POLITECNICO
MILANO 1863

DEPARTMENT OF MECHANICAL ENGINEERING

INTRODUCTION TO PYTHON

Matteo Bugatti

mecc

DIPARTIMENTO DI ECCELLENZA
MIUR 2018-2022

Milano

INTRODUCTION TO PYTHON

Python is a high-level, interpreted programming language used for web development, scientific computing, data analysis, and artificial intelligence. It has a simple syntax and a rich standard library.

The First Python version was conceived in the late 80s by Guido van Rossum and released in 1991. Then, major version releases were in 2000 (Python 2.0) and 2008 (Python 3.0). Nowadays it is one of the most used programming languages.

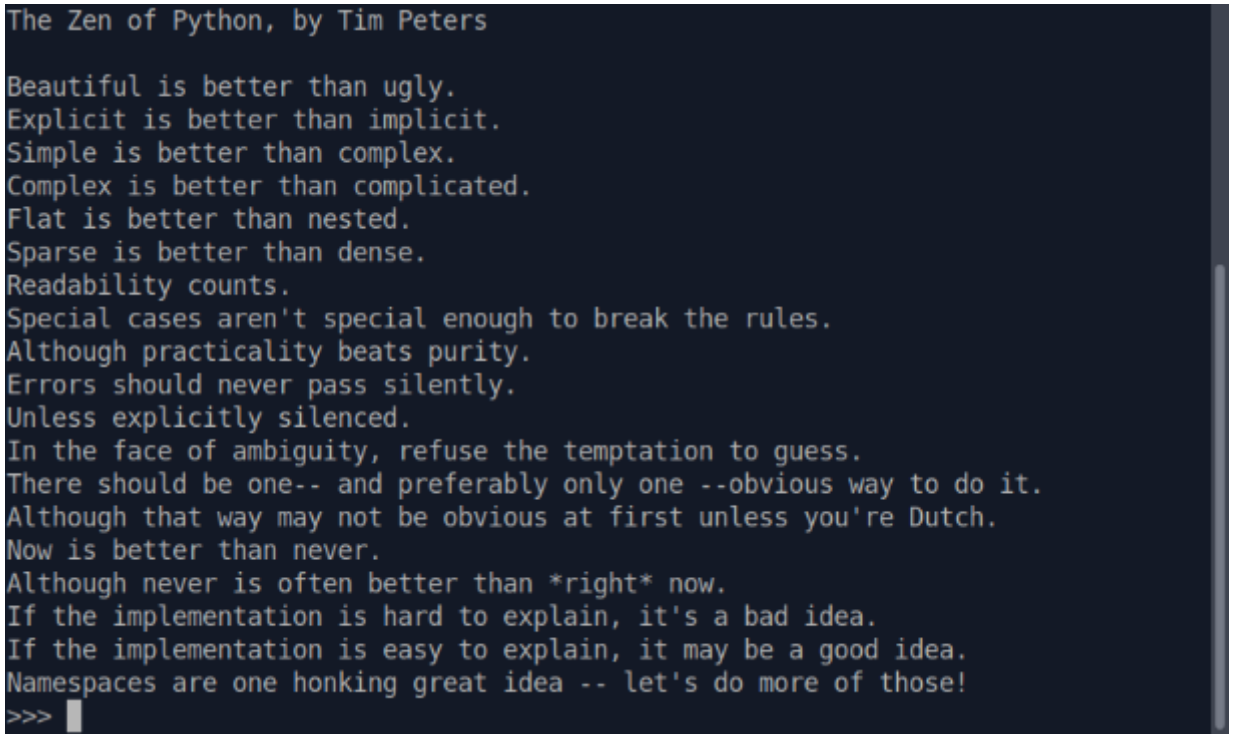
It has an official website where is possible to find documentation, materials, and other useful information:

https://www.python.org/

Beginner's guide to Python: https://wiki.python.org/moin/BeginnersGuide

INTRODUCTION TO PYTHON

Python is very popular because it is easy to be learned, open source, and very versatile. Its philosophy is summarized by the document The Zen of Philosophy by Tim Peters:



INTRODUCTION TO PYTHON

Python is an **object-oriented** programming (OOP) language (such as C++ and Java): data are organized into objects which are instances of classes. Its environment is easy to set up and includes a standard library with numerous modules and the possibility to install and import several libraries. Its **syntax** emphasizes code readability, using **indentation** and explicit constructs rather than brackets hierarchy.

