# Functions

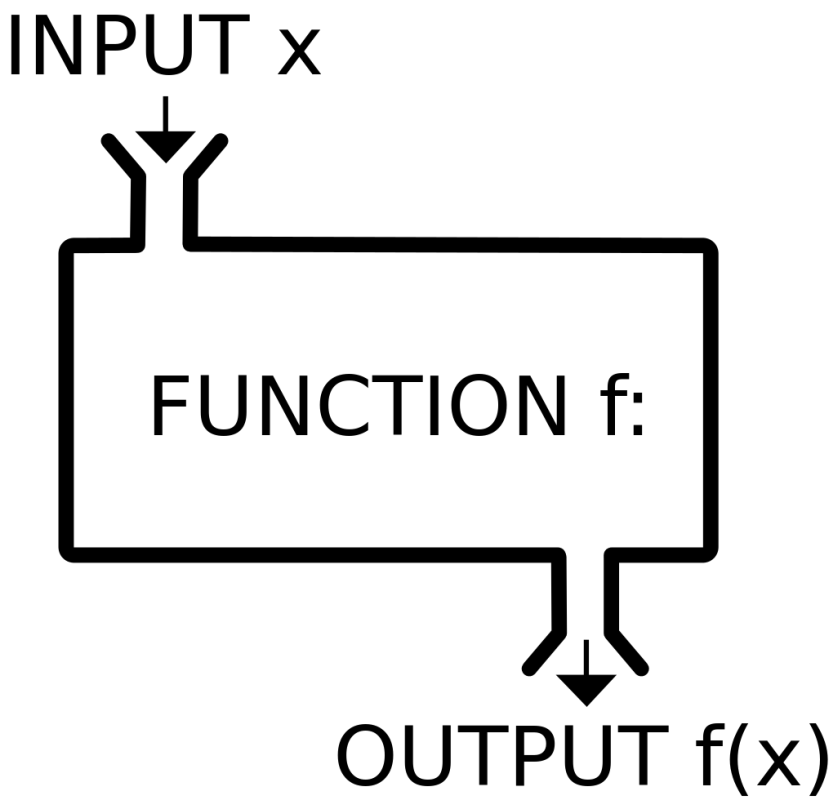
# (catchup class)

Example of code

**with duplicated code**

To avoid code duplication, we can use

**FUNCTIONS (like in math)**




Example:

```
Y = x*x +3
```

**inputs** are parameters

**output** are returns

4

```
def squarePlus3(x):
    return (x * x) + 3
```

19

# How to DEFINE a function ?

Parameters are defined inside the ‘(‘ and ‘)‘

def
to define a function

A function have a name

Don’t forget the :

```
def squarePlus3(number) :
    return number * number
```

Use the return key word to **return** the result of your function

Same example of code

**with a function this time !**

# How to CALL a function ?

1- Define the parameters for your function call

```
myResult = square(5)
```

2- Get the result of the function call

# Let’s see step by step !

## Here I will call a function many times

## Here I define a function

# Let’s see step by step !

## Here I will call a function many times

## Here I define a function

```
def addNumbers(number1 , number2) :
    return number1 + number2
```

# Let’s see step by step !

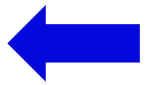
## Here I will call a function many times

```
a = 15
b = 23

c = addNumbers(a, b)

d = addNumbers(10, 5)
```

First call to my function



Second call to my function

## Here I define a function

```
def addNumbers(number1 , number2) :
        return number1 + number2
```

# Let’s see step by step !

## Here I will call a function many times

```
a = 15
b = 23

c = addNumbers(a, b)

d = addNumbers(10, 5)
```

15 23

15, 23

## Here I define a function

```
def addNumbers(number1 , number2) :
    return number1 + number2
```

1- Call function addNumbers with parameters 15 and 23

# Let’s see step by step !

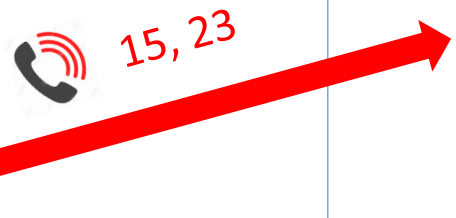
## Here I will call a function many times

```
a = 15
b = 23

c = addNumbers(a, b)

d = addNumbers(10, 5)
```

