

6.

METHODS AND FUNCTIONS – PART I

- Hands-on Tests!
- Quizzes

✓ Input Parameters

✓ Pass

✓ Return Value

✓ Default Values

✓ Built-In
Functions

✓ Clean
Code

✓ Function and Method's
Definition

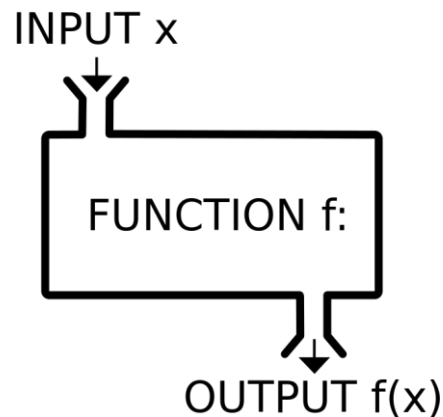
✓ Why Use Functions?

✓ Scopes

✓ Lambda Expressions

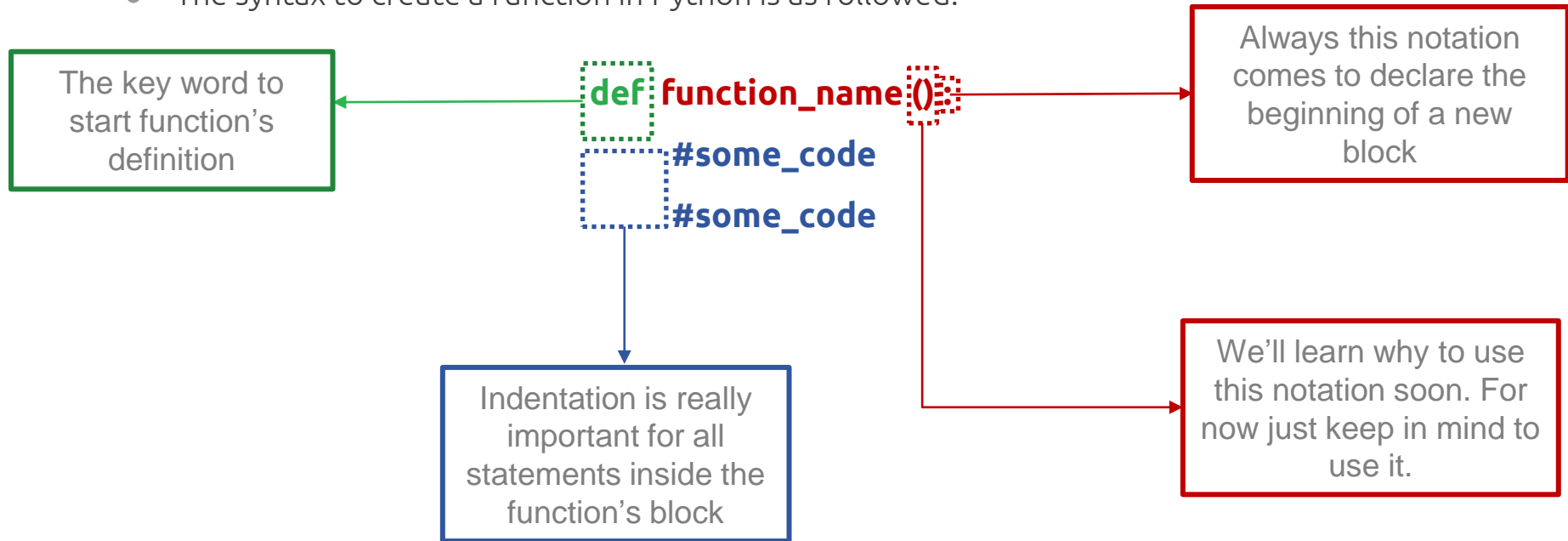
Function Definition

- Functions → is a block of code which
 - ✓ 1. Accomplishes only a certain task → Single performance
 - ✓ 2. For the same inputs, it always *returns* the same output → Injective, One to one
 - ✓ 3. It can be executed only and only when it is *called*.
 - ✓ 4. It can be called multiple times → Reusability



Functions in Python – Definition

- The syntax to create a function in Python is as followed:



Functions in Python – Definition

A simple example...

```
def my_first_function():  
    print("Hey I just created my first function!")
```

Functions in Python – Calling

- The created function will only be executed whenever it's *called*.
- A function can be called multiple times.
- But how to call a function? Take a look at the following example.

The function is
defined here

```
def my_first_function():  
    print("Hey I just created my first function!")
```

```
my_first_function()  
my_first_function()  
my_first_function()
```

The function is
called here 3
times

```
>> Hey I just created my first function!  
>> Hey I just created my first function!  
>> Hey I just created my first function!
```


Functions in Python – Arguments

- The information and variables can be passed into function:

```
def function_name (arguments) :
```

```
#some_code
```

```
#some_code
```




You can pass as many variables as you want into the function

Functions in Python – Arguments

A simple example...

```
def my_first_arguments(i , j , k):  
    print(i)  
    print(j)  
    print(k)  
  
my_age = 22  
my_name = ["Kaveh" , "Masoumi"]  
my_course = "Python"  
  
my_first_arguments(my_age , my_name , my_course)  
  
>> 22  
>> ['Kaveh', 'Masoumi']  
>> Python
```

Functions in Python – Arguments

- Some important notes about arguments (short form, *args*):
 - Always remember:
 - Number of *args* in function calls **=** Number of *args* in definition
 - Different types of data can be passed to each arg.
 - *int*, *float*, *str*, *bool* variables are only passed as *values*.
 - *list*, *dict*, *set*, *tuple* variables are passed as the *variable* itself.
- 

Let's see some examples to see the concept...

Functions in Python – return

- A function can also return a value as a result.

```
def function_name ():
```

```
    #some_code
```

```
    #some_code
```

```
    return some_variable
```

The value of the variable
written in front of the *return*
will be returned



Functions in Python – return

A simple example...

```
def next_year(age):  
    my_age_next_year = age + 1  
    return my_age_next_year  
  
print("next year, i will be", next_year(22))  
  
>> next year, i will be 23
```

Now let's get back to the code...

Thanks!

Got any questions or suggestions?

Here's some contact info:

@KMasoumi

