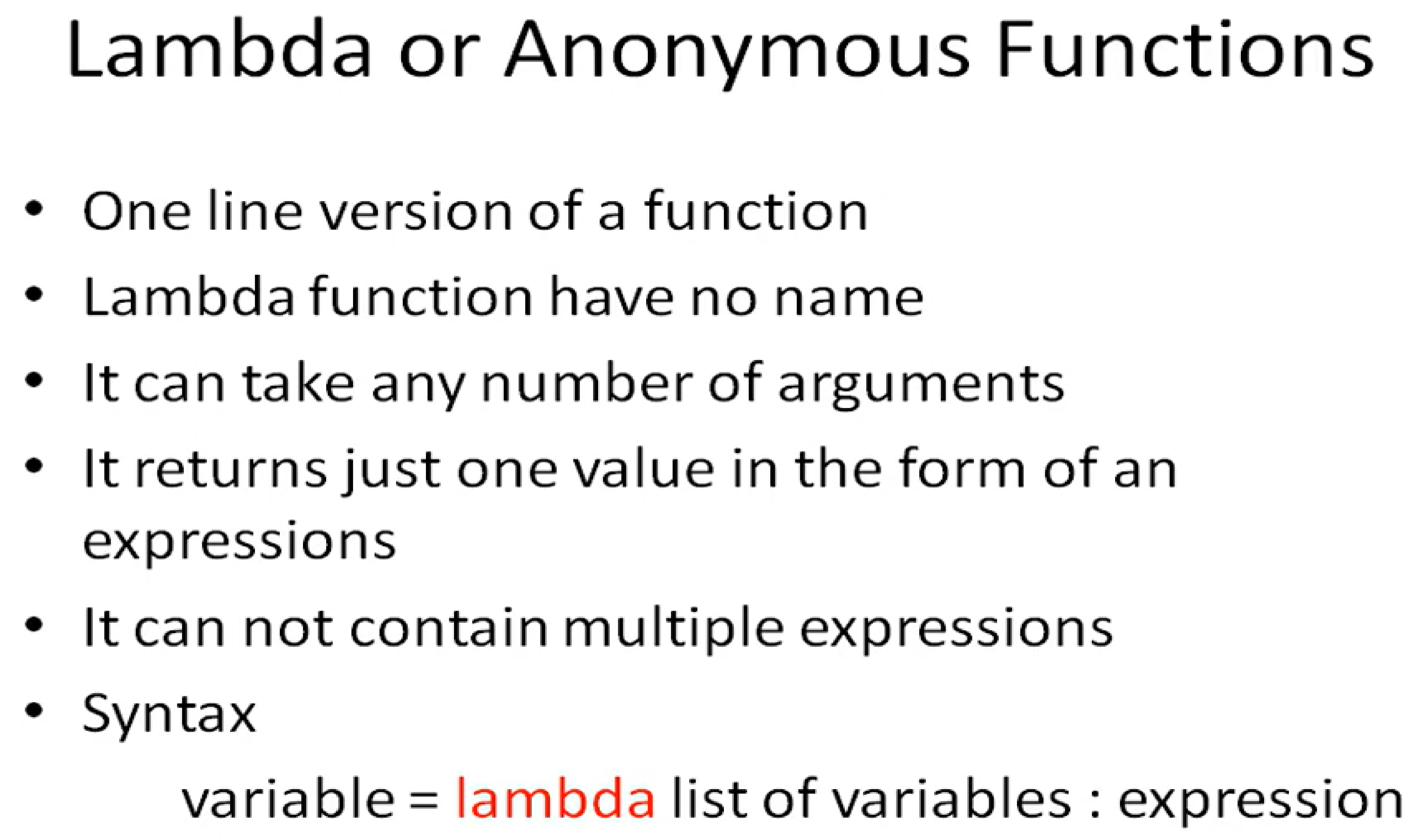
4) Variable-length arguments

- In some situations, it is not known in advanced how many arguments will be passed to a function. In such cases, Python allows programmers to make function calls with arbitrary(or any) number of arguments

