

Introduzione al SO Ubuntu e alla programmazione in Python



DI
C
Ma
PI

Dipartimento
di Ingegneria Chimica,
dei Materiali e della
Produzione Industriale
Università degli Studi
di Napoli Federico II

Speaker: Eng. Giulio Mattera

mail: giulio.mattera@unina.it



DI
C
Ma
PI
Dipartimento
di Ingegneria Chimica,
dei Materiali e della
Produzione Industriale
Università degli Studi
di Napoli Federico II

Linux (I)

- Linux si riferisce a una famiglia di sistemi operativi modellati su Unix.
- Può eseguire molte delle stesse funzioni di Windows.
- Chiunque può usare, modificare o distribuire il kernel Linux.
- Chiunque può sviluppare software da far girare sul kernel Linux.
- Molti sistemi operativi Linux e programmi aggiuntivi sono gratuiti.



Linux (I)

- Linux si riferisce a una famiglia di sistemi operativi modellati su Unix.
- Può eseguire molte delle stesse funzioni di Windows.
- Chiunque può usare, modificare o distribuire il kernel Linux.
- Chiunque può sviluppare software da far girare sul kernel Linux.
- Molti sistemi operativi Linux e programmi aggiuntivi sono gratuiti.

Ci sono molti gusti diversi (OS) costruiti a partire dal kernel Linux kernel.

- **Ubuntu:** La versione più popolare. È gratuito ed è molto facile da usare.
- **Mint:** Una variante popolare di Linux, simile all'ambiente Windows.
- **Red Hat:** Progettato da una società che sviluppa versioni specializzate per governo e grandi aziende.
- **Fedora:** Una versione open-source e gratuita di Red Hat. Usata frequentemente come test per i programmi Red Hat.

Queste versioni sono simili a livello di base, ma possono avere interfacce molto diverse e comandi specializzati.



Source: <http://www.ubuntu.com/>



Source: <http://www.linuxmint.com/>



Source: <http://fedoraproject.org/>

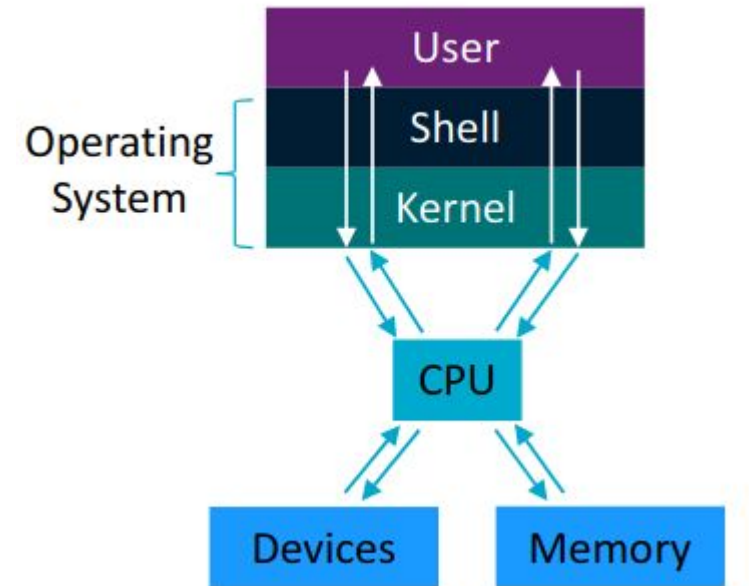
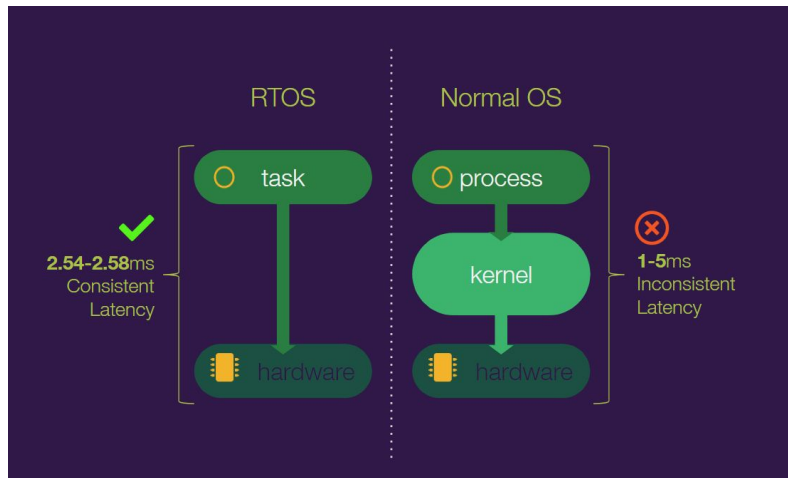


Source: <http://www.redhat.com/>

Linux (II)

Un kernel è il componente centrale di un sistema operativo. Ed è l'elemento che gestisce le risorse di sistema (memoria, processi, dispositivi di input e dispositivi di input e output).

Quando un utente fa qualcosa nella shell il kernel traduce il comando e gli dà la priorità rispetto ad altre richieste di risorse, in modo che possa essere compreso ed eseguito dalla CPU.



Linux (III)

- Gratis
- Aperto
- Più sicuro (meno malware)
- Certi hardware non lavorano con linux
- Non tutti i programmi sono sviluppati per poter lavorare in linux

Linux (IV)

- **Root** - l'account amministratore di Linux. Come l'amministratore integrato in Windows, Linux è dotato di un account di root e quando qualcuno ottiene i permessi di root, può accedere a tutti i file e eseguire tutti i comandi su un sistema, così come impostare le politiche per gli altri utenti. Gli utenti root sono autorizzati a fare molti compiti diversi, ma devono prima autenticare la loro identità inserendo la loro password.
- **Percorsi:** In ubuntu non si specifica su quale unità è memorizzata una cartella e si usa slash (/) per identificare le directory principali.

Esempi

Windows C:\Documents\hello.txt

Linux /home/CyberTaipan/hello.txt

Cartelle importanti

- **/home:** memorizza i documenti di ogni utente, i file multimediali, ecc. Gli utenti possono solo accedere solo alle proprie cartelle, a meno che non abbiano ottenuto i permessi di root.
- **/boot:** contiene i file di avvio e i file del kernel. Non dovrebbe essere modificato a meno che tu non sia un utente esperto.