Complete documentation: https://docs.python.org/3

- Lexical hints: https://docs.python.org/3/reference/lexical_analysis.html
- Data types: https://docs.python.org/3/reference/datamodel.html
- Expressions: https://docs.python.org/3/reference/expressions.html
- Statements: https://docs.python.org/3/reference/simple_stmts.html
- Compound statements (if, for, while, …): https://docs.python.org/3/reference/compound_stmts.html

INTRODUCTION TO PYTHON: SYNTAX

Keywords

```
False      await      else       import     pass
None       break      except     in         raise
True       class      finally    is         return
and        continue   for        lambda     try
as         def        from       nonlocal   while
assert     del        global     not        with
async      elif       if         or         yield
```

Operators

```
+       -       *       **      /       //      %       @
<<      >>      &       |       ^       ~       :=
<       >       <=      >=      ==      !=
```

Start a comment

```
#
```

INTRODUCTION TO PYTHON: BASIC FUNCTIONS

The print() function allows to print out a string or a combination of string and numbers.

```
print('Hello world!')
```
[1] ✓ 0.4s

```
Hello world!
```

```
year = 2023
course = 'Quality and Data Analysis'

print('Welcome to the', course, 'course,', 'class of', year)
```
[5] ✓ 0.2s

```
Welcome to the Quality and Data Analysis course, class of 2023
```

INTRODUCTION TO PYTHON: BASIC FUNCTIONS

The printed strings can be also constructed using the appropriate syntax:

- integers `%d`
- float `%f`
- strings `%s`

```
year = 2023
course = 'Quality and Data Analysis'

print('Welcome to the course %s, class of %d' % (course, year))
```

✓ 0.1s

```
Welcome to the course Quality and Data Analysis, class of 2023
```

```
year = 2023
course = 'Quality and Data Analysis'

print('Welcome to the course %s, class of %f' % (course, year))
```

✓ 0.4s

```
Welcome to the course Quality and Data Analysis, class of 2023.000000
```

INTRODUCTION TO PYTHON: COMPOUND STATEMENTS

The for cycle is used to iterate over a sequence of numbers or other iterable objects. The inner part of the cycle must be indented, while the colons in the first line are mandatory. Examples:

```
course = 'QDA 2023'

for num in course:
    print(num)
```

✓ 0.2s

```
Q
D
A


2
0
2
3
```

```
numbers = [1, 2, 3, 4, 5]

for num in numbers:
    print(num)
```

✓ 0.3s

```
1
2
3
4
5
```

```
for i in range(5):
    print(i)
```

✓ 0.2s

```
0
1
2
3
4
```

NOTICE: indexing for numerable objects in python ranges from 0 to n (excluded)!

INTRODUCTION TO PYTHON: COMPOUND STATEMENTS

The while loop is used to repeatedly execute a block of code as long as a certain condition is true:

```
count = 5

while count > 0:
    print(count)
    count -= 1
```

✓ 0.2s

```
5
4
3
2
1
```

INTRODUCTION TO PYTHON: COMPOUND STATEMENTS

The if statement is used to conditionally execute a block of code based on a boolean expression. Is it possible to include one or more elif clauses and an else clause to provide additional conditions.
Remind colons and indentation!

```
condition1 = False
condition2 = True

if condition1:
    # body of the if statement
    print('Condition 1 is true')
elif condition2:
    # body of the elif statement
    print('Condition 2 is true')
else:
    # body of the else statement
    print('All conditions are false')
```

✓ 0.2s

```
Condition 2 is true
```

```
num = 5
threshold1 = 10
threshold2 = -20

if num > threshold1:
    # body of the if statement
    print('Condition 1 is true')
elif num <= threshold2:
    # body of the elif statement
    print('Condition 2 is true')
else:
    # body of the else statement
    print('All conditions are false')
```