15 23




## Here I define a function

15 23

```
def addNumbers(number1 , number2) :
    return number1 + number2
```

2- Execute function and compute the return value

# Let’s see step by step !

## Here I will call a function many times

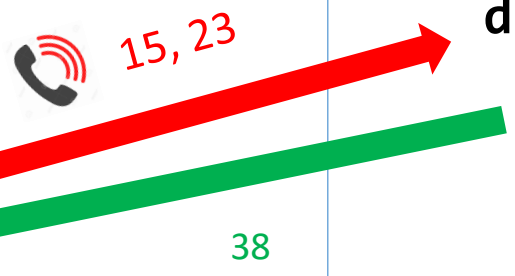
```
a = 15
b = 23

c = addNumbers(a, b)

d = addNumbers(10, 5)
```

38

15 23




## Here I define a function

15 23

```
def addNumbers(number1 , number2) :
    return number1 + number2
```

3- Exit function with the return value 38

# Let’s see step by step !

## Here I will call a function many times

```
a = 15
b = 23

c = addNumbers(a, b)

d = addNumbers(10, 5)
```


## Here I define a function

```
def addNumbers(number1 , number2) :
    return number1 + number2
```

3- Call function addNumbers with parameters 10 and 5

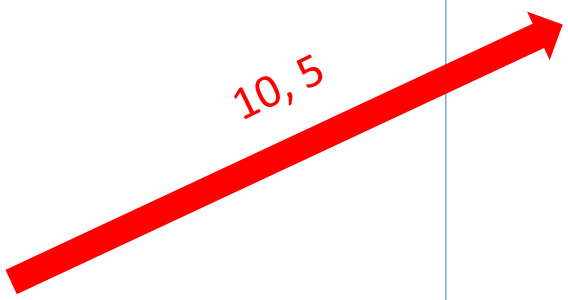
# Let’s see step by step !

## Here I will call a function many times

```
a = 15
b = 23

c = addNumbers(a, b)

d = addNumbers(10, 5)
```




## Here I define a function

10 5

```
def addNumbers(number1 , number2) :
    return number1 + number2
```

2- Execute function and compute the return value

# Let’s see step by step !

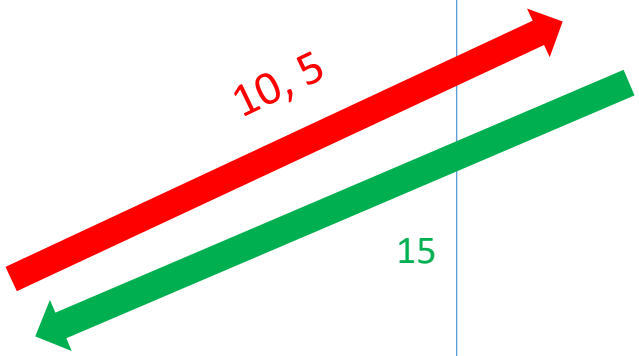
## Here I will call a function many times

```
a = 15
b = 23

c = addNumbers(a, b)

d = addNumbers(10, 5)
```