Linux (V)

- Il terminale può essere aperto con Ctrl+Alt+T
 - Vantaggi:
 - Utilizza meno risorse dell GUI
 - Maggior controllo delle richieste al kernel (non passo per la GUI)
 - Svantaggi:
 - No user-friendly
 - Complicato per multi-tasking



Il comando dice al computer cosa fare e tutti gli altri componenti della sintassi dipendono da cos'è il comando. In questo esempio:

- Il comando 'cat' crea, visualizza o copia file.

L'opzione personalizza l'output del comando.

- '-n' dice al computer di aggiungere un numero ad ogni riga di testo nel file creato.

L'effetto di un'opzione varia a seconda del comando e non è richiesta per tutti i comandi.

Operator dirige l'output del comando. Non richiesto per tutti i comandi.

- File Name/Location dice al sistema operativo in quale file volete che il comando e le opzioni vengano eseguiti.



Linux : Alcuni comandi (VI)

- **pwd (print working directory)**
 - indica la cartella in cui ti trovi (absolute path)
- **ls**
 - print a terminale di tutte le sotto-cartelle e file che si trovano nella cartella di apertura del terminale
- **cd**
 - entra in una cartella nuova
- **mkdir & rmdir**
 - mkdir crea cartella
 - rmdir elimina cartella
- **touch:**
 - crea un file. Per esempio .txt vuoto per codice

```
giulio@giulio: ~/model
giulio@giulio:~$ pwd
/home/giulio
giulio@giulio:~$ ls
'2022-03-18 12-47-32.mkv'      matlab_crash_dump.182728-1
anaconda3                    model
catkin_ws                    Modelli
Documenti                    Musica
Downloads                     nvidia
ESE_traini.mkv               project_ws
ese_train.mkv                 Pubblici
GitHubDesktop-linux-2.9.3-linux3.deb  rl_prj
Immagini                      Scaricati
java.log.51656                Scrivania
libfranka                     slprj
matlab                        snap
MATLAB                        src
'MATLAB Add-Ons'              teams_1.3.00.16851_amd64.deb
                               Video
giulio@giulio:~$ cd model
giulio@giulio:~/model$ pwd
/home/giulio/model
giulio@giulio:~/model$
```


Linux : Alcuni comandi (VI)

- **chmod**

- Si utilizza per rendere **eseguibili** i file [-x option] e per cambiare i permessi [a+rw option]
- `chmod +x numbers.py` rende eseguibile il file python
- Utile per accesso a porte esterne come USB
- `chmod 777 /dev/ttyUSB0`

- **sudo** (SuperUser Do)

- Prima di un comando gli dona i privilegi del root

drwxrwxrwx

d = Directory
r = Read
w = Write
x = Execute

7	rwX	111
6	rw-	110
5	r-X	101
4	r--	100
3	-wX	011
2	-w-	010
1	--X	001
0	---	000

chmod 777

rwX | rwX | rwX
Owner | Group | Others

Linux : APT (VI)

In ubuntu il gestore di pacchetti si chiama APT (Advanced Packaging Tool).

APT semplifica il processo di gestione del software su sistemi informatici Unix-like automatizzando il reperimento, la configurazione e l'installazione dei pacchetti software.

Per installare un nuovo software, dovresti usare la seguente sintassi:

- `sudo apt-get install [NOME PACCHETTO]`

Il comando `update` è usato per risincronizzare i file indice dei pacchetti dalle loro fonti ed è consigliato prima dell'installazione di un nuovo pacchetto (per le dipendenze).

- `sudo apt-get update`

Il comando `upgrade` viene usato per installare le versioni più recenti di tutti i pacchetti attualmente installati sul sistema

- `sudo apt-get upgrade`

Linux : Package Installer for Python (V)

- The pip command stands for **P**ackage **I**nstaller for **P**ython. Similar to the apt command for Debian-based distributions, yum and rpm commands for Red Hat-based distributions, and pacman for Arch-based distributions, pip command helps install packages for Python.

```
root@ubuntu:~# apt -y install python-pip
OR
root@ubuntu:~# apt -y install python3-pip
```

```
root@ubuntu:~# pip3 install <package name>
For Example --
root@ubuntu:~# pip3 install numpy
```

```
root@ubuntu:~# pip3 install numpy
Collecting numpy
  Downloading https://files.pythonhosted.org/packages/62/20/4d43e141b5bc426ba38274931
  (20.2MB)
    100% |████████████████████████████████████████| 20.2MB 52kB/s
Installing collected packages: numpy
Successfully installed numpy-1.18.1
root@ubuntu:~#
```

```
root@ubuntu:~# pip3 uninstall numpy
```

```
root@ubuntu:~# pip3 list
```

Python (I)



- Python è un linguaggio di programmazione open source di alto livello **interpretato**.
- E' un linguaggio general purpose, molto utilizzato per applicazioni di IA e CV.
- I programmi che sono scritti in linguaggi compilati macchina nativo tendono ad essere più veloci dei codici interpretati perché si aggiunge al tempo di esecuzione il tempo per effettuare processo di traduzione in codice macchina (overhead), però i primi presentano generalmente una sintassi meno complessa e più orientata al linguaggio umano.

Python (II)

Term	Definition
program	a sequence of instructions that designate how to execute a computation
programming	taking a task and writing it down in a programming language that the computer can understand and execute

Term	Definition
Integer	Positive or negative whole numbers without a decimal point <i>Example: 5, 10, -3, -15</i>
Floating point (float)	Real numbers. Hence, they have a decimal point <i>Example: 4.75, -5.50, 11.0</i>
Boolean value	a True or False value, corresponding to the machine's logic of understanding 1s and 0s, on or off, right or wrong, true or false. <i>Example: True, False</i>

Python (III)

Operator	Description
+	Addition
-	Subtraction
/	Division <i>Note: If you want to divide 16 by 3, when you use Python 2, you should look for the quotient of the float 16 divided by 3 and not of the integer 16 divided by 3. So, you should either transform the number into a float or type it as a float directly.</i>
%	Returns remainder
*	Multiplication
**	Performs power calculation

Python (IV)

Operator	Description
==	Verifies the left and right side of an equality are equal
!=	Verifies the left and right side of an equality are not equal
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

Python (V)

Function	Description
<code>type()</code>	obtains the type of variable you use as an argument
<code>int()</code>	transforms its argument in an <i>integer</i> data type
<code>float()</code>	transforms its argument in a <i>float</i> data type
<code>str()</code>	transforms its argument in a <i>string</i> data type
<code>max()</code>	Returns the highest value from a sequence of numbers
<code>min()</code>	Returns the lowest value from a sequence of numbers
<code>abs()</code>	Allows you to obtain the absolute value of its argument
<code>sum()</code>	Calculates the sum of all the elements in a list designated as an argument