✓ 0.3s

```
All conditions are false
```

INTRODUCTION TO PYTHON: LIBRARIES FOR THE COURSE

https://numpy.org/doc/stable/user/

https://pandas.pydata.org/docs/user_guide/index.html#user-guide

https://matplotlib.org/stable/users/index.html

https://docs.scipy.org/doc/scipy/reference

INTRODUCTION TO PYTHON: NUMPY

Numpy is a library for Python that is widely used for scientific and technical computing. It provides a array object and a variety of functions, such as mathematical operations, and linear algebra. The main advantages of using Numpy are its speed, convenience, and compatibility with other scientific libraries.

```python
# Import the library
import numpy as np

# Creating a one-dimensional array
a = np.array([1, 2, 3, 4])
print('a =', a)

# Creating a two-dimensional array
b = np.array([[1, 2], [3, 4]])
print('b = ', b)
```

✓ 0.2s

```
a = [1 2 3 4]
b =  [[1 2]
 [3 4]]
```

INTRODUCTION TO PYTHON: NUMPY

Elements of an array can be accessed using brackets [ ]. Remind: elements are numbered starting from 0

```
print('first element of a is: ', a[0])

print('element in position [0,1] of b is: ', b[0,1])

print('the first column of b is: ', b[:,0])

print('the second row of b is: ', b[1,:])
```

✓ 0.3s

```
first element of a is:  1
element in position [0,1] of b is:  2
the first column of b is:  [1 3]
the second row of b is:  [3 4]
```

INTRODUCTION TO PYTHON: NUMPY

Useful functions to define arrays are:

- `np.arange(start, stop, step)` to create an evenly spaced array of values from start to stop
- `np.linspace(start, stop, n)` to evenly cover a range with n values

```
c = np.arange(-5, 5, 1)
print('c = ', c)

d = np.linspace(-5, 5, 11)
print('d = ', d)
```

✓ 0.2s

```
c =  [-5 -4 -3 -2 -1  0  1  2  3  4]
d =  [-5. -4. -3. -2. -1.  0.  1.  2.  3.  4.  5.]
```

INTRODUCTION TO PYTHON: NUMPY

Numpy library can be used to compute common descriptive statistics and create random data from a given distribution

```
# Calculate the mean of a column
df['cond1'].mean()
```

```
16.322222222222223
```

```
# Generate a random array of 1000 elements from a normal distribution
mu = 10          # mean
sigma = 2.5      # standard deviation
y = np.random.normal(mu, sigma, 1000)
```

- `mean()` mean
- `std()` standard deviation
- `var()` variance
- `min()` minimum
- `max()` maximum
- `median()` median
- `percentile()` for computing any percentile or quartiles (Q1 = 25 percentile, Q3 = 75 percentile)

Detailed documentation: https://numpy.org/doc/stable/reference/routines.statistics.html

INTRODUCTION TO PYTHON: PANDAS

Pandas is a powerful library for data manipulation and analysis. It provides data structures such as Series (1-dimensional) and DataFrame (2-dimensional) for storing and manipulating data.
The library is also useful to import the CSV files for all the exercise classes.

```python
import pandas as pd

# Loading a CSV file into a DataFrame
df = pd.read_csv('data.csv')
```

INTRODUCTION TO PYTHON: PANDAS

Some basic and typical operations on DataFrames are: display the head of DF, change specific values by index or by using a conditional statement

```
# Display the first 3 rows of a DataFrame
df.head(3)
```

	cond1	cond2
0	19.8	14.9
1	18.5	12.7
2	17.6	11.9

```
# Change the value of a cell
df.loc[0,'cond1'] = 20.0

# See if the change was made
df.head()
```

	cond1	cond2
0	20.0	14.9
1	18.5	12.7
2	17.6	11.9
3	16.7	11.4
4	16.7	10.1

INTRODUCTION TO PYTHON: PANDAS

Some basic and typical operations on DataFrames are: display the head of DF, change specific values by index or by using a conditional statement

```
# Find the row equal to 17.6 value in the first column
# and change the value to 17.0
df.loc[df['cond1'] == 17.6, 'cond1'] = 17.0

# See if the change was made
df.head()
```

	cond1	cond2
0	20.0	14.9
1	18.5	12.7
2	17.0	11.9
3	16.7	11.4
4	16.7	10.1

```
# Find the row with the maximum value in a column and change the value
df.loc[df['cond1'].idxmax(), 'cond1'] = 20.1

# See if the change was made
df.head()
```

	cond1	cond2
0	20.1	14.9
1	18.5	12.7
2	17.0	11.9
3	16.7	11.4
4	16.7	10.1

INTRODUCTION TO PYTHON: MATPLOTLIB

Matplotlib is a powerful library for creating static, animated, and interactive data visualizations.

```
import matplotlib.pyplot as plt
```