- (*) asterisk before the variable length argument is compulsory

```
def record(name,*events):
    print(name, " Participated in ")
    for a in events:
        print(a)

record("Ram","Coding Competition","Robotics")
record("Sachin","Robotics")
```

Functions

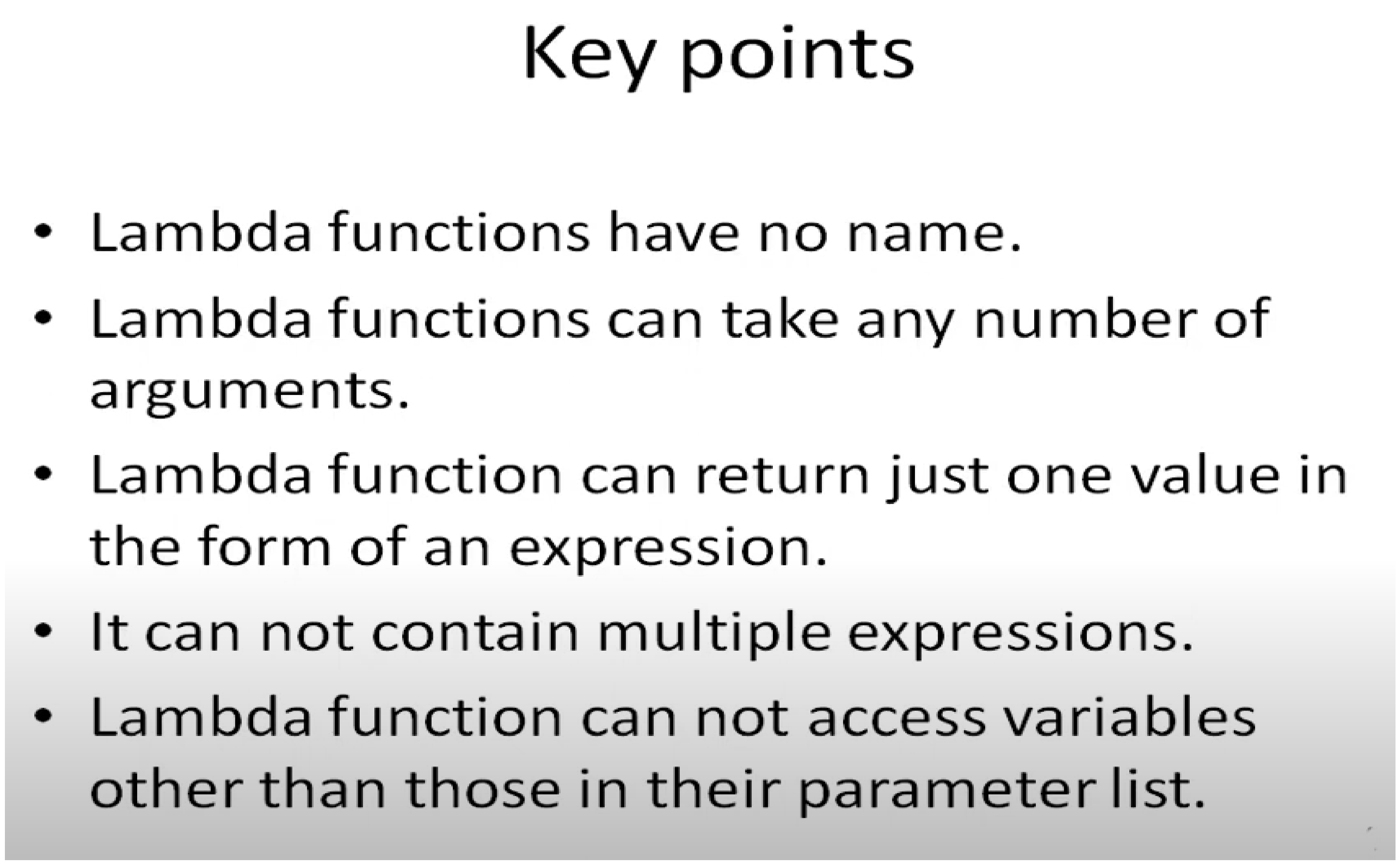


Lambda or Anonymous Functions

- One line version of a function
- Lambda function have no name
- It can take any number of arguments
- It returns just one value in the form of an expressions
- It can not contain multiple expressions
- Syntax

