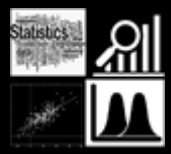
# An Introduction to Python

Data Science 2 / Data & AI 3

KdG Karel de Grote Hogeschool

# Agenda




1. Introduction
   - Who uses Python, what & Why
2. Variables and data types
3. Program flow, functions and modules
4. Notebook time!

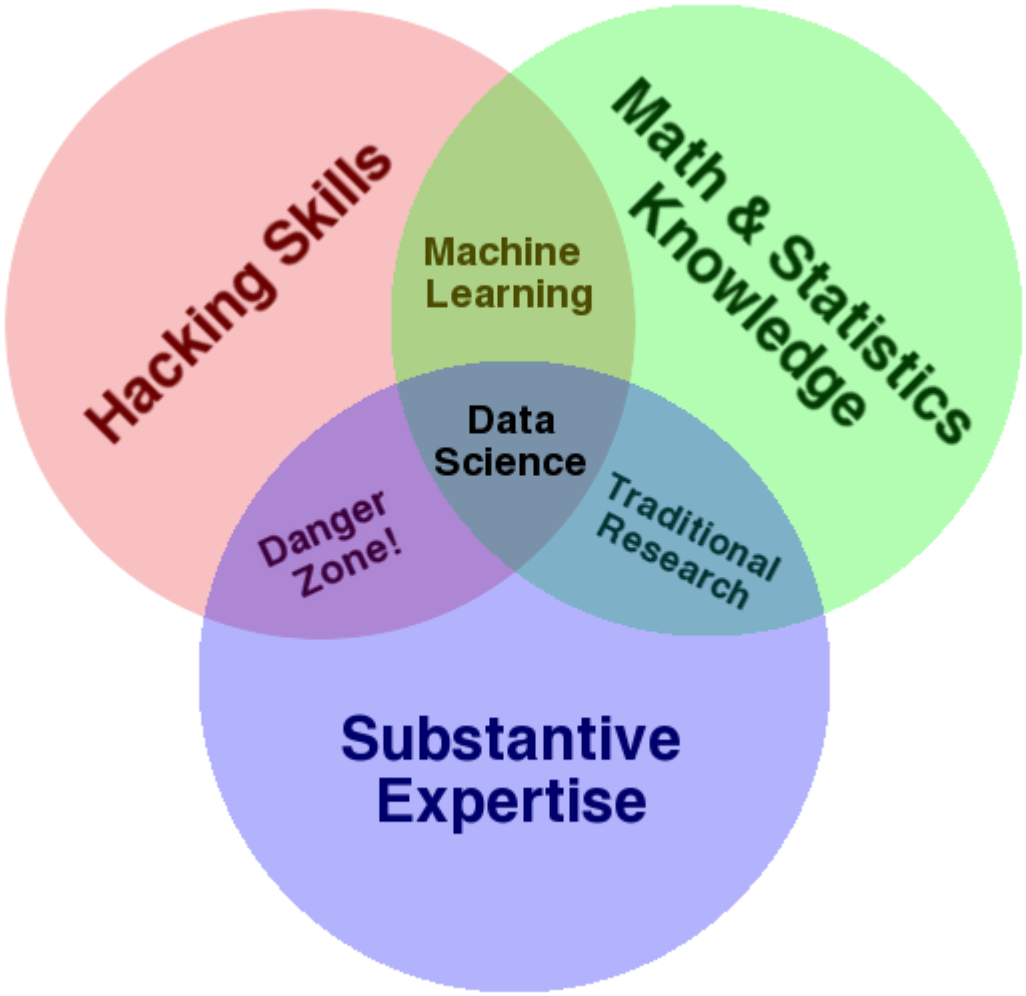

# Introduction

1

# Introduction

- This book is about learning data science in python

- What is data science?