15




## Here I define a function

10 5

```
def addNumbers(number1 , number2) :
    return number1 + number2
```

3- Exit function with the return value 15

# Make the difference !

Function definition

```
def squarePlus3(x):
    return (x * x) + 3
```

Function call

```
print(squarePlus3(10))
print(squarePlus3(15))
```

Call function with parameter 10

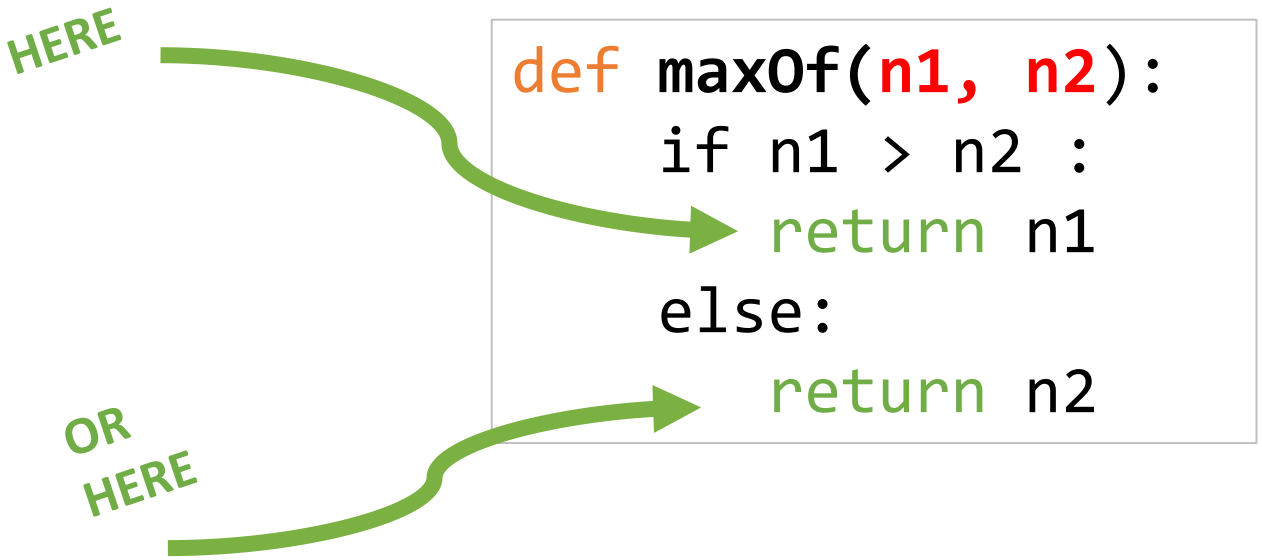
Call function with parameter 15

Let’s demo

**Using a debugger**

# You exit a function

## Using the return key word



# Can we have an instruction after return ?

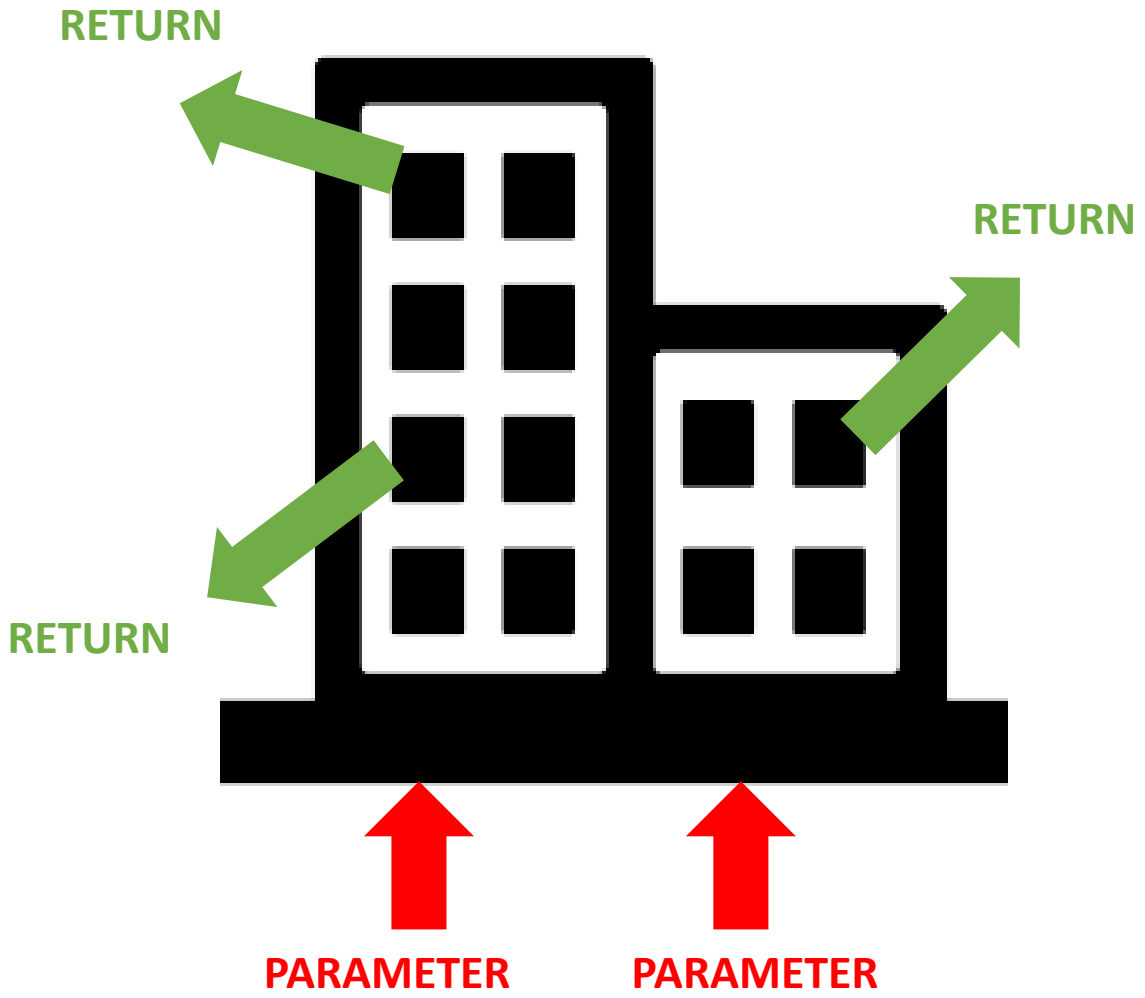

```
def addNumbers(number1 , number2) :
    return number1 + number2
    number1 = 4
```