  variable = lambda list of variables : expression

Functions

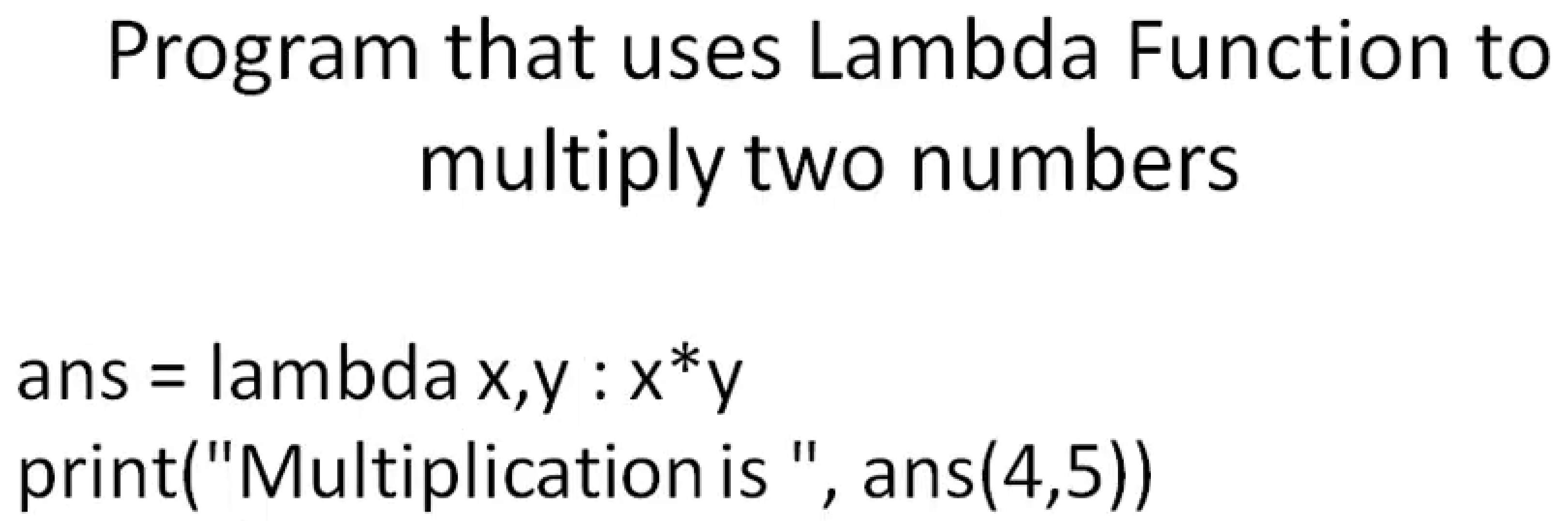


Key points

- Lambda functions have no name.
- Lambda functions can take any number of arguments.
- Lambda function can return just one value in the form of an expression.
- It can not contain multiple expressions.
- Lambda function can not access variables other than those in their parameter list.