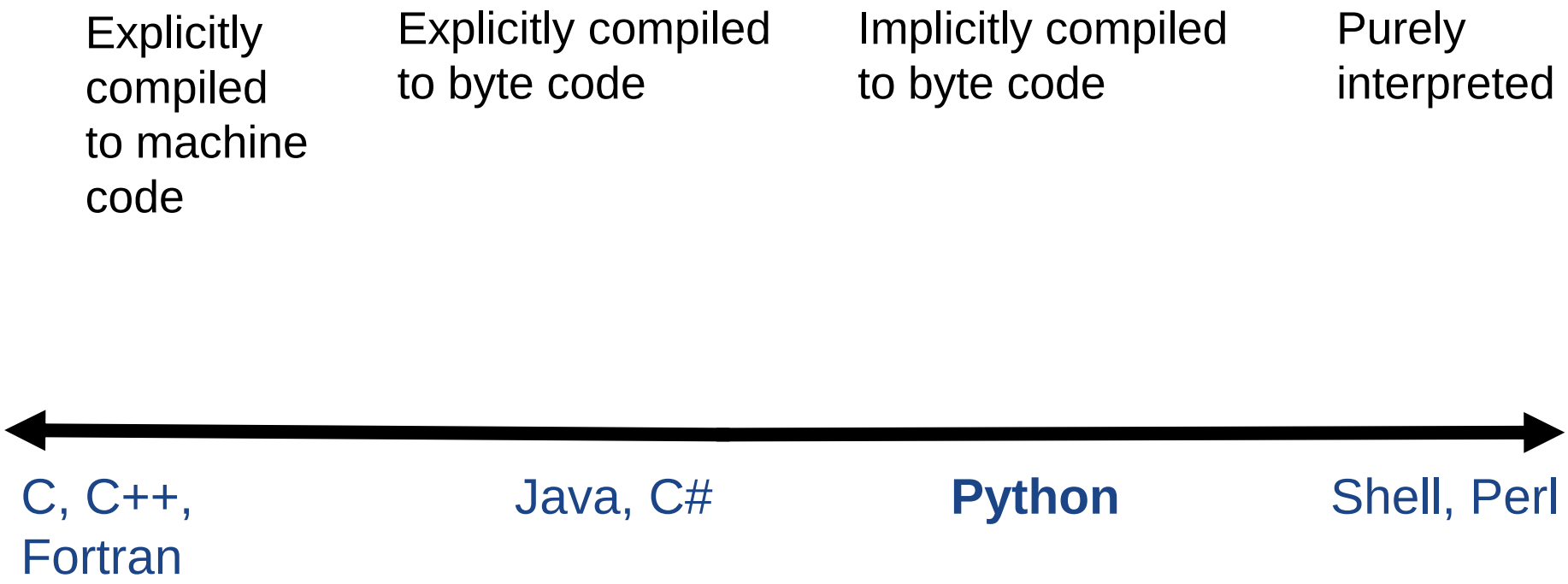


# Introduction

What sort of language is Python?

- Most popular programming languages

**Compiled** **Interpreted**



# Introduction

Who uses Python?

1. Online games
2. Software testing & prototyping
3. Web services / development
4. Automation
5. Science
   - Data science
   - Data analysis
   - Machine learning
   - Data engineering

# Introduction

## Why Python?

1. Use of Modules and Packages
   - Extensive Support Libraries
     - NumPy for manipulation of homogeneous array-based data
     - Pandas for manipulation of heterogeneous and labeled data
     - SciPy for common scientific computing tasks
     - Matplotlib for publication-quality visualizations
     - IPython for interactive execution and sharing of code
     - Scikit-Learn for machine learning
2. Used in everyday life which make it a general purpose for several range of applications
3. Beginner friendly programming language