# But a function can have <u>many exits</u>!!

```
def maxOf(n1, n2):
    if n1 == n2 :
        return n1

    elif if n1 > n2:
        return n1

    else:
        return n2
```



# Each exit points shall return the same type of value

```
def compute(n1, n2):
    if n1 == n2 :
        return n1

    else:
        return n1 + 1
```

INT

INT

```
def compute(n1, n2):
    if n1 == n2 :
        return n1

    else:
        return n1 > 1
```

INT

BOOL

# You know !!

1 - Difference btw a function definition and function call

2 - A function can be called many times with different values

3 - A function must return something (*this is the "y" in math*)

4 - A function can have many exist (*many "return"*)

- If many returns : each return shall be of the SAME type


# What this code will print?

```
def computeAbsolute(number) :
    if number<0:
        return -1 * number
    else
        return number

print(computeAbsolute(-4))
```

| A | B | C | D |
|---|---|---|---|
| -4 | 4 | computeAbsolute(-4) | error |


# What this code will print?

```
def computeAbsolute(number) :
    if number<0:
        return -1 * number
    else
        return number

print(computeAbsolute(-4))
```

A

-4

B

4

C

computeAbsolute(-4)

D

error


# Write a function to compute the sum of 2 numbers

| | |
|---|---|
| **Name** | computeSum |
| **Parameters** | number1, number2 |
| **Return** | the sum of the 2 numbers |

```
result = computeSum(4,5)  # result should be 9

result = computeSum(2,4)  # result should be 6
```


# Write a function to compute the minimum of 2 numbers

| **Name** | computeMin |
|---|---|
| **Parameters** | number1, number2 |
| **Return** | the min of the 2 numbers |

```
result = computeMin(4,5)  # result should be 4

result = computeMin(3,7)  # result should be 3
```

# What will be the return type of this function ?

INT

```
def myFunction(number):
    return “hello” + str(number)
```

A

string

B

boolean

C

integer

D

This code can produce errors

# What will be the return type of this function ?

INT

```
def myFunction(number):
    return “hello” + str(number)
```

A

string

B

boolean

C

integer

D

This code can produce errors

# What will be the return type of this function ?


```
def myFunction(number1, number2):
    if number1> 10
        return number1
    return number2
```

| A | B | C | D |
|---|---|---|---|
| string | boolean | integer | This code can produce errors |

# What will be the return type of this function ?

INT INT

```
def myFunction(number1, number2):
    if number1> 10
        return number1
    return number2
```