Functions

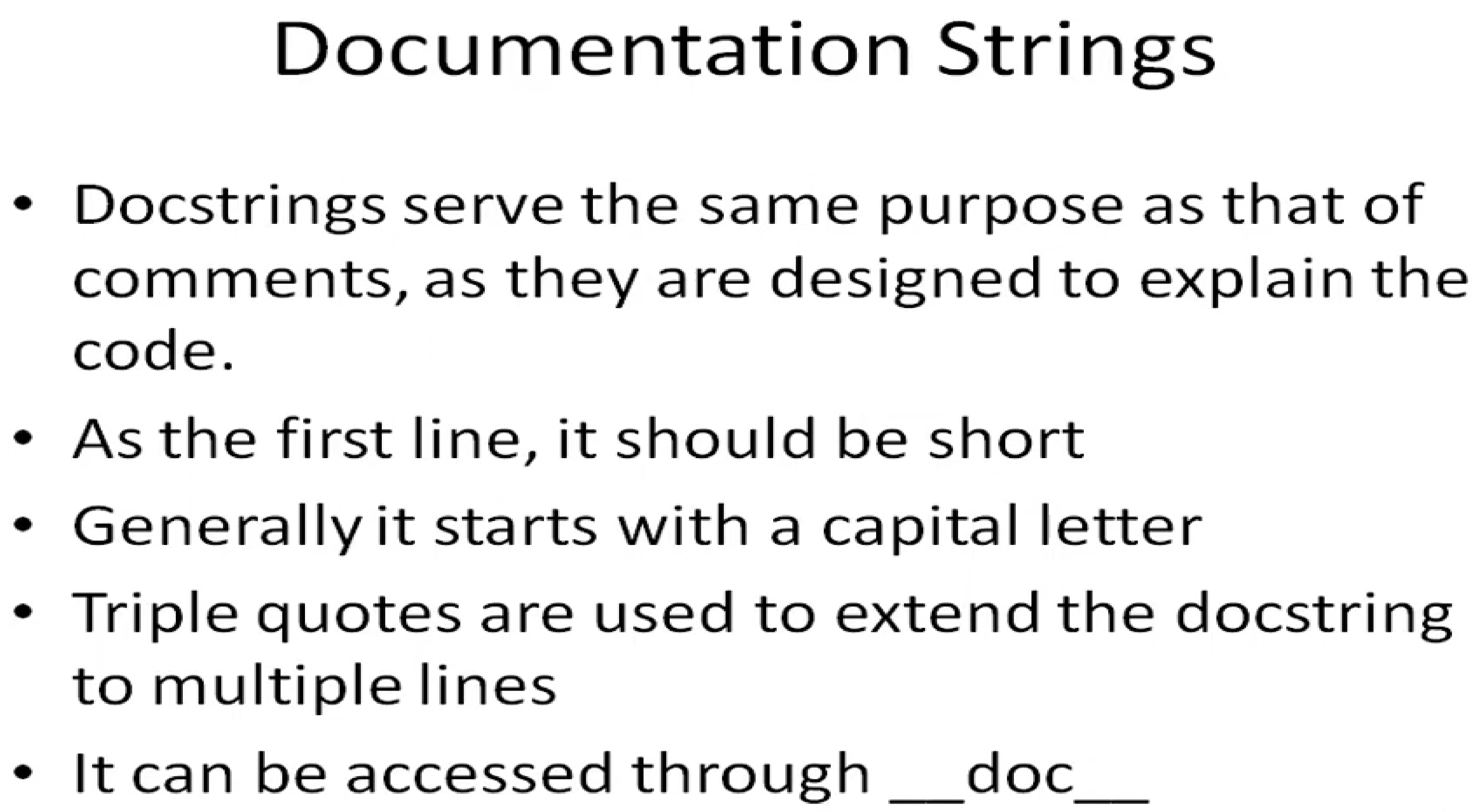
·



Program that uses Lambda Function to multiply two numbers

```
ans = lambda x,y : x*y
print("Multiplication is ", ans(4,5))
```