# Introduction

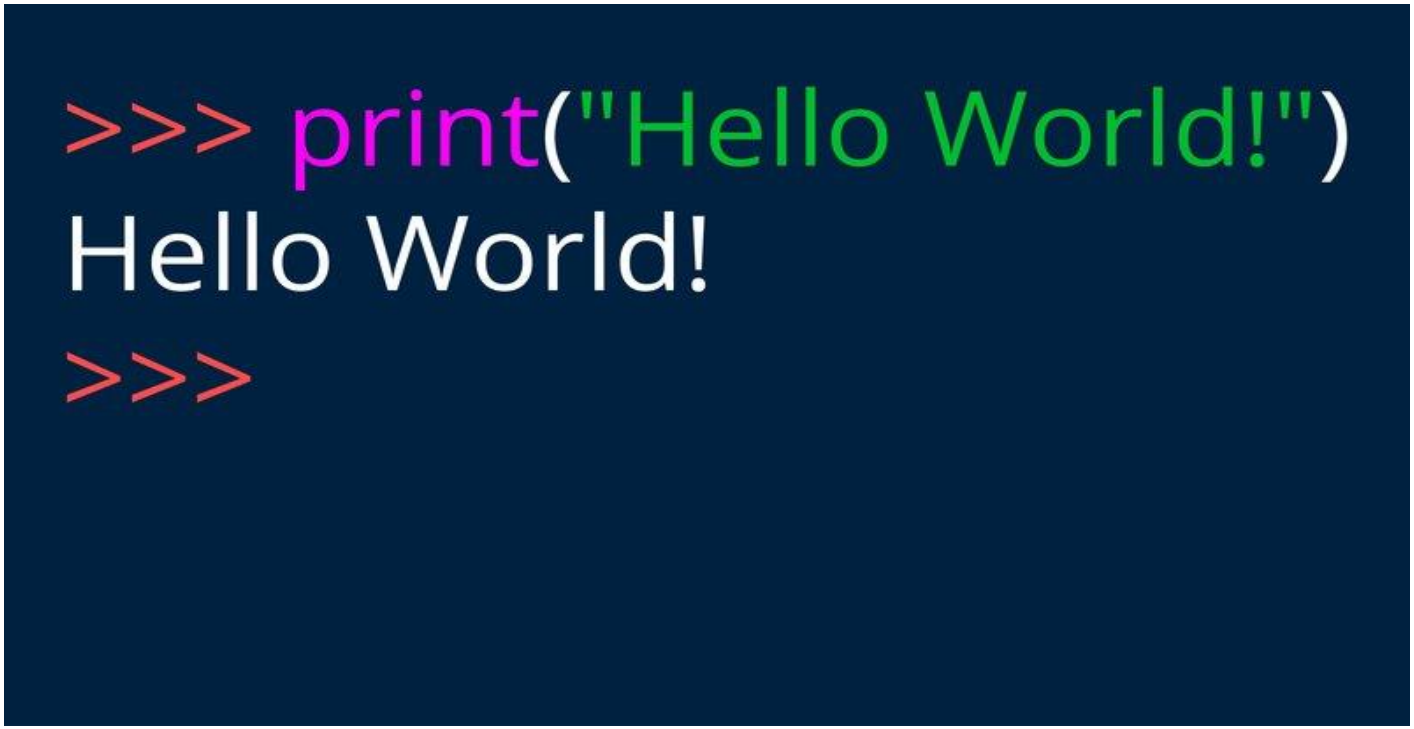
1. Running Python
   - Jupyter notebook
   - IntelliJ IDEA
   - **PyCharm**

# Installation

# Introduction / Syntax

- This notebook introduces all the important Python basics
  - Data Science course.

Python commands!!



