Programmazione con Python

Introduzione e fondamenti del linguaggio

Leonardo Essam Dei Rossi

ITT "M. Buonarroti" - Trento (TN)

Anno scolastico 2025/2026

Licenze e crediti

Questo materiale è disponibile sul sito Web del docente per il corso di Tecnologie informatiche per le studentesse e gli studenti dell'anno scolastico 2025/2026.

Versione: `1.7.3 (A)`
Ultima modifica: 10/04/2026 20:41
Riferimenti: [1, Unità P1 - La Programmazione]

Indice dei contenuti

Indice dei contenuti

Indice dei contenuti

Il linguaggio Python

- All'inizio degli anni '90 Guido van Rossum progettò un nuovo linguaggio di programmazione con l'obiettivo di avere una sintassi più semplice e "alla portata" rispetto ad altri linguaggi dell'epoca come C e C++;
- Van Rossum non era soddisfatto dei linguaggi esistenti:
  - Erano ottimizzati per scrivere grandi programmi, eseguibili in modo efficiente;

Il linguaggio Python

- All'inizio degli anni '90 Guido van Rossum progettò un nuovo linguaggio di programmazione con l'obiettivo di avere una sintassi più semplice e "alla portata" rispetto ad altri linguaggi dell'epoca come C e C++;

- Van Rossum non era soddisfatto dei linguaggi esistenti:
  - Erano ottimizzati per scrivere grandi programmi, eseguibili in modo efficiente;

Il linguaggio Python

- Voleva un linguaggio che permettesse di creare rapidamente i programmi, ma anche modificarli in modo semplice:
  - L'ambiente Python aveva un approccio "batterie comprese", offrendo subito la disponibilità di molte funzioni utili in modo standard;
  - Python è interpretato, rendendo più facile lo sviluppo ed il test di brevi programmi.

- I programmi Python sono eseguiti dall'interprete Python:
  - L'interprete legge il programma e lo esegue.

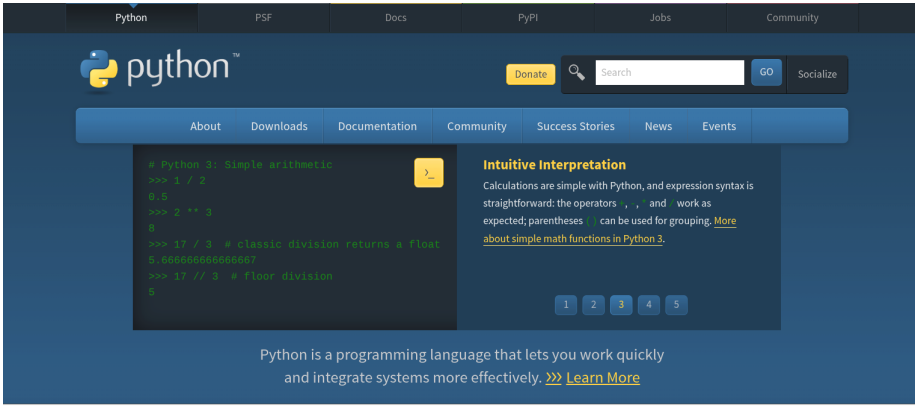
Il linguaggio Python

- Voleva un linguaggio che permettesse di creare rapidamente i programmi, ma anche modificarli in modo semplice:
  - L'ambiente Python aveva un approccio "batterie comprese", offrendo subito la disponibilità di molte funzioni utili in modo standard;
  - Python è interpretato, rendendo più facile lo sviluppo ed il test di brevi programmi.

- I programmi Python sono eseguiti dall'interprete Python:
  - L'interprete legge il programma e lo esegue.

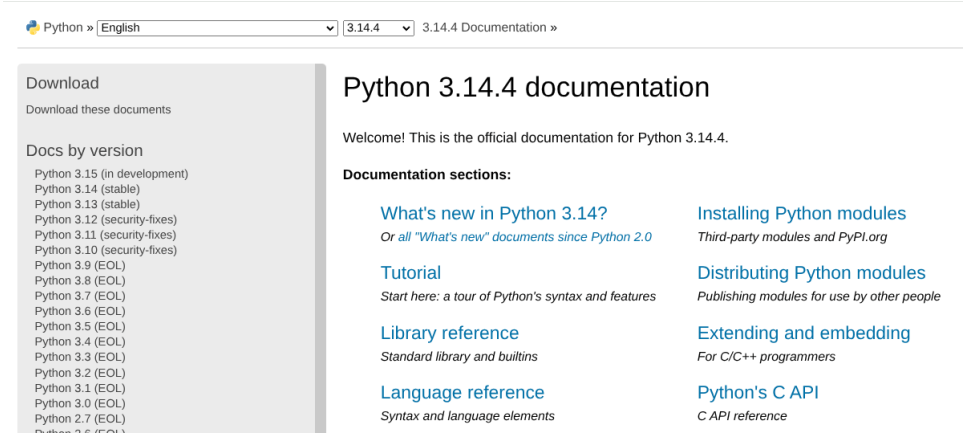
Link utili

Sito Web principale: https://www.python.org/



Link utili

Sito Web della documentazione: https://docs.python.org/



Link utili

Altri link utili:

- https://realpython.com/
  - Molti tutorial a diversi livelli di approfondimento
- https://devdocs.io/python/
  - Guida alle funzioni della libreria standard ed elenco dei moduli disponibili

Indice dei contenuti

Ambienti di programmazione

Ci sono vari modi per creare un programma:

- Usando un sistema integrato di sviluppo[1]:
  - IDLE, PyCharm, Visual Studio Code, ...
- Usando un semplice editor di testi:
  - Blocco note, Notepad++, Atom, vi, gedit, ...

Nel corso useremo l'IDE Visual Studio Code (link).

[1] IDE, in inglese: "*Integrated Development Environment*"

Ambienti di programmazione

Ci sono vari modi per creare un programma:

- Usando un sistema integrato di sviluppo[1]:
  - IDLE, PyCharm, Visual Studio Code, ...
- Usando un semplice editor di testi:
  - Blocco note, Notepad++, Atom, vi, gedit, ...

Nel corso useremo l'IDE Visual Studio Code (link).

[1] IDE, in inglese: "*Integrated Development Environment*"

Ambienti di programmazione

Ci sono vari modi per creare un programma:

- Usando un sistema integrato di sviluppo[1]:
  - IDLE, PyCharm, Visual Studio Code, ...
- Usando un semplice editor di testi:
  - Blocco note, Notepad++, Atom, vi, gedit, ...

Nel corso useremo l'IDE Visual Studio Code (link).

[1] IDE, in inglese: "*Integrated Development Environment*"

Componenti di un IDE

- L'editor del codice sorgente aiuta il programmatore con:
  - Visualizzazione dei numeri di linea del codice;
  - Evidenziazione e colorazione della sintassi (commenti, testi, ...);
  - Indentazione automatica del codice;
  - Evidenziazione degli errori di sintassi;
  - Completamento automatico dei nomi.
- Finestra di output:
  - L'output (testuale) generato dal programma.
- Debugger:
  - Strumenti di ausilio alla ricerca degli errori logici nel programma.

Componenti di un IDE

- L'editor del codice sorgente aiuta il programmatore con:
  - Visualizzazione dei numeri di linea del codice;
  - Evidenziazione e colorazione della sintassi (commenti, testi, ...);
  - Indentazione automatica del codice;
  - Evidenziazione degli errori di sintassi;
  - Completamento automatico dei nomi.

- Finestra di output:
  - L'output (testuale) generato dal programma.

- Debugger:
  - Strumenti di ausilio alla ricerca degli errori logici nel programma.

Componenti di un IDE

- L'editor del codice sorgente aiuta il programmatore con:
  - Visualizzazione dei numeri di linea del codice;
  - Evidenziazione e colorazione della sintassi (commenti, testi, ...);
  - Indentazione automatica del codice;
  - Evidenziazione degli errori di sintassi;
  - Completamento automatico dei nomi.
- Finestra di output:
  - L'output (testuale) generato dal programma.
- Debugger:
  - Strumenti di ausilio alla ricerca degli errori logici nel programma.

Programmazione con un editor di testi

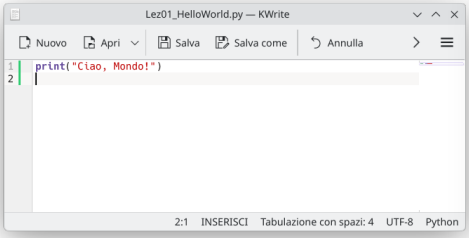
- Si può anche usare un semplice editor di testi per scrivere il codice...
  - ... ma è come scrivere un tema su un rotolo di carta igienica!
- Salvando il file come hello.py, usare una finestra di comando per:
  - Compilare & eseguire il programma.

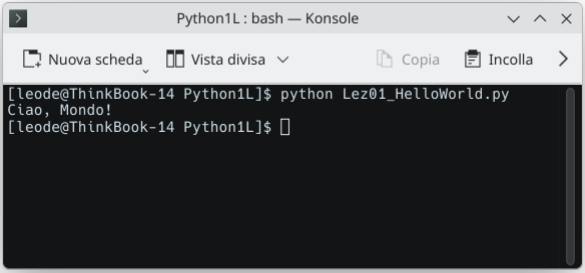
Programmazione con un editor di testi

- Si può anche usare un semplice editor di testi per scrivere il codice...
  - ... ma è come scrivere un tema su un rotolo di carta igienica!