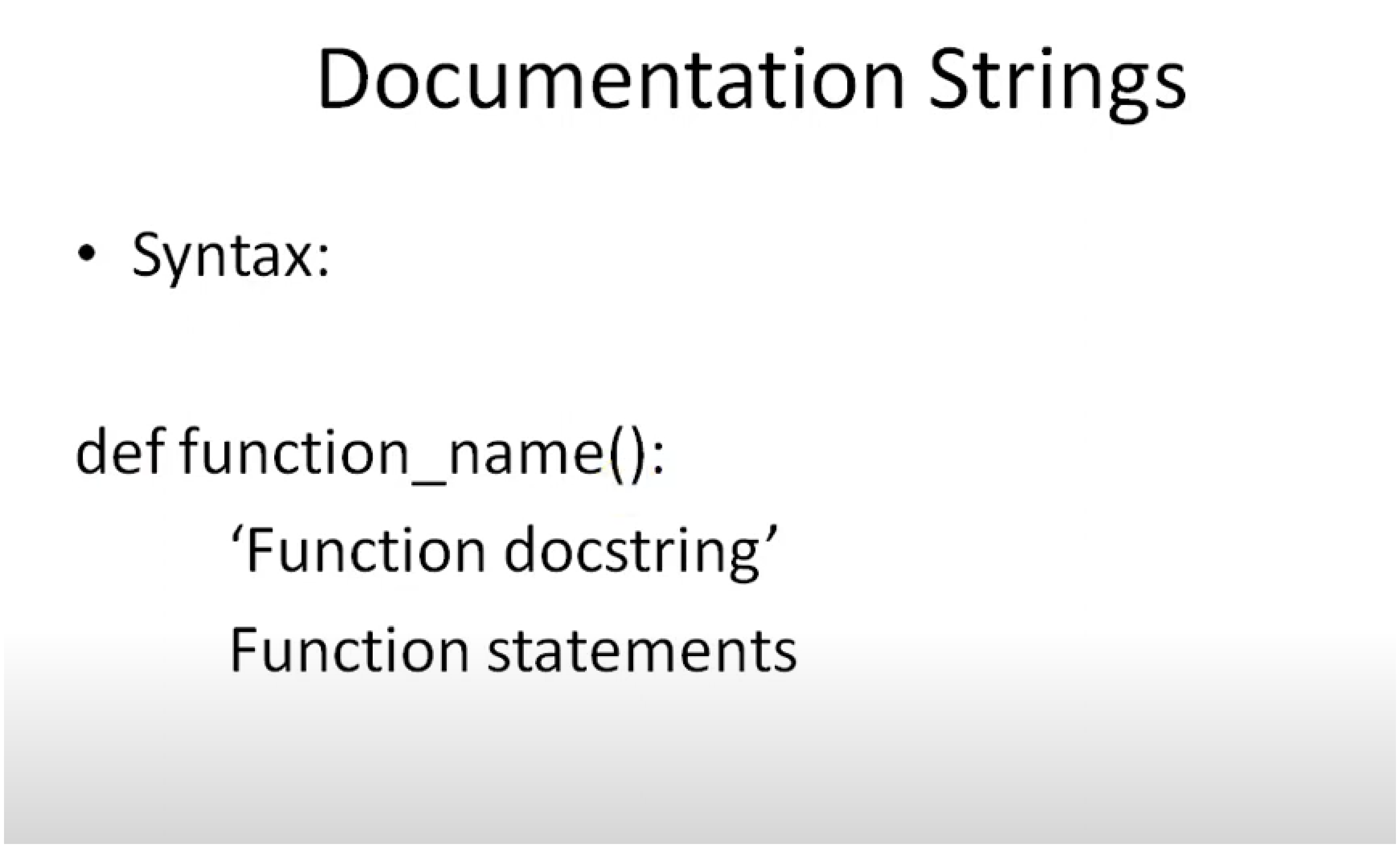
Functions



Documentation Strings

- Docstrings serve the same purpose as that of comments, as they are designed to explain the code.
- As the first line, it should be short
- Generally it starts with a capital letter
- Triple quotes are used to extend the docstring to multiple lines
- It can be accessed through __doc__