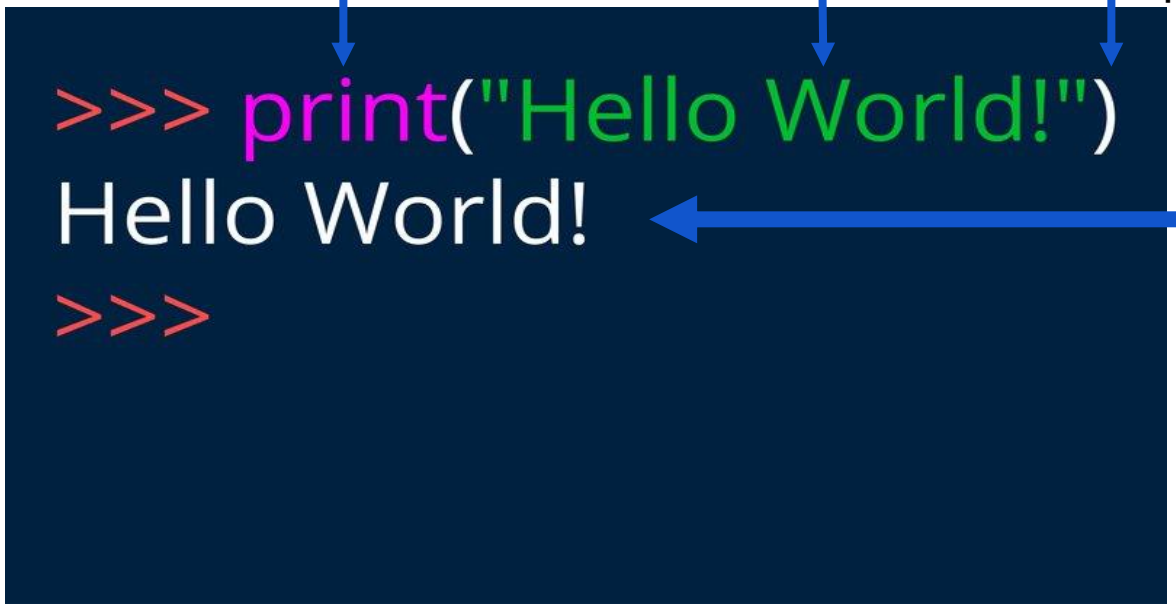
# Introduction / Syntax

Python commands!!

Function's "argument"

Python function

Round brackets — "parentheses"




Output

print ≠ **PRINT**

Case sensitive


# Variables and data types

2

# Variables and data types

1. List elements: A list in Python is an ordered, mutable collection of elements.
   - Lists can contain elements of different types (e.g., integers, strings, ETC).
     - # List of integers
       - numbers = [1, 2, 3, 4, 5]
     - # List of strings
       - fruits = ['apple', 'banana', 'cherry']
     - # Mixed-type list
       - mixed_list = [1, 'apple', 3.5, [2, 3]]
     - # Nested list (list within a list)
       - nested_list = [[1, 2], [3, 4], [5, 6]]
   - Lists are created by placing elements inside square brackets [], separated by **commas**.
2. **Slicing** refers to extracting a subset of elements from a list. In Python, you can slice a list using the syntax:
   - list_name[start:stop:step]