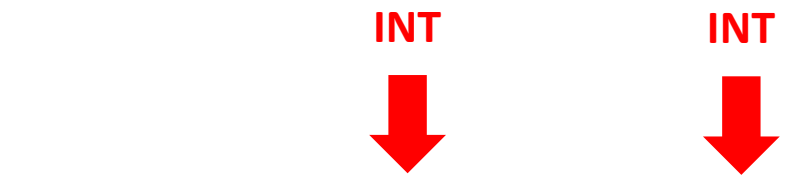
A string

B boolean

C integer

D This code can produce errors

# What will be the return type of this function ?

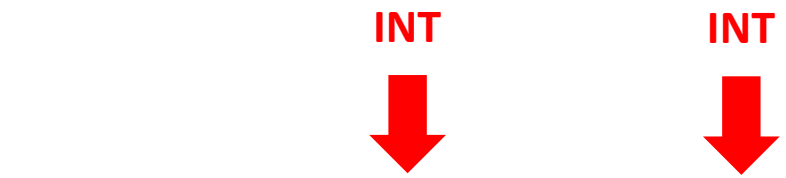



```
def myFunction(number1, number2):
    if number1 > 10
        return number1 == 20
    return number2

print(myFunction(11) + 2)
```

| A | B | C | D |
| --- | --- | --- | --- |
| string | boolean | integer | This code can produce errors |

# What will be the return type of this function ?




```
def myFunction(number1, number2):
    if number1 > 10
        return number1 == 20
    return number2

print(myFunction(11) + 2)
```

A

string

B

boolean

C

integer

D

This code can produce errors

# What will be the return type of this function ?

INT

```
def myFunction(number1):
    if number1 > 10 :
        return number1


result = myFunction(4) + 4
```

| A | B | C | D |
| --- | --- | --- | --- |
| string | boolean | integer | This code can produce errors |

# What will be the return type of this function ?

INT

```
def myFunction(number1):
    if number1 > 10 :
        return number1


result = myFunction(4) + 4
```

A

string

B

boolean

C

integer

D

This code can produce errors

# What this code will print on console ?

INT

```
def myFunction(number1):
      return 99
      return number1 + 1

print(myFunction(9))
```

A

9

B

99

C

10

D

This code can produce errors

# What this code will print on console ?

INT

```
def myFunction(number1):
      return 99
      return number1 + 1

print(myFunction(9))
```

A

B

C

D

9

99

10

This code can produce errors

# What this code will print on console ?

```
def myFunction(text):
      nbChars = len(text)
      if nbChars>10:
          return nbChars
      else :
          return nbChars + 1

result = myFunction(“rady”)
print(result)
```

| A | B | C | D |
|---|---|---|---|
| rady | 4 | 5 | This code can produce errors |

# What this code will print on console ?

```
def myFunction(text):
      nbChars = len(text)
      if nbChars>10:
          return nbChars
      else :
          return nbChars + 1

result = myFunction("rady")
print(result)
```

| A | B | C | D |
|---|---|---|---|
| rady | 4 | 5 | This code can produce errors |

# What this code will print on console ?

```
result = myFunction(“rady”)
print(result)


def myFunction(text):
      nbChars = len(text)
      if nbChars>10:
          return nbChars
      else :
          return nbChars + 1
```

A

rady

B

4

C

5

D

This code can produce errors

# What this code will print on console ?

```
result = myFunction("rady")
print(result)


def myFunction(text):
      nbChars = len(text)
      if nbChars>10:
          return nbChars
      else :
          return nbChars + 1
```

A

rady

B

4

C

5

D

This code can produce errors

# What this code will print on console ?

```
def moreOne(number):
        return number + 1

def multiplyBy2(number):
        return number * 2


result = moreOne(multiplyBy2(moreOne(moreOne(2))))
print(result)
```

| A | B | C | D |
|---|---|---|---|
| 2 | 8 | 9 | This code can produce errors |

# What this code will print on console ?

```
def moreOne(number):
        return number +1

def multiplyBy2(number):
        return number * 2


result = moreOne(multiplyBy2(moreOne(moreOne(2))))
print(result)
```

| A | B | C | D |
|---|---|---|---|
| 2 | 8 | 9 | This code can produce errors |