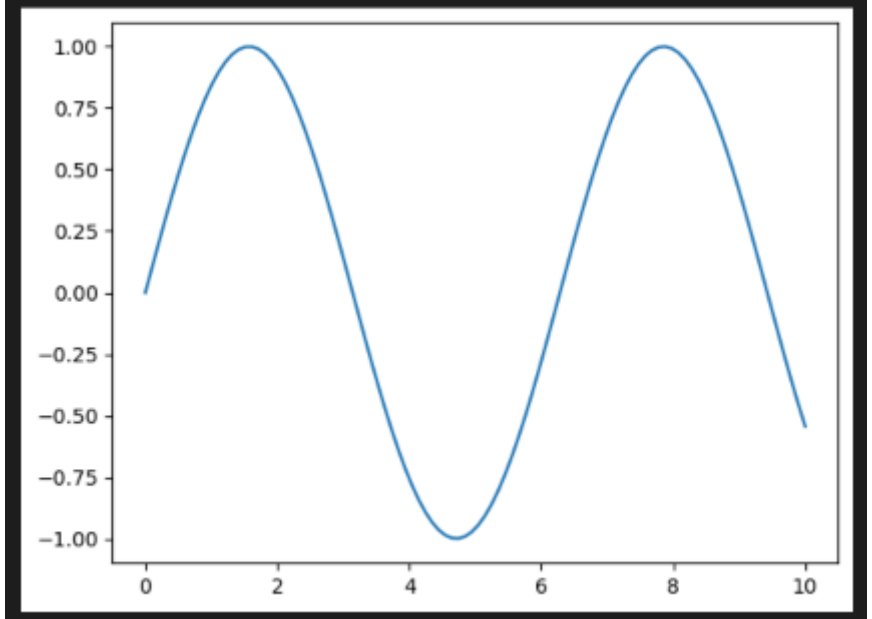
Matplotlib can be used to:

- Show a plot
- Show a scatterplot
- Generate and plot random data from given distributions
- Plot histograms
- Create Boxplots
- Edit graphs with labels, legends, and markers for multiple data plots

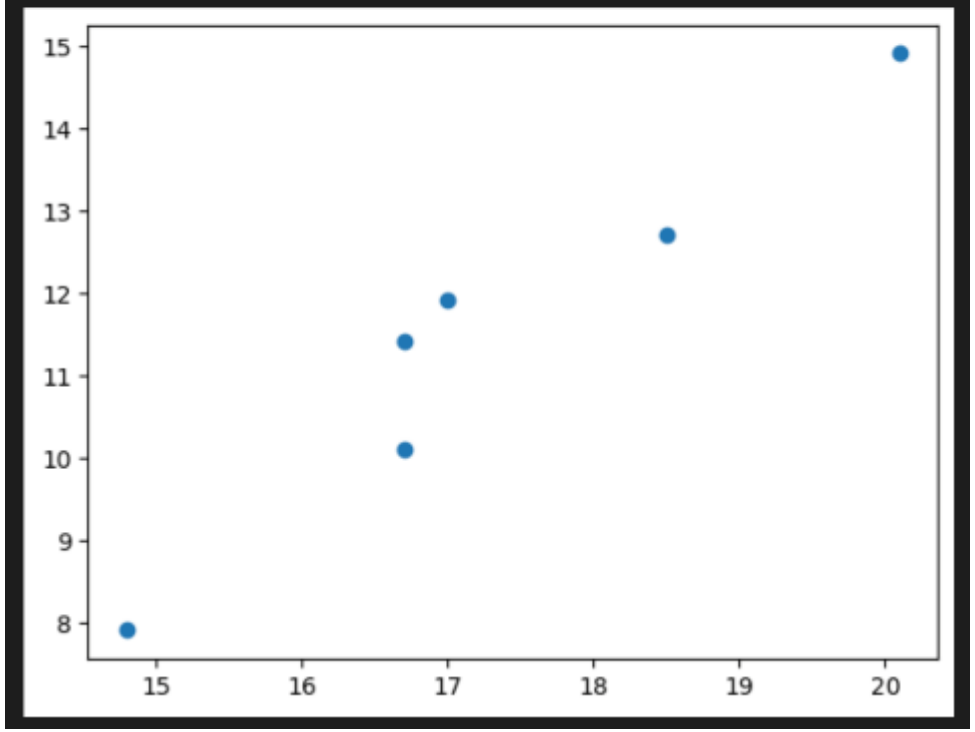
INTRODUCTION TO PYTHON: MATPLOTLIB

Example of plot and scatterplots using functions plt.plot() and plt.scatter()

```
import matplotlib.pyplot as plt
# Create some data
x = np.linspace(0, 10, 100)
y = np.sin(x)

# Create a line plot
plt.plot(x, y)

# Show the plot
plt.show()
```