Functions

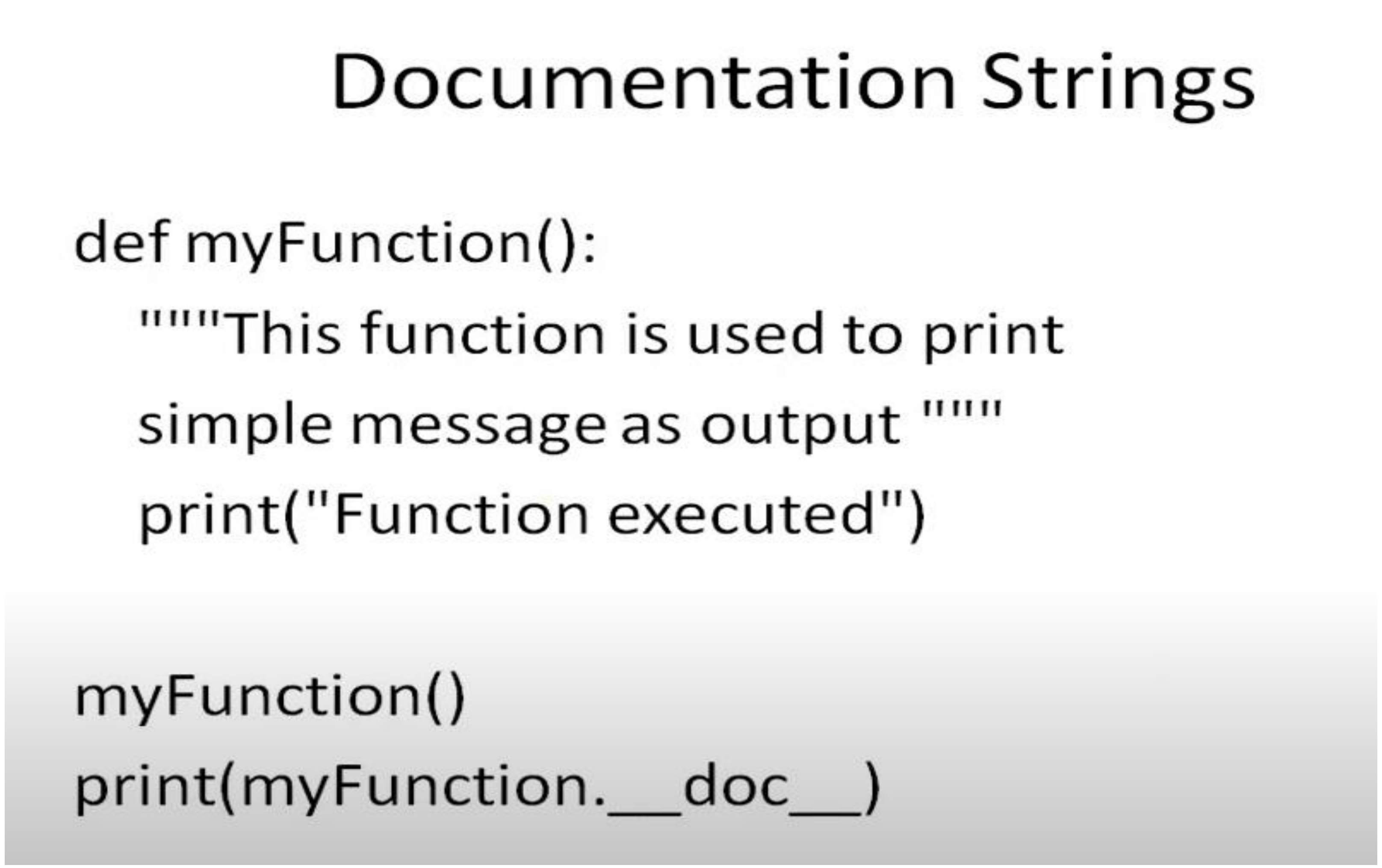


Documentation Strings

- Syntax:

```
def function_name():
        ‘Function docstring’
        Function statements
```

Functions



Documentation Strings

```
def myFunction():
    """This function is used to print
    simple message as output """
    print("Function executed")

myFunction()
print(myFunction.__doc__)
```

NTT DATA