Function	Description
<code>round(x,y)</code>	returns the float of its argument (x), rounded to a specified number of digits (y) after the decimal point
<code>pow(x,y)</code>	returns x to the power of y
<code>len()</code>	returns the number of elements in an object

Python (VI)

Compare returns a boolean "=="

```
In [1]: y = 5 ** 3
```

```
In [2]: y
```

```
Out[2]: 125
```

```
In [3]: y == 125
```

```
Out[3]: True
```

```
In [4]: y == 126
```

```
Out[4]: False
```

Variable assignment and change using «=»

```
In [1]: z = 1  
z
```

```
Out[1]: 1
```

```
In [2]: z = 3  
z
```

```
Out[2]: 3
```

```
In [3]: z + 5
```

```
Out[3]: 8
```

```
In [4]: z = 7  
z
```

```
Out[4]: 7
```

Python : Esempio da terminale

```
giulio@giulio: ~/Scrivania
giulio@giulio:~$ cd Scrivania
giulio@giulio:~/Scrivania$ ls
CASTING_CONTROLLER_SIL.slx
'Deep neural networks for defects detection in Gas Metal Arc Welding.pdf'
fatture
NotosVisionSys
PhD
ReportIMG
giulio@giulio:~/Scrivania$ touch my_python.py
giulio@giulio:~/Scrivania$ ls
CASTING_CONTROLLER_SIL.slx
'Deep neural networks for defects detection in Gas Metal Arc Welding.pdf'
fatture
my_python.py
NotosVisionSys
PhD
ReportIMG
giulio@giulio:~/Scrivania$
```

- Entro nella cartella scrivania
- creo file my_python.py
- uso ls per vedere il file nelle cartella

Python : Esempio da terminale

```
giulio@giulio:~/Scrivania$ sudo nano my_python.py
giulio@giulio:~/Scrivania$ python my_python.py
('La somma e ', 13)
giulio@giulio:~/Scrivania$
```

```
GNU nano 4.8 my_python.py
print('La somma e ', 6+7)
```

[Letta 1 riga]

^G Guida	^O Salva	^W Cerca	^K Taglia	^J Giustifica	^C Posizione
^X Esci	^R Inserisci	^I Sostituisci	^U Paste Text	^T Ortografia	^_ Vai a riga

- Entro nella cartella scrivania
- creo file my_python.py
- uso ls per vedere il file nella cartella
- uso il comando nano per scrivere nel file my_python
- lancio il file con il comando python

Python: Le librerie



Working with n -D array



Data visualization



Machine learning library



Neural networks framework



IDE

<https://www.anaconda.com/products/individual>

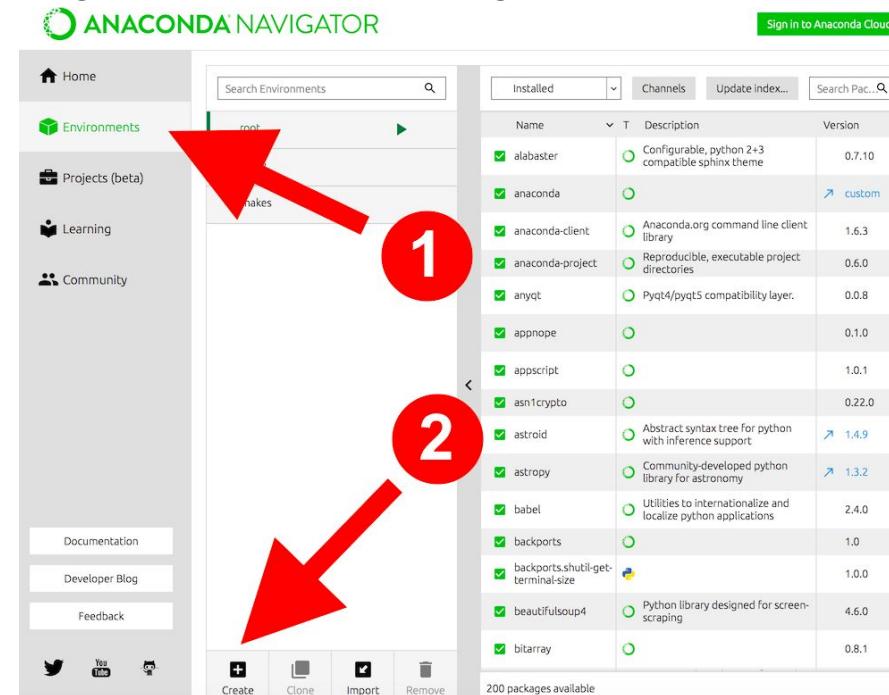
<https://problemsolvingwithpython.com/01-Orientation/01.03-Installing-Anaconda-on-Windows/>