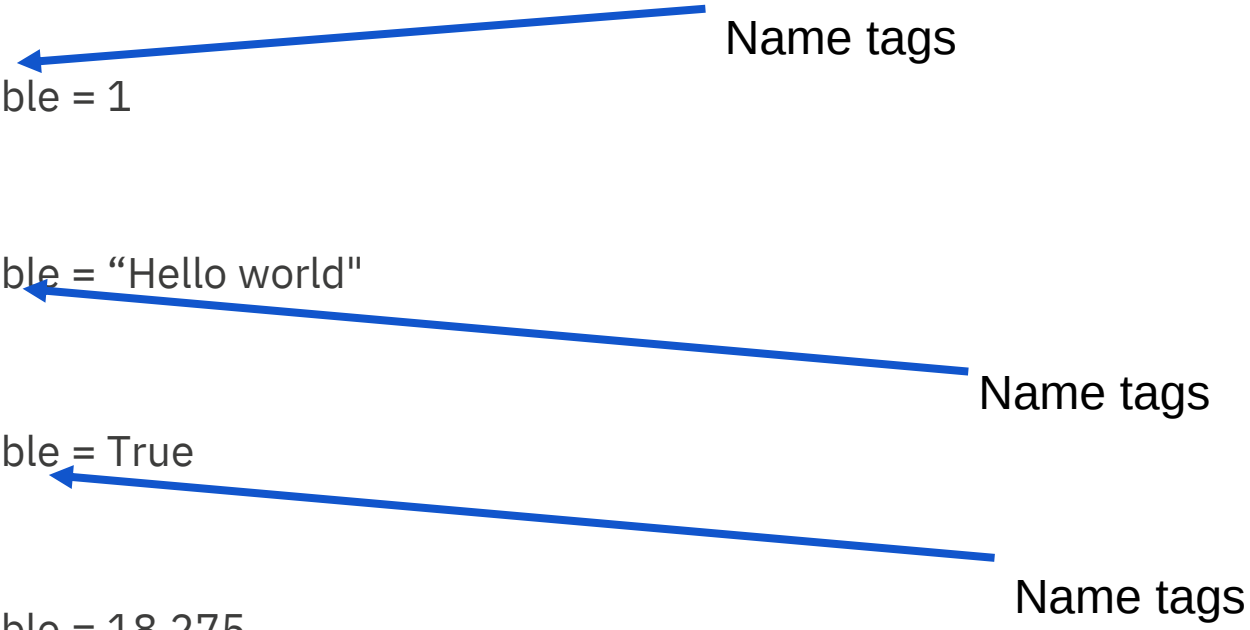
# Variables and data types

1. Sets:
   - Unordered: The elements are not stored in any particular order, and their position can change.
   - Unique: A set cannot contain duplicate elements. If you try to add a duplicate, it will be ignored.
   - Mutable: You can add or remove elements from a set.
   - Defined using curly braces {} or the set() function.

1. Dictionary:
   - Key-value pairs: Each key is mapped to a value (similar to a real-world dictionary where each word is associated with a definition).
   - Keys are unique: A dictionary cannot have two identical keys.
   - Keys must be immutable: Keys can be strings, numbers, or tuples, but not lists or other dictionaries.
   - Values can be of any type: Values can be of any data type, including lists, tuples, or even other dictionaries.
   - Defined using curly braces {} or the dict() function.

# Variables and data types

```
# int
my_variable = 1

# str
my_variable = “Hello world"

# bool
my_variable = True

# float
my_variable = 18.275

# print formatted string
print(f’The temperature is {my_variable:.1f} degrees’)

# type function
type(1.2)
type(3>2)
```



# Variables and data types

```
# list
list1 = [0, 1.0, “two"]
list1[1] = 2.0

# tuple, is unchangeable
tuple1 = (0, 1.0, “two")
tuple1[1] = 2.0        # error

# set
set1 = {5, “six"}

# dict
dict1 = {1:1.0, “s":“six"}
dict1[“s"]

# 2-dimensional list
a = [[1, 2, 3],
     [4, 5, 6],
     [7, 8, 9]]
a[0, 1] = 10           # 2 is replaced by 10, first index is row, second is column
print(a[1])            # gives  [4, 5, 6]
```

# Operators

- Assignment

  =, +=
- Numerical

  +, -, *, /, %, **
- Comparison

  <, >, ==, <=, >=, !=
- Boolean

  not, and, or
- Conversion

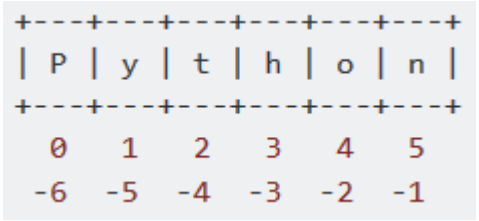
  int(3.2), float(2), str(3)
- String

  .count('x'), .find('x'), .lower(), .upper(), .replace('a', 'b'), .strip()
- Lists

  .append(item), .pop(index), .insert(index, item), .sort()

