





POLITÉCNICA Universidad Politécnica de Madrid

Practical IoT with Raspberry Pi

Module 2. Introduction to Python Language

Practical IoT with Raspberry Pi



POLITÉCNICA Universidad Politécnica de Madrid

Session 2.4 Functions and Modules

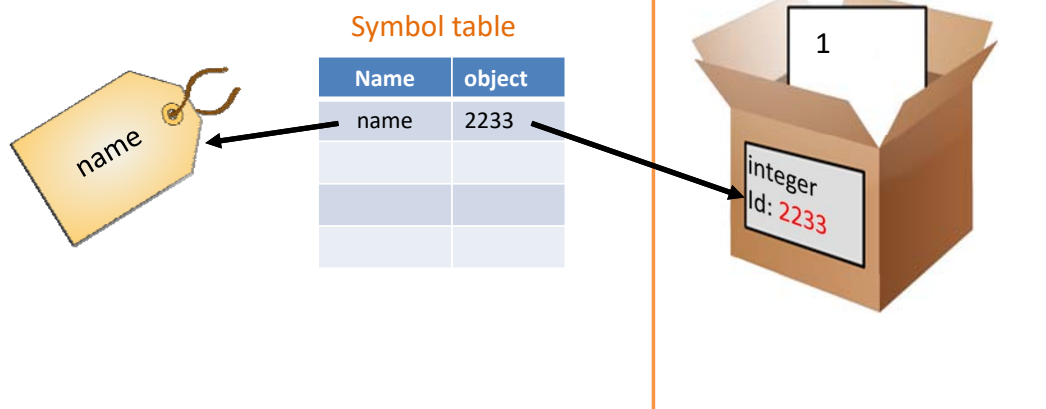
- **Functions:**
 - Symbol tables.
 - Local and global variables
 - Parameters
- **Modules**
- **Packages**

Module 2: Introduction to Python Language Speaker: M. Hernando

Practical IoT with Raspberry Pi



2.4.1 Names and values



Module 2: Introduction to Python Language

Speaker: M. Hernando

Practical IoT with Raspberry Pi

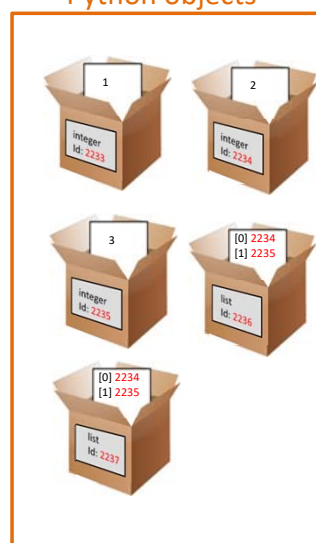


2.4.1 Names and values

Python symbols

Name	object

Python objects



Module 2: Introduction to Python Language

Speaker: M. Hernando

Practical IoT with Raspberry Pi

2.4.2 Functions

docstring

```
def fib(n):
    """Print a Fibonacci series up to n"""
    a, b = 0, 1
    while a < n:
        print(a, end=' ')
        a, b = b, a+b
    print()
```

Python symbols

Name	object
fib	123

Python objects

Python is a programming language that lets you work quickly and integrate systems more effectively. >>> [Learn More](#)

Module 2: Introduction to Python Language

Speaker: M. Hernando

Practical IoT with Raspberry Pi

2.4.2 Functions

```
>>> def fib(n): #write Fibonacci series up to n
    a, b = 0, 1
    while a < n:
        print(a, end=' ')
        a, b = b, a+b
    print()
>>> f = fib
>>> fib(100)
0 1 1 2 3 5 8 13 21 34 55 89
>>> f(100)
0 1 1 2 3 5 8 13 21 34 55 89
```

global symbols

Name	object
n	200
a	300
b	400

fib symbols

Name	object
n	200
a	300
b	400

f symbols

Name	object
n	200
a	300
b	400

Python objects

Module 2: Introduction to Python Language

Speaker: M. Hernando

2.4.3 Function parameters

2.4.4 The return statement

Function without return

```
>>> def no_return():
        print('No return... ')
>>> no_return()
No return...
>>> print(no_return())
No return...
None
```