<https://docs.anaconda.com/anaconda/user-guide/tasks/tensorflow/>

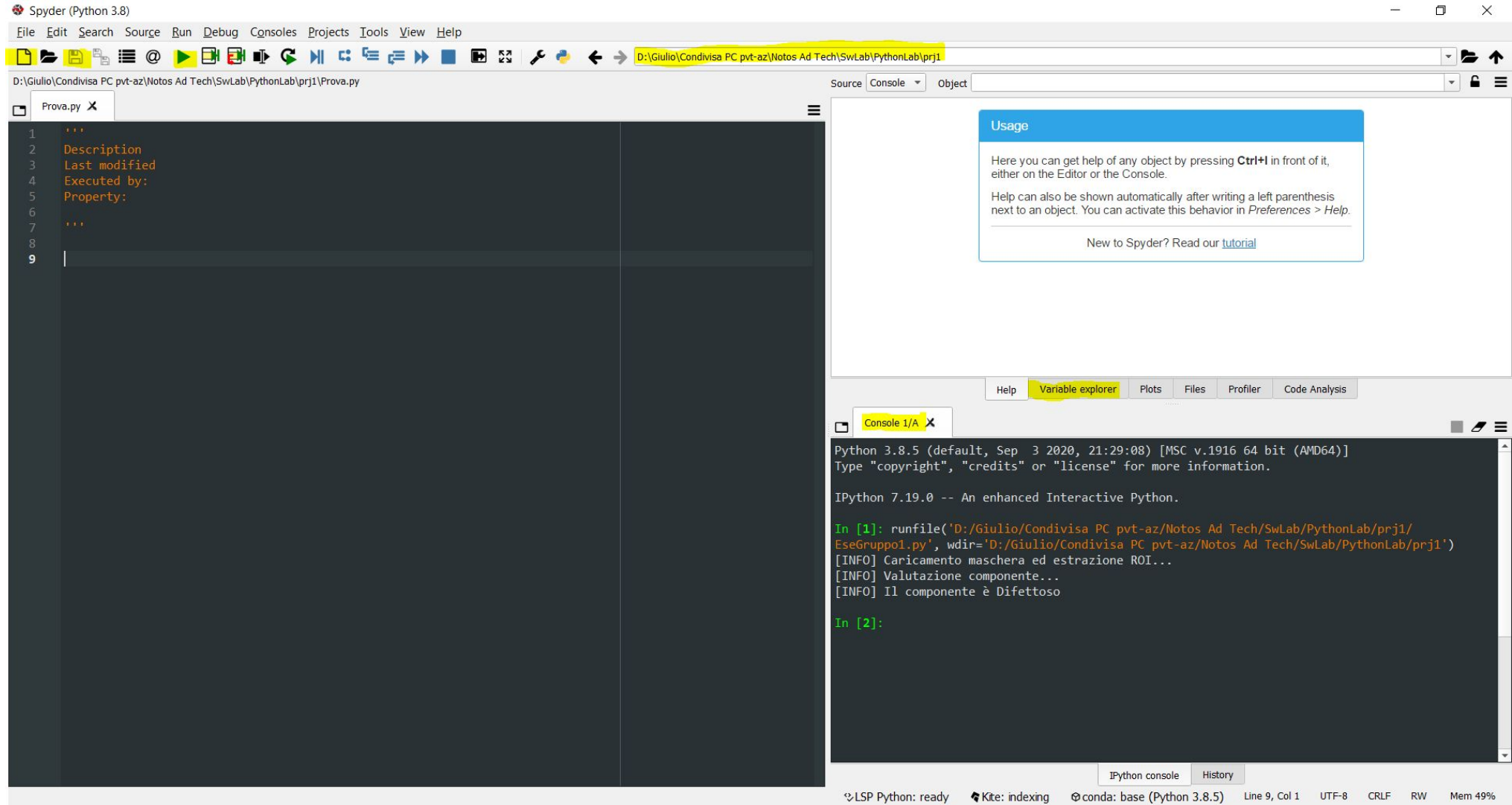
Anaconda



- Anaconda aims to simplify package management and deployment.
- Package versions in Anaconda are managed by the package management system, conda, which analyzes the current environment before executing an installation to avoid disrupting other frameworks and packages
 - The big difference between conda and the **pip package manager** is in how package dependencies are managed, which is a significant challenge for Python data science. When pip installs a package, it automatically installs any dependent Python packages without checking if these conflict with previously installed packages.



Python: La GUI di Spyder



Python: Funzione print

La funzione **print()** viene utilizzata per stampare a schermo o nel terminale un qualsiasi risultato o avviso per l'utente:

```
In [1]: runfile('D:/Giulio/Condivisa PC pvt-az/Notos Ad Tech/SwLab/PythonLab/prj1/
EseGruppo1.py', wdir='D:/Giulio/Condivisa PC pvt-az/Notos Ad Tech/SwLab/PythonLab/prj1')
[INFO] Caricamento maschera ed estrazione ROI...
[INFO] Valutazione componente...
```

E' possibile continuare una qualsiasi operazione (definizioni di argomenti di funzioni, somme, testi) alla riga successive attraverso il simbolo `\`, oppure scrivere sulla stessa riga operazioni diverse seprandole con un `;` ;

In Python non è necessario dichiarare il tipo della variabile. Automaticamente viene assegnato il tipo più opportuno in funzione del valore della variabile:

es:

Var = 'ciao' -> str

Var = 6 -> int

Var = 4.3 -> float

```
1  '''
2  Description
3  Last modified
4  Executed by:
5  Property:
6
7  '''
8
9  b = 2; c = 3; d = 1
10
11  a = 5 + 4 + 2 + \
12      3 + 6 + 7 + \
13          2
14
15
16  print('[INFO] il risultato è', a + c + b + d)
17
```


Python: L'indentazione e costrutto if-else

In altri linguaggi è solito utilizzare la parentesi graffa per indicare quando inizia e quando finisce un operazione, come un ciclo for o un while. In python viene utilizzato a tale scopo **l'indentazione**.

```
1  '''
2  Description
3  Last modified
4  Executed by:
5  Property:
6
7  '''
8  #Commento
9  for i in range(10):
10
11      if i < 5:
12          print ('[INFO] Flag basso')
13          out = 0
14
15      elif i > 5 and i < 8:
16          print ( '[INFO] Flag alto')
17          out = 1
18      else:
19          print ('[INFO] Flag basso')
20          out = 0
21
22  print ('[INFO] Flag value', out)
```

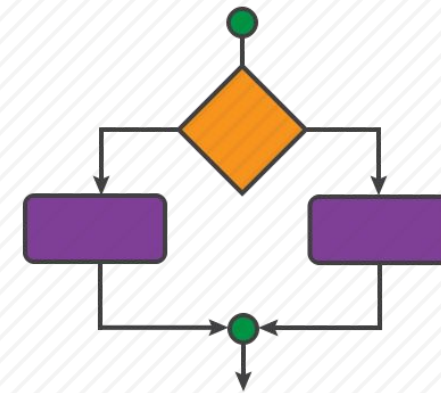
E' buona norma scrivere il nome delle variabili attraverso la CamelCase e in grassetto i nomi delle costanti in modo da rendere leggibile il codice:

MyVariable
BLUR_KERNEL_SIZE

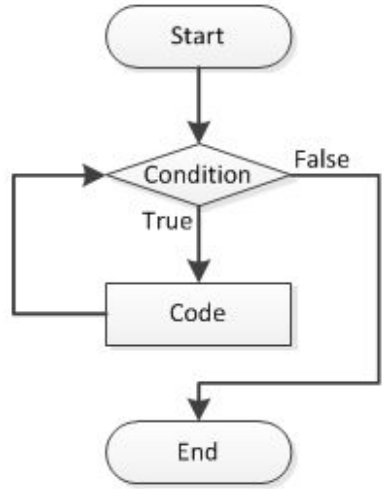


```
[INFO] Flag basso
[INFO] Flag basso
[INFO] Flag basso
[INFO] Flag basso
[INFO] Flag basso
[INFO] Flag alto
[INFO] Flag alto
[INFO] Flag basso
[INFO] Flag basso
[INFO] Flag basso
[INFO] Flag value 0
```

If-else



Python: For loop (I)



La funzione di default per i cicli for è **range(n)**, in forma estesa scritta **range(0, n, 1)** dove 0 è l'indice iniziale, n è l'indice finale e 1 è il passo.