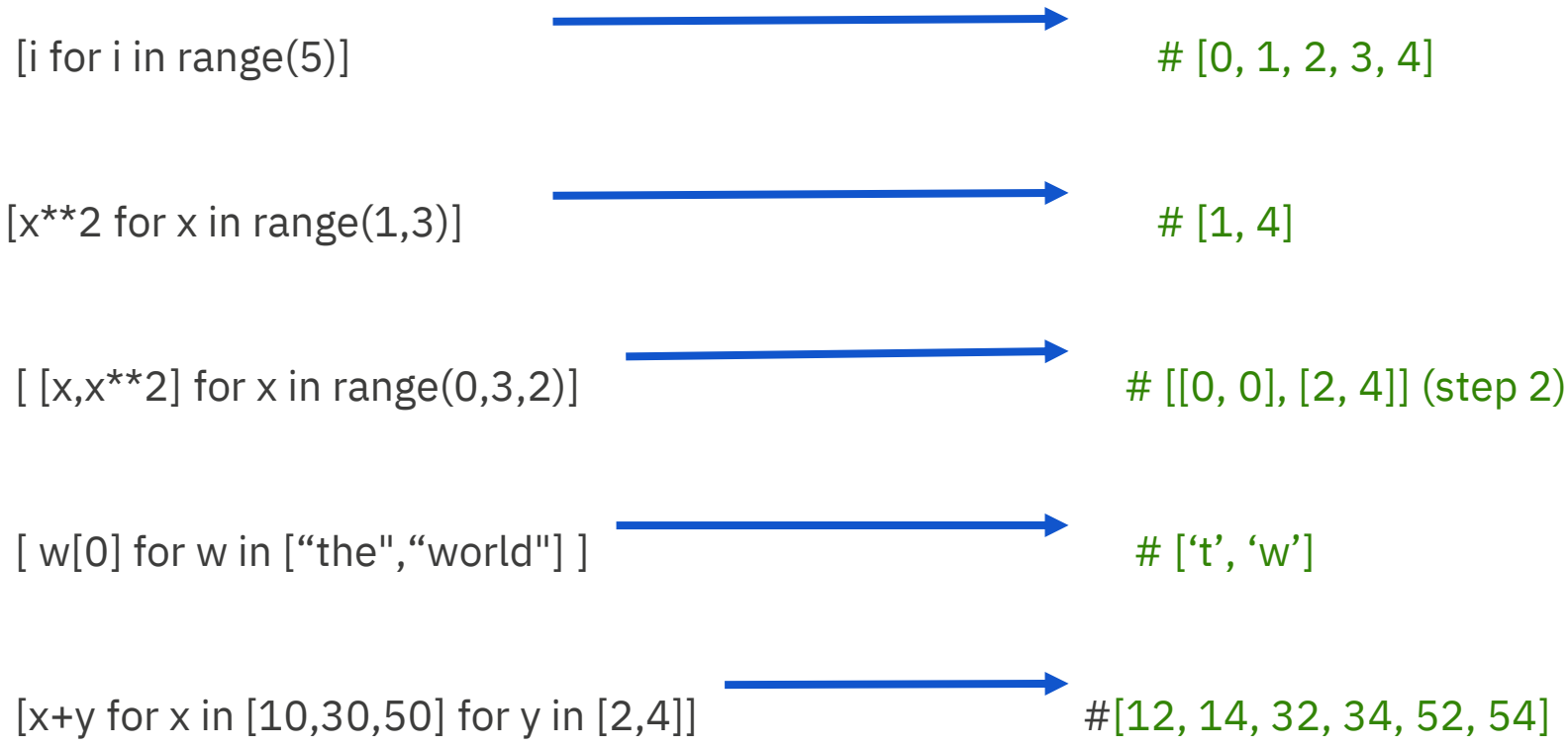
  len(list1), item in list1

# List elements and slicing




```
list = [1, 2, 3, 4, 5, 6, 7 ]
list                                    # [1, 2, 3, 4, 5, 6,
7]
list[0]                                 # 1
list[-2]                                # 6
list[1:4]                               # [2, 3, 4], 1 → 4 (not
including)
list[:3]                                # [1, 2, 3], first 3
list[-2:]                               # [6, 7], last 2
list[:]                                 # [1, 2, 3, 4, 5, 6, 7], a
copy
list[3::2]                              # [4, 6], stap = 2
list[::-1]                              # [7, 6, 5, 4, 3, 2, 1]
```

# List comprehension




# Program flow, functions and modules

# Program flow

```
while True:
    print("Looping")                    # indentation


for i in range(4,12,2):                 # 4 -> 12 (not including), step 2
    print(i**2)


if i <0:
    print("negative")
elif i == 0:
    print(“zero")
else:
    print("positive")
```

# Functions

```
def fac(n):
    if n <= 1:
        return 1
    else:
        return n * fac(n-1)
```

```
def power(base, exponent=2):                    # optional parameters last
    return base ** exponent
```

```
power(2)
power(exponent=1, base=5)
```

# Modules

```
# helloworld.py
def hello():
        print("Hello World")


import helloworld
helloworld.hello()


from helloworld import hello
hello()


from helloworld import *
hello()
```


# Python code guidelines

4

# Python code guidelines

- Indentation:

  4 spaces
- Class:

  CapWords
- Methods/Functions:

  lowercase_with_undercores
- Variables:

  lowercase_with_undercores
- Constants:

  UPPERCASE
- Modules:

  lowercase
- Private attributes or methods:

  _single_leading_underscore or __double_leading_underscore


# Notebook time!

See 00.00 XTR Python – Introduction.ipnb