Return without arguments

```
>>> def multiple_of_7(n):
        if n % 7:
            return
        print('Multiple of 7')
>>> multiple_of_7(8)
>>> multiple_of_7(14)
Multiple of 7
>>> print(multiple_of_7(8))
None
```

2.4.4 The return statement

```
>>> def fib2(n):
    result = []
    a, b = 0, 1
    while a < n:
        result.append(a)
        a, b = b, a+b
    return result

>>> series = fib2(100)
>>> series
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

Instead of printing, we create a list with the Fibonacci sequence

At the end of the algorithm we return the list

2.4.4 The return statement

```
>>> def fib3(n):
    result = []
    a, b = 0, 1
    while a < n:
        result.append(a)
        a, b = b, a+b
    return result, len(result)

>>> info = fib3(100)
>>> info
([0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89], 12)
>>> serie, number = fib3(100)
>>> serie
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
>>> number
12
```

A comma-separated expression is a tuple

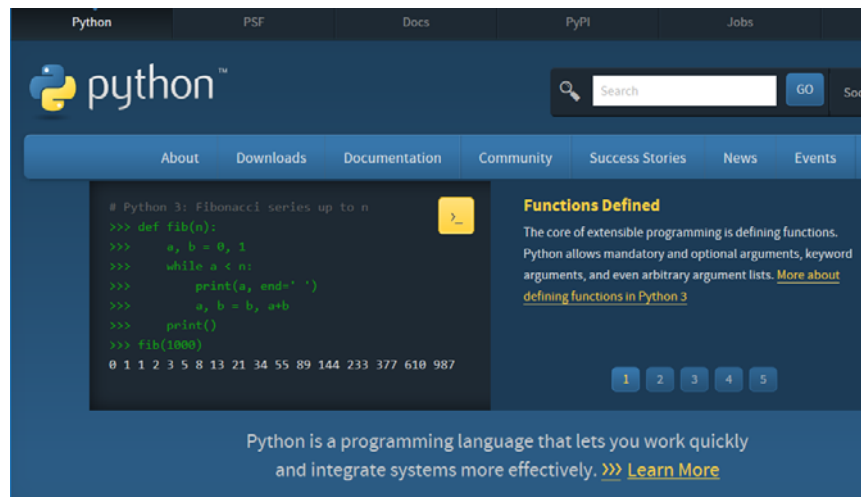
Info is a tuple with a list and an integer

Each element of the returned tuple is assigned to each element of the receiving tuple

Practical IoT with Raspberry Pi



2.4.4 Functions



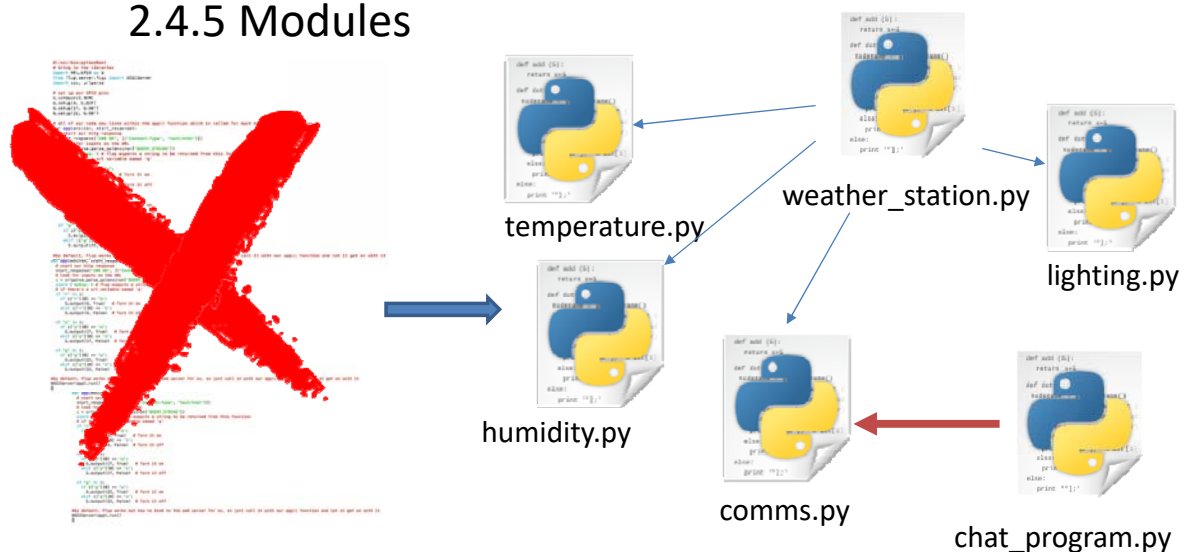
Module 2: Introduction to Python Language

Speaker: M. Hernando

Practical IoT with Raspberry Pi



2.4.5 Modules



Module 2: Introduction to Python Language

Speaker: M. Hernando

Practical IoT with Raspberry Pi



2.4.5 Modules

The module name is minmax

minmax.py