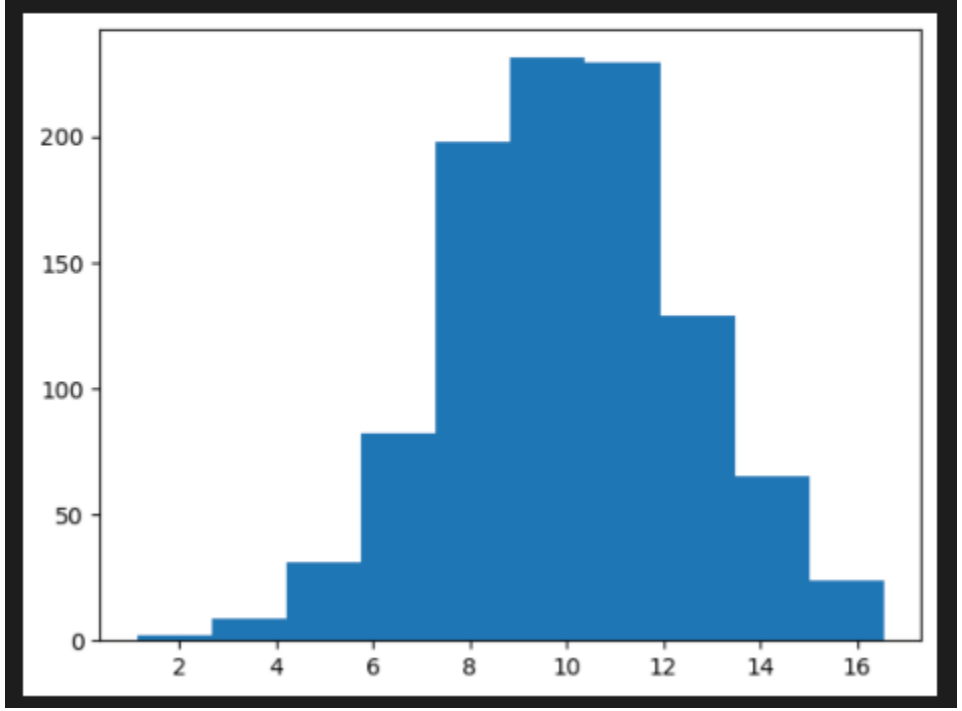

```
# Create a scatter plot
plt.scatter(df['X'], df['Y'])

# Show the plot
plt.show()
```