N.B. In Python il primo elemento ha come indice 0 e non 1 come in Matlab.

Posso fare un ciclo for non solo sugli indici, ma anche sugli elementi presenti in una lista; *la lista è un contenitore in cui posso inserire oggetti e variabili di diverso tipo.*

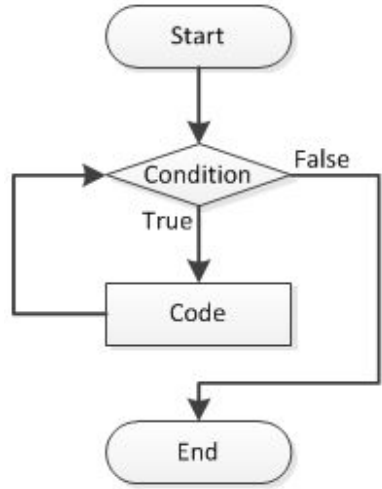
```
lista = ['Ciao', 'Hello', 'Hola', 10, 5.4]
for elemento in lista:
    print (elemento)
```

```
Ciao
Hello
Hola
10
5.4
```

Per accedere al valore dell'elemento i-esimo della lista uso `NameList[i]`

```
In [34]: lista[0]
Out[34]: 'Ciao'
```

Python: For loop (II)



La funzione di default per i cicli for è **range(n)**, in forma estesa scritta **range(0, n, 1)** dove 0 è l'indice iniziale, n è l'indice finale e 1 è il passo.

N.B. In Python il primo elemento ha come indice 0 e non 1 come in Matlab.

Posso fare un ciclo for non solo sugli indici, ma anche sugli elementi presenti in una lista; *la lista è un contenitore in cui posso inserire oggetti e variabili di diverso tipo.*

```
#Nested Loop

lista2 = ['Nel nested loop printo prima tutti i valori di lista2', \
        'poi ritorno in lista', True]

for x in lista:
    for y in lista2:
        print(x, ' ', y)

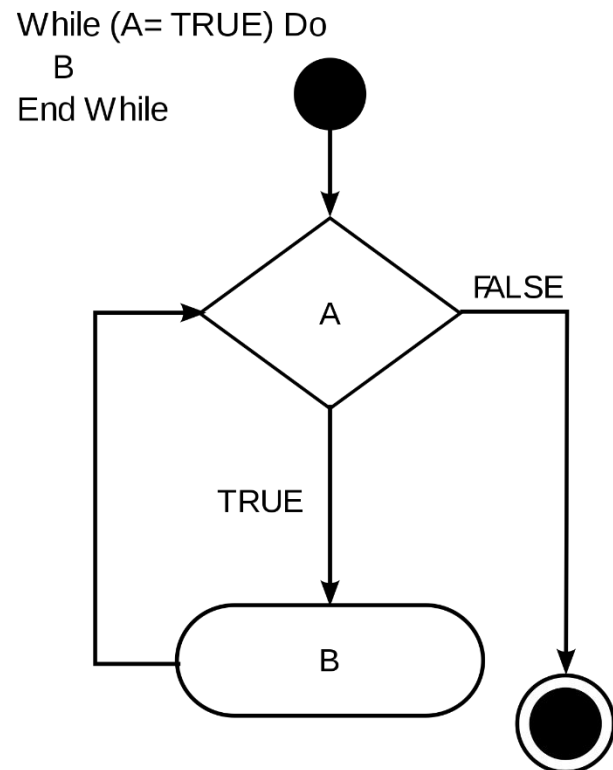
s = len(lista) #conta numero elementi di una lista
print('Il numero di elementi in lisata è ', s)
```



```
Ciao    Nel nested loop printo prima tutti i valori di lista2
Ciao    poi ritorno in lista
Ciao    True
Hello   Nel nested loop printo prima tutti i valori di lista2
Hello   poi ritorno in lista
Hello   True
Hola    Nel nested loop printo prima tutti i valori di lista2
Hola    poi ritorno in lista
Hola    True
10      Nel nested loop printo prima tutti i valori di lista2
10      poi ritorno in lista
10      True
5.4     Nel nested loop printo prima tutti i valori di lista2
5.4     poi ritorno in lista
5.4     True
Il numero di elementi in lisata è  5
```

Python: While loop

Viene utilizzato quando il numero di iterazioni non sono definite. All'interno di codici RT ed embedded nel main è infatti presente il costrutto **while = true**, che rappresenta un ciclo infinito.



```
while s != 0:
    print ('Ok')
    s = s - 1
```

```
signal = True
count = 0
THR = 8
while signal == True:
    print('Output presente')
    count = count + 1
    if count < THR and signal == True:
        print('Il segnale è alto')
    else:
        print('Il segnale è basso')
        signal = False
```

```
def fun1():
    s = 0
    return s

def fun2():
    s = 4
    return s

def main():
    while True:
        i = fun1()
        b = fun2()
        print(i+b)

if __name__ == "__main__":
    main()
```

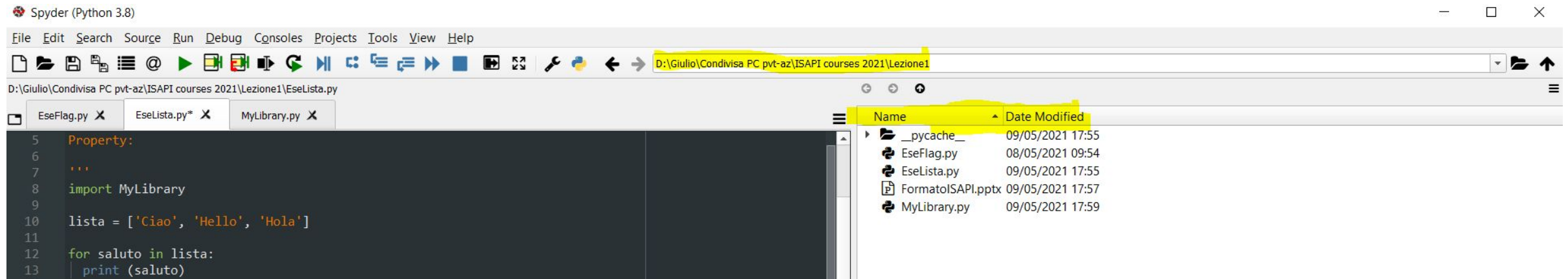
4
4
4
4
4
4
4
4
4
4
4
4
4

Python: Definizione delle funzioni

```
def function_name (parameters):  
    ↔ function body
```

Le funzioni vengono raccolte solitamente in librerie o header, altri file py da cui possono essere richiamate. Possono essere vuote o restituire un risultato con l'opzione **return**.

