Introduction to Computer Programming Lecture 4.2:

Function Arguments & Scope

Hemma Philamore

Department of Engineering Mathematics

Passing Arguments

- Arguments allow you to pass parameters to a function (**the input**)
- The part between () **in the function definition** tells you what arguments are expected.
- We can have **multiple** arguments, separated by commas
- The **order** of the arguments is **important**
- You can **define default arguments** with the syntax **MyArgument = Value**, providing these arguments are then optional

Passing Arguments

multiple arguments using commas

```
# Function to print a message
def MyFunction (MyMessage):
    print(MyMessage)


# Function to print a message or several times
def MyFunction2(MyMessage,Copies=1):
    for C in range(Copies):
        print(MyMessage)


# Function with more than two arguments
# arguments with default values have to come last
def MyFunction3(MyMessage,a=3, Copies=1):
    for C in range(Copies):
        print(MyMessage + " " +  str(a))


MyFunction("Hello there!")

MyFunction2("Hello there!",10)
MyFunction2("Hello there!")

MyFunction3("We will need",3,2)
MyFunction3("We will need",2)
MyFunction3("We will need",Copies=2)
```

Named argument: default value

default values have to come last

can use arguments as variables later

change value by using named argument

More Complex Arguments

```
# sums all items in a list
def SumItems(ListArg):
    Sum = 0
    for Item in ListArg:
        Sum = Sum + Item
    print(Sum)

# list can also be of mixed types
def ShowInfo(InfoList):
    print("Name: " + InfoList[0])
    print("Age: " + str((InfoList[1])))
```

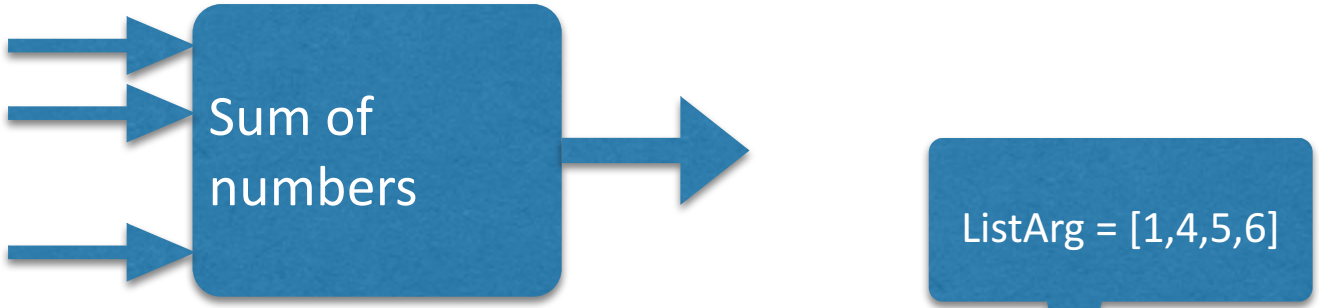
goes through the list

```
SumItems([1,2,3,4,5])
ShowInfo(["John Doe",23])
```

call using list

More Complex Arguments

Sometimes we don't know the exact number of inputs




```
def SumItems(ListArg):
```

- Solution (A): Input is **one** list

- Solution (B): use the form ***name**
  - function receives a **tuple** containing all parameters

```
def SumItems2(*Args):
```

immutable

More Complex Arguments

```
# sums all items passed as ar
def SumItems2(*Args):
    Sum = 0
    for Item in Args:
        Sum = Sum + Item
    print(Sum)
```

Python puts arguments in a tuple…

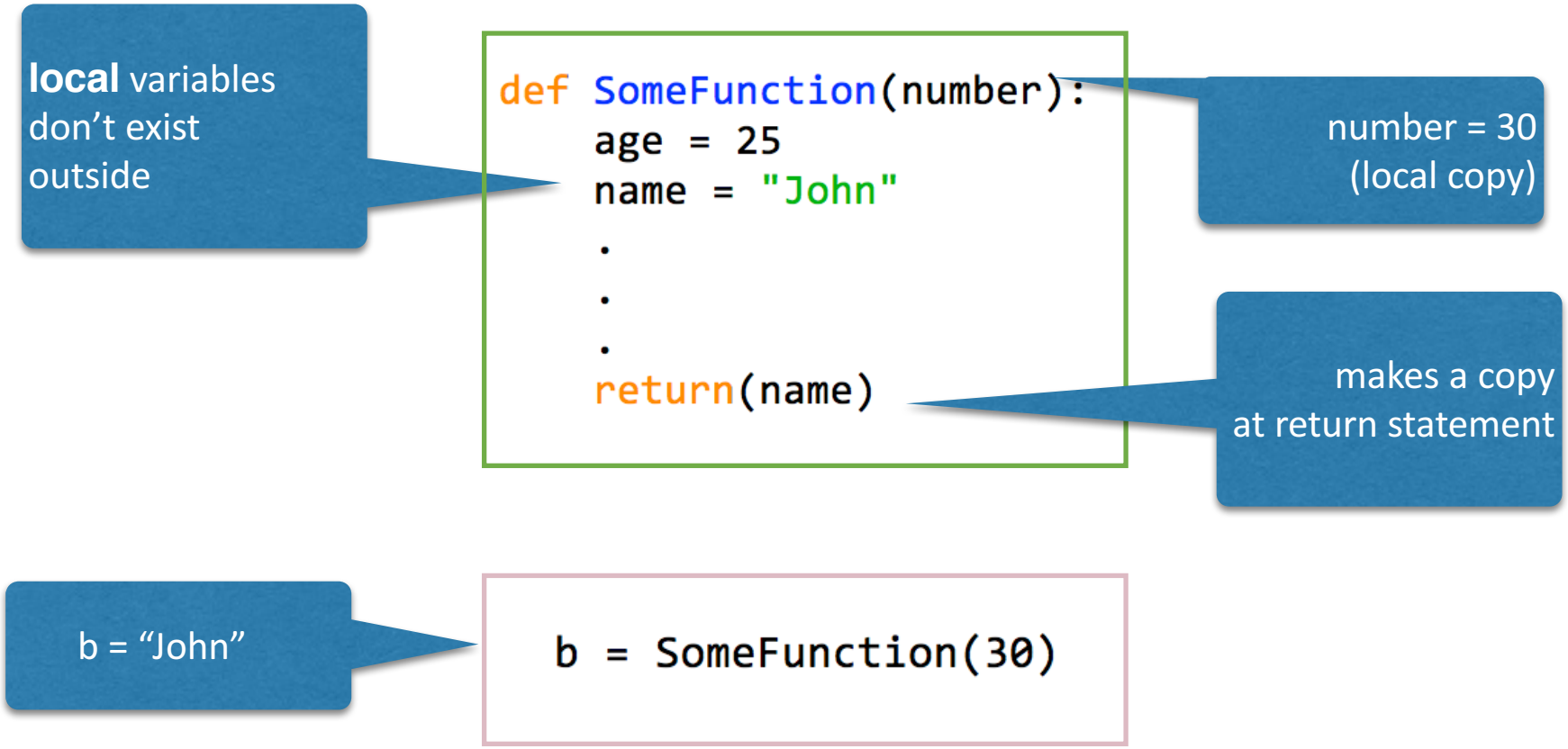
…goes through the tuple

```
SumItems2(1,2,3,4)
```

Call with multiple items

Variable Scope

Variables **defined in** the function are **only local**!



Variable Scope

```
def PrintString():
  MyText = "This is local"
  print(MyText)


def PrintString2():
  global MyText
  print(MyText)


MyText = "GLOBAL"

PrintString()
PrintString2()
```