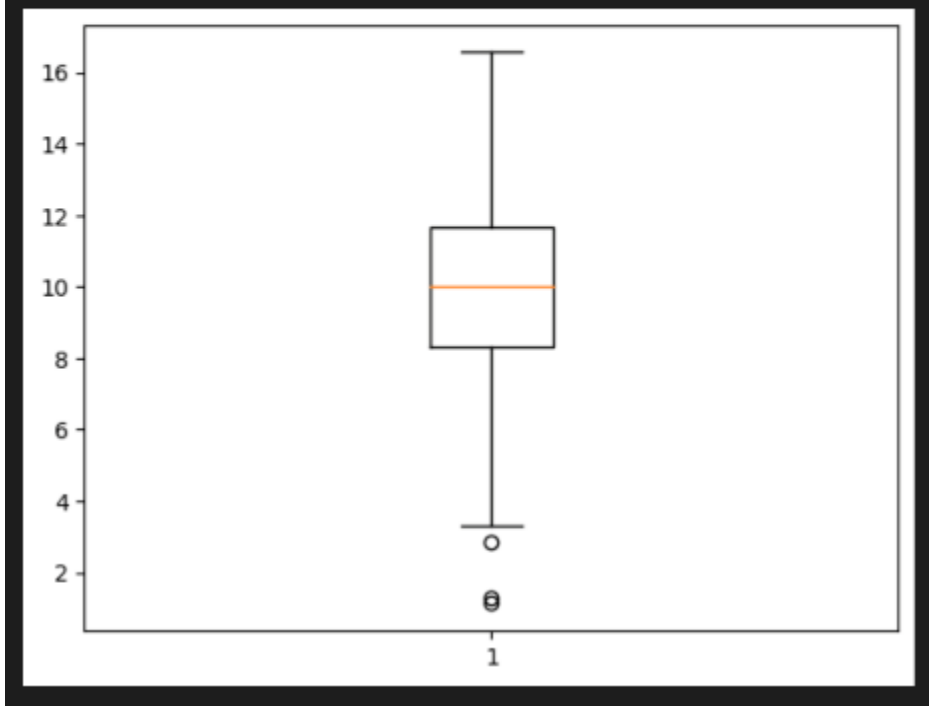
INTRODUCTION TO PYTHON: MATPLOTLIB

Example of plot and scatterplots using functions plt.hist() and plt.boxplot()

```
# Create a histogram
plt.hist(y)

# Show the plot
plt.show()
```




```
# Boxplot of the random data drawn from the normal distribution
plt.boxplot(y)

# Show the plot
plt.show()
```



INTRODUCTION TO PYTHON: SCIPY

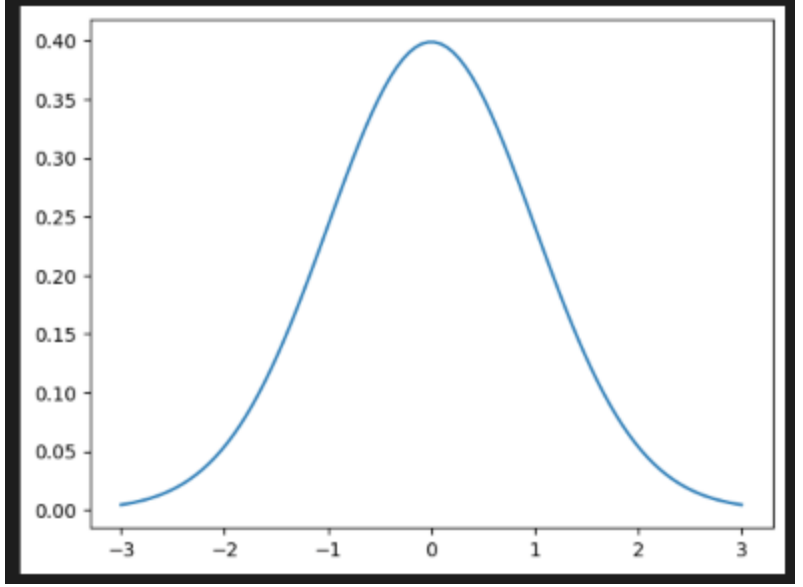
Scipy is a python library that is widely used for scientific and technical computing. For the purpose of the course, the `stats module` in scipy is used for probability distributions and statistical functions.

```
from scipy import stats
```

scipy.stats has a wide range of probability distributions that can be used for statistical analysis. Some of the commonly used distributions are:

- Normal Distribution (`norm`)
- t-Student (`t`)
- Chi-squared (`chi2`)
- F (`f`)
- etc.

```
# Plot the probability distribution function of a normal distribution
x = np.linspace(-3, 3, 100)
mu = 0
sigma = 1
y = stats.norm.pdf(x, mu, sigma)
plt.plot(x, y)
plt.show()
```



USEFUL LINKS:

https://www.python.org/

https://jakevdp.github.io/PythonDataScienceHandbook/

Others:

https://wiki.python.org/moin/BeginnersGuide

https://numpy.org/doc/stable/user/

https://numpy.org/doc/stable/reference/routines.statistics.html

https://pandas.pydata.org/docs/user_guide/index.html#user-guide

https://matplotlib.org/stable/users/index.html

https://docs.scipy.org/doc/scipy/reference

DIPARTIMENTO DI ECCELLENZA
MIUR 2018-2022

CONTACTS

Prof.

Matteo Bugatti

matteo.bugatti@polimi.it

www.mecc.polimi.it

@meccpolimi

POLITECNICO
MILANO 1863