Il file header che contiene le funzioni deve essere contenuto nella stessa cartella Progetto dello script che si scrive, altrimenti c'è bisogno di funzioni aggiuntive dalla libreria OS per richiamare file in cartelle fisse.



Python: Importare le librerie

Import «name of library»

La libreria è un file py in cui sono definite delle funzioni o delle costanti o oggetti. Per importare le costanti e le funzioni basta puntare a questi una volta importata la libreria:

MyLibrary.PI -> call constat

MyLibrary.StatFeatures() -> call function

```
#Importa le librerie
import cv2
import numpy as np
import scipy as scp
import sklearn as sk
import matplotlib.pyplot as plt

#Definizione costanti
PI = 3.14
GRAVITY = 9.81
NEPER = 2.71

#Funzioni
def MyFun(array, THR):
    CONT = 0

    for i in range(array.shape[0]):
        for j in range(array.shape[1]):
            if array[i][j] > THR:
                CONT = CONT + 1

    return CONT

def StatFeatures(Data):
    #Data format (n, [features])7
    mu = []
    var = []

    for i in range(Data.shape[0]):
        mu = Data[i].mean()
        var = Data[i].var()

    return mu, var
```

```
In [41]: import MyLibrary
```

```
In [42]: MyLibrary
```

```
Out[42]: <module 'MyLibrary' from 'D:\\Giulio\\Condivisa PC pvt-az\\ISAPI courses 2021\\
\\Lezione1\\MyLibrary.py'>
```

```
In [43]: MyLibrary.PI
```

```
Out[43]: 3.14
```

```
mean, variance = MyLibrary.StatFeatures(img) #prendo tutte e due le uscite
mean, _ = MyLibrary.StatFeatures(img) #prendo soltanto mean (la prima)
```


Python: Gli Array Numpy

Gli array sono oggetti gestiti dalla libreria numpy e possono essere n-dimensionali. Questo significa che gli array portano al loro interno funzioni che si possono chiamare.



nameArray.function [oggetto.funzione_dell_oggetto(input)] -> OOP

```
''' Create an array '''  
A=numpy.array(['element1','element2',....])  
A=numpy.zeros('dimension')  
A=numpy.ones('dimension')  
A=numpy.zeros_like('another array')  
  
A.shape['channel']  
M=numpy.array([('row1'),('row2'),...,'(rowN)'])  
np.dot('arr1','arr2')  
A*B #elementwise
```

```
#prima riga e prima colonna di un array a 2D  
row = imgM[0, :]  
col = imgM[:, 0]
```

- **nomeArray.shape** returns the dimensions; for example a color image returns w, h , channel
- **nomeArray.mean()** returns the average of the array
- **nomeArray.var()** the variance assuming in both cases a normal distribution
- returns the number of elements
nomeArray.size() present in the array

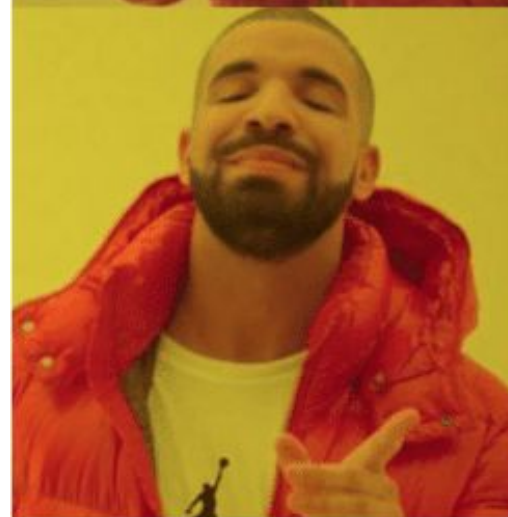
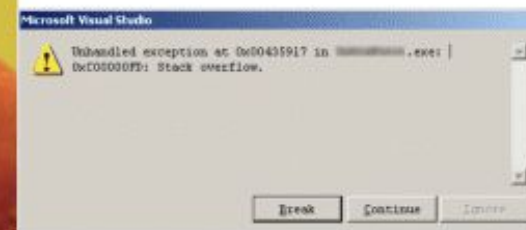
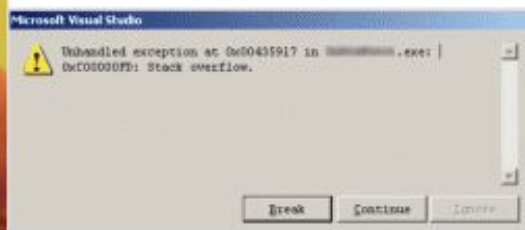
Python: Importare i dati con pandas



In Pandas I file excel possono essere importati e manipolati. Una volta importati diventano un'oggetto **dataFrame**.

- **df=pandas.read_excel();** carica il file excel e assegna a variabile
- **df.to_numpy();** trasforma in numpy array il dataset
- **df.describe();** Ottieni tutte le statistiche del dataset (media, min, max, std...)
- **df["A"]** accedi alla struttura A del dataset (se è una struttura)
- **df[0:3]** accedi ai primi 4 elementi

https://pandas.pydata.org/pandas-docs/stable/user_guide/10min.html







Tutti Video Shopping Immagini Notizie Altro Impostazioni Strumenti

Circa 70.500.000 risultati (0,54 secondi)

To **unzip** it first create a ZipFile object by opening the zip **file** in read mode and then call `extractall()` on that object i.e. It will **extract** all the **files** in zip at current **Directory**. If **files** with same name are already present at **extraction** location then it will overwrite those **files**. 1 dic 2018

<https://thispointer.com/python-how-to-unzip-a-file-extra...>

Python: How to unzip a file | Extract Single, multiple or all files ...

Informazioni sugli snippet in primo piano Feedback

<https://stackoverflow.com/extra...> Traduci questa pagina

Extract all files from multiple folders with Python - Stack Overflow



2 risposte

11 nov 2019 — This should be what you are looking for, first we get all subfolders in our main **folder**. Then for each subfolder we get **files** contained inside and ...