```
def min(a, b):
    if a < b:
        return a
    else:
        return b

def max(a, b):
    if a > b:
        return a
    else:
        return b
```

Module 2: Introduction to Python Language

Speaker: M. Hernando

Practical IoT with Raspberry Pi



2.4.5 Modules

```
>>> from minmax import min, max
>>> import sys
>>> sys.path
```

1. Directory of the **current script** or the current directory when no file specified
2. Content of **PYTHONPATH** (a list of directory names)
3. The **installation-dependent** default

Module 2: Introduction to Python Language

Speaker: M. Hernando

Practical IoT with Raspberry Pi



2.4.5 Modules

```
pi@raspberrypi:~/Desktop/test $ python minmax.py
```

↓

```
__name__ == '__main__'
```

```
>>> from minmax import *
```

↓

```
__name__ == 'minmax'
```

```
if __name__ == '__main__':
    print("Max between 3 and 10 is", max(3, 10))
    print("Max between 3 and -10 is", max(3, -10))
    print("Min between 3 and 10 is", min(3, 10))
    print("Min between 3 and -10 is", min(3, -10))
```

Practical IoT with Raspberry Pi



2.4.5 Modules

minmax.py

```
def min(a, b):
    if a < b:
        return a
    else:
        return b

def max(a, b):
    if a > b:
        return a
    else:
        return b

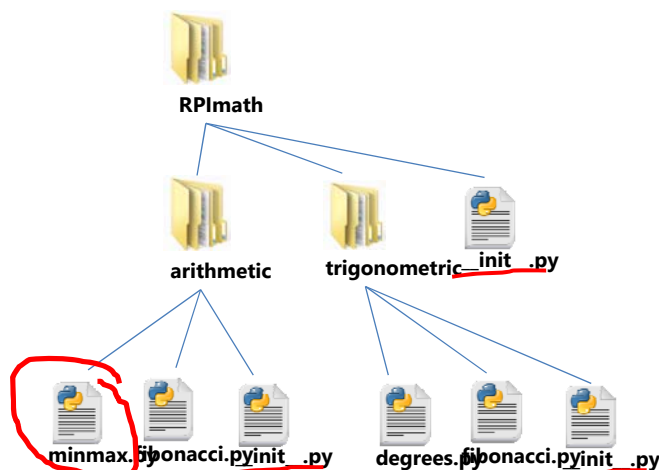
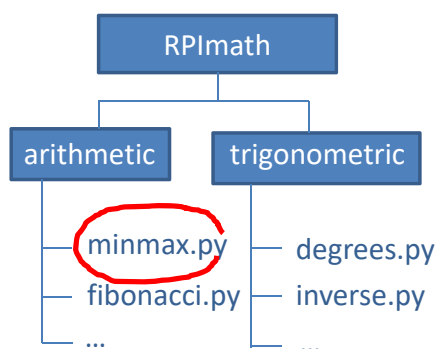
epsilon = 0.0001
print("Module initiated")