Unzip all zipped **files** in a **folder** to that same **folder** ... 4 risposte 2 lug 2019

How can I **extract** the **folder path** from **file path** in ... 6 risposte 12 giu 2013

Extract file name from **path**, no matter what the **os/path** ... 19 risposte 5 dic 2011

How to **extract data** from multiple **files** with **Python** ... 1 risposta 24 feb 2014

Altri risultati in stackoverflow.com

2 Answers

Active Oldest Votes



5



This should be what you are looking for, first we get all subfolders in our main folder. Then for each subfolder we get files contained inside and create our source path and destination path for `shutil.move`.

```
import os
import shutil

folder = r"<MAIN FOLDER>"
subfolders = [f.path for f in os.scandir(folder) if f.is_dir()]

for sub in subfolders:
    for f in os.listdir(sub):
        src = os.path.join(sub, f)
        dst = os.path.join(folder, f)
        shutil.move(src, dst)
```



Demo del corso



DI
C
Ma
PI

Dipartimento
di Ingegneria Chimica,
dei Materiali e della
Produzione Industriale
Università degli Studi
di Napoli Federico II

Speaker: Eng. Giulio Mattera

mail: giulio.mattera@unina.it



DI
C
Ma
PI
Dipartimento
di Ingegneria Chimica,
dei Materiali e della
Produzione Industriale
Università degli Studi
di Napoli Federico II

Esercitazioni Python

- **Organizzazione di un progetto e sviluppo applicazione python in Ubuntu**
- La libreria Open-CV: Funzioni principali, convoluzione e filtri
- Le capacità di generalizzazione delle reti neurali: introduzione a Tensorflow e la funzione seno
- Run or walk detector
- Predittore di difetti in saldatura
- Sviluppo di una CNN per classificazione dei fiori (Transfer learning in tensorflow)
- Homework

Esercizi per casa



DI
C
Ma
PI

Dipartimento
di Ingegneria Chimica,
dei Materiali e della
Produzione Industriale
Università degli Studi
di Napoli Federico II

Speaker: Eng. Giulio Mattera

mail: giulio.mattera@unina.it



DI
C
Ma
PI
Dipartimento
di Ingegneria Chimica,
dei Materiali e della
Produzione Industriale
Università degli Studi
di Napoli Federico II

Esercizio 1

- Se ENABLE è alto:
 - Se INPUT1 è maggiore di THR
 - $OUTPUT1 = INPUT1 - INPUT2$
 - Altrimenti:
 - $OUTPUT1 = INPUT2 - INPUT1$
- Altrimenti
 - $OUTPUT1 = 0$

Esercizio 3

Scrivere una funzione chiamata **esponente(base, exp)** che restituisce un valore **int** di base elevato alla potenza di exp.

Esercizio 2

- Fintanto che s è alto
 - aggiungi un elemento random alla lista A
 - Se la somma di tutti gli elementi di A è maggiore di THR
 - s è alto
 - Altrimenti
 - s è basso

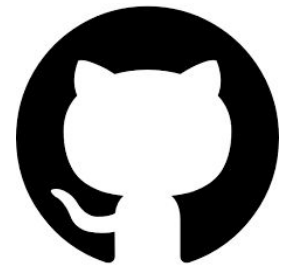
Esercizio 4

Utilizzare il dataset **Automobile_data.csv** e trovare:

- Il prezzo più alto di ogni casa automobilistica
- Il valore medio (m) e la varianza (v) del wheelbase

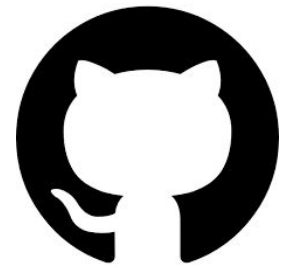
Plottare la gaussiana associata ad **m** , **v**

GitHub (I)



- At a high level, GitHub is a website and cloud-based service that helps developers store and manage their code, as well as track and control changes to their code
 - With **branching**, a developer duplicates part of the source code (called the **repository**). The developer can then safely make changes to that part of the code without affecting the rest of the project.
 - Then, once the developer gets his or her part of the code working properly, he or she can **merge** that code back into the main source code to make it official.
- Git is a **specific open-source version control system** created by Linus Torvalds in 2005. Specifically, Git is a **distributed version control system**, which means that the entire codebase and history is available on every developer's computer, which allows for easy branching and merging. According to a [Stack Overflow developer survey](https://stackoverflow.com/survey/2016), over 87% of developers use Git.

GitHub (II)



main [Cartpole-DDPG-agent-ros-bridge-for-simulink](#) / [rl_connection](#) / [src](#) / **agent_node.py** / <> Jump to ▾

Go to file

...



giuliomattera Add files via upload

Latest commit 040aa74 3 days ago

History

1 contributor

```
@@ -211,9 +211,10 @@ def exp_decay(epoch, ini_value, decay):
211 211+         actor_model.load_weights('./checkpoints/actor')
212 212         critic_model.load_weights('./checkpoints/critic')
213 213         best = np.load('./checkpoints/last_td.npy')
214 214+         print(' Last best : ', best)
215 215     else:
216 216         print('[INFO] Initializing agent s weights...')
217 217+         best = 1000000
218 218
219 219         critic_lr = config.CRITIC_LR
220 220         actor_lr = config.ACTOR_LR
@@ -231,7 +232,7 @@ def exp_decay(epoch, ini_value, decay):
231 232         action = Float32()
232 233         done = False
233 234         TS = config.TIME_STEP
234 234-         i, episode, t, started, j= 0 , 0, 0, 0, 0
235 235+         i, episode, t, started, j, EARLY= 0 , 0, 0, 0, 0
236 236         states, actions, traj, total_trajectory, = [], [], [], []
237 237
238 238         BATCH = config.BATCH_SIZE
```