if __name__ == '__main__':
    print("Max between 3 and 10 is", max(3, 10))
    print("Max between 3 and -10 is", max(3, -10))
    print("Min between 3 and 10 is", min(3, 10))
    print("Min between 3 and -10 is", min(3, -10))
```


Practical IoT with Raspberry Pi



2.4.6 Packages



Module 2: Introduction to Python Language

Speaker: M. Hernando

Practical IoT with Raspberry Pi

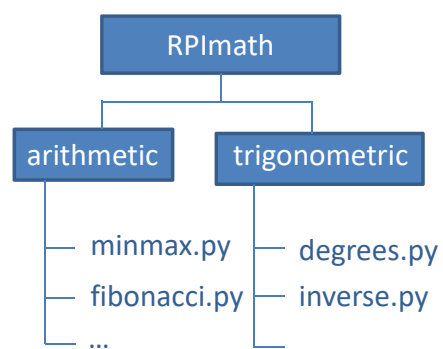


2.4.6 Packages

```
>>> import RPImath.arithmetic.minmax
>>> RPImath.arithmetic.minmax.min(2,3)
```

```
>>> from RPImath.arithmetic import minmax
>>> minmax.min(2,3)
```

```
>>> from RPImath.arithmetic.minmax import min, max
>>> min(2,3)
```



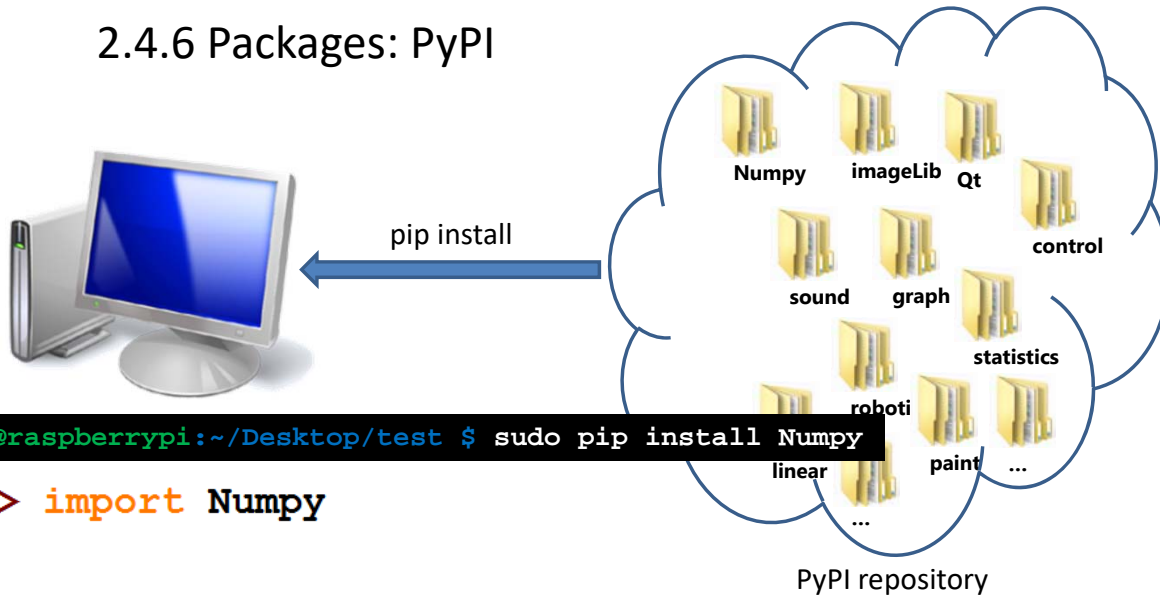
Module 2: Introduction to Python Language

Speaker: M. Hernando

Practical IoT with Raspberry Pi

POLITÉCNICA Universidad Politécnica de Madrid

2.4.6 Packages: PyPI



```
pi@raspberrypi:~/Desktop/test $ sudo pip install Numpy
>>> import Numpy
```

Module 2: Introduction to Python Language

Speaker: M. Hernando

Practical IoT with Raspberry Pi

POLITÉCNICA Universidad Politécnica de Madrid

Summary

- ✓ Function definition
- ✓ Symbol table concept
- ✓ Function parameters and return statement
- ✓ Concept of modules
- ✓ Packages and pip

Module 2: Introduction to Python Language

Speaker: M. Hernando