Add new features

- Saving checkpoint based on TD error
- Saving the last best TD error
- Add exp learning rate decay for actor and critic
- Add exp decay for action space searching

main

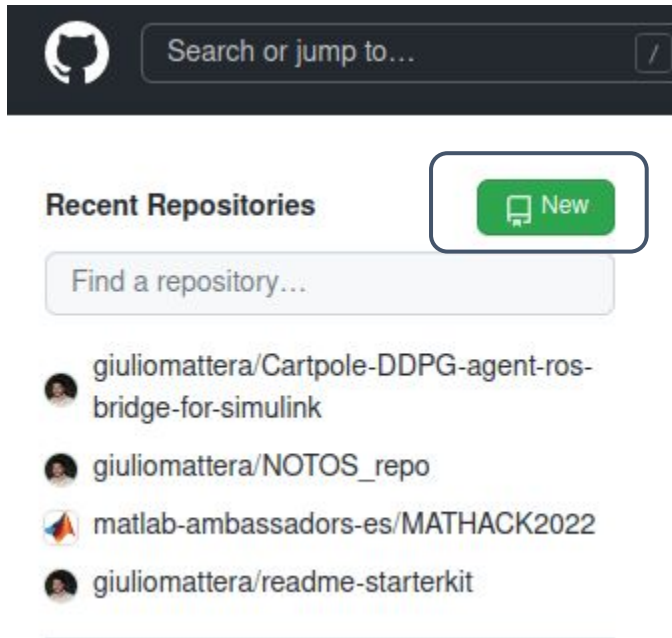
giuliomattera committed 7 days ago Verified



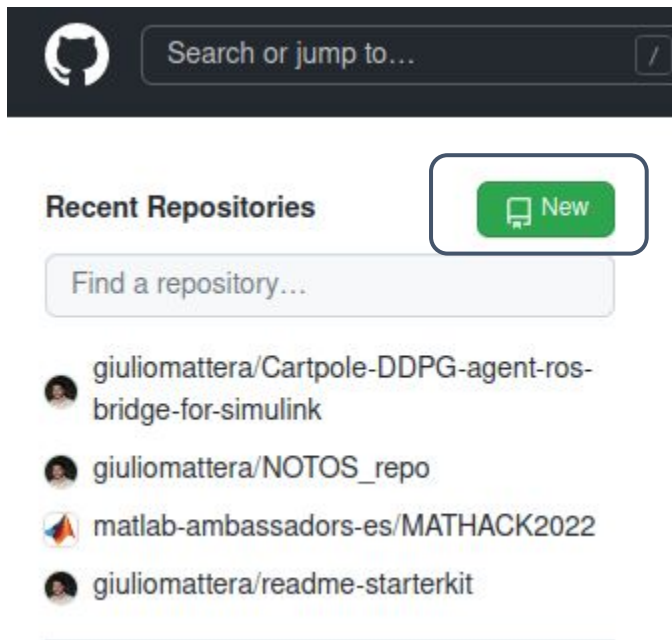
DI
C
Ma
PI
Dipartimento
di Ingegneria Chimica,
dei Materiali e della
Produzione Industriale
Università degli Studi
di Napoli Federico II

<https://kinsta.com/knowledgebase/what-is-github/>

Guideline per assessment



Guideline per assessment



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *

 giuliomattera ▾

Repository name *

/ ISAPI2022 ✓

Great repository names are short and memorable. Need inspiration? How about **didactic-robot**?

Description (optional)

☒  **Public**

Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☒ **Add a README file**

This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**

Choose which files not to track from a list of templates. [Learn more.](#)

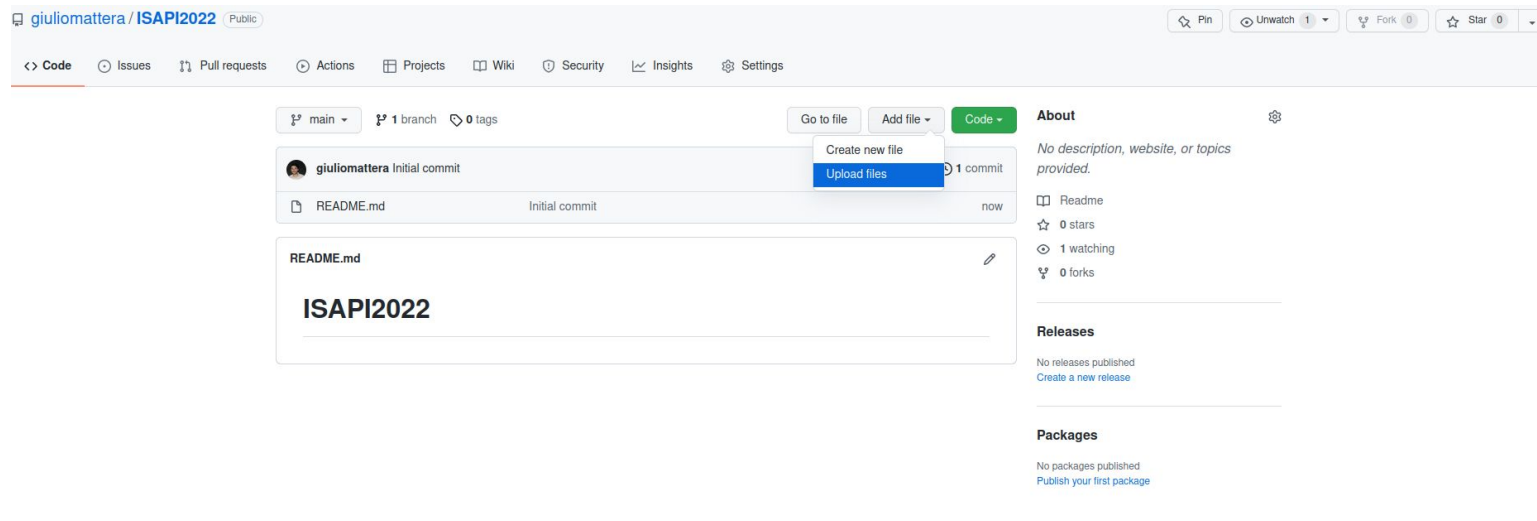
☐ **Choose a license**

A license tells others what they can and can't do with your code. [Learn more.](#)

This will set  **main** as the default branch. Change the default name in your [settings](#).


Create repository

Guideline per assessment




Guideline per assessment

Who has access

PUBLIC REPOSITORY 


This repository is public and visible to anyone.

[Manage](#)

DIRECT ACCESS 

0 collaborators have access to this repository. Only you can contribute to this repository.

Manage access




You haven't invited any collaborators yet

[Add people](#)


Guideline per assessment

Who has access

PUBLIC REPOSITORY 

This repository is public and visible to anyone.


[Manage](#)

DIRECT ACCESS 

0 collaborators have access to this repository. Only you can contribute to this repository.

- Once invited to collaborate, create your own folder
- Load all _prj folders in yours space

Manage access



You haven't invited any collaborators yet

[Add people](#)