- Salvando il file come hello.py, usare una finestra di comando per:
  - Compilare & eseguire il programma.

Programmazione con un editor di testi





Indice dei contenuti

Il nostro primo programma

- Il classico programma "Ciao, Mondo!" in Python:
  - `print` è un esempio di una istruzione (*statement*) in Python.

Esempio 1: il primo programma

```
# Il mio primo programma
print("Ciao, Mondo!")
```

Scrivere un programma in Python

- Attenzione agli errori di battitura:
  - Esempio: `'print'` vs. `'primpt'`
- PyTHon fA diFFereNza tra MAIUscole e minuSCOLE;
- Gli spazi sono importanti, soprattutto all'inizio della linea (indentazione o rientro);
- Le linee che iniziano con # sono commenti (vengono ignorati da Python).

Scrivere un programma in Python

- Attenzione agli errori di battitura:
  - Esempio: `'print'` vs. `'primpt'`
- PyTHon fA diFFereNza tra MAIUscole e minuSCOLE;
- Gli spazi sono importanti, soprattutto all'inizio della linea (indentazione o rientro);
- Le linee che iniziano con # sono commenti (vengono ignorati da Python).

Scrivere un programma in Python

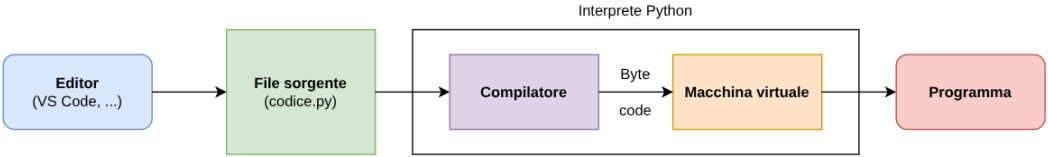
- Attenzione agli errori di battitura:
  - Esempio: `'print'` vs. `'primpt'`
- PyTHon fA diFFereNza tra MAIUscole e minuSCOLE;
- Gli spazi sono importanti, soprattutto all'inizio della linea (indentazione o rientro);
- Le linee che iniziano con # sono commenti (vengono ignorati da Python).

Scrivere un programma in Python

- Attenzione agli errori di battitura:
  - Esempio: `'print'` vs. `'primpt'`
- PyTHon fA diFFereNza tra MAIUscole e minuSCOLE;
- Gli spazi sono importanti, soprattutto all'inizio della linea (indentazione o rientro);
- Le linee che iniziano con # sono commenti (vengono ignorati da Python).

Dal sorgente all'esecuzione del programma

- Il compilatore legge il programma e genera le istruzioni binarie (in inglese: *bytecode*, semplici istruzioni per la Macchina Virtuale Python):
  - La Macchina Virtuale Python è un programma che si comporta come la CPU del computer (esegue istruzioni, in software);
  - Ogni libreria necessaria (es. per la grafica) viene automaticamente trovata ed inclusa dalla macchina virtuale.



Indice dei contenuti

L'IDE di Visual Studio Code

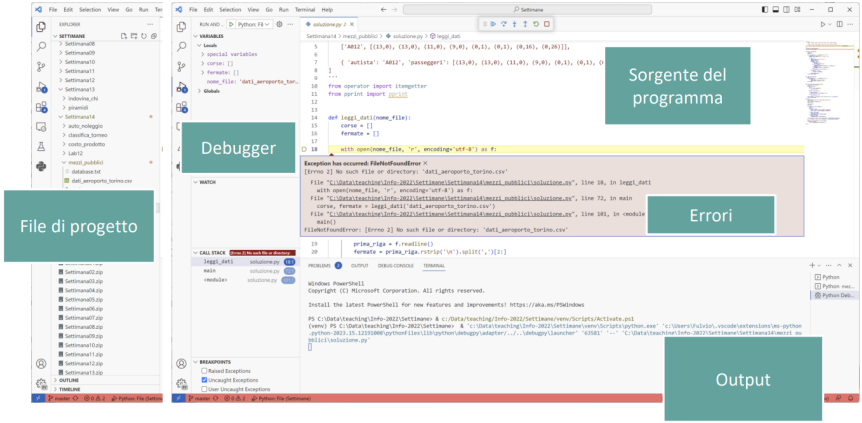


Figura 1: Immagine del prof. Fulvio Corno (Politecnico di Torino)

Progetto vs. Programma vs. File

- Un singolo Programma potrebbe essere molto grande, in tal caso sarà composto da molti File diversi;
- Gli IDE permettono di raggruppare un insieme di File correlati in un "Progetto":
  - In Visual Studio Code, un progetto corrisponde ad una cartella.
- Ogni volta che vogliamo creare un nuovo Programma, dovremo:
  - Creare un nuovo Progetto (con la relativa cartella);
  - Creare uno (o più) File Python (con estensione `.py`) all'interno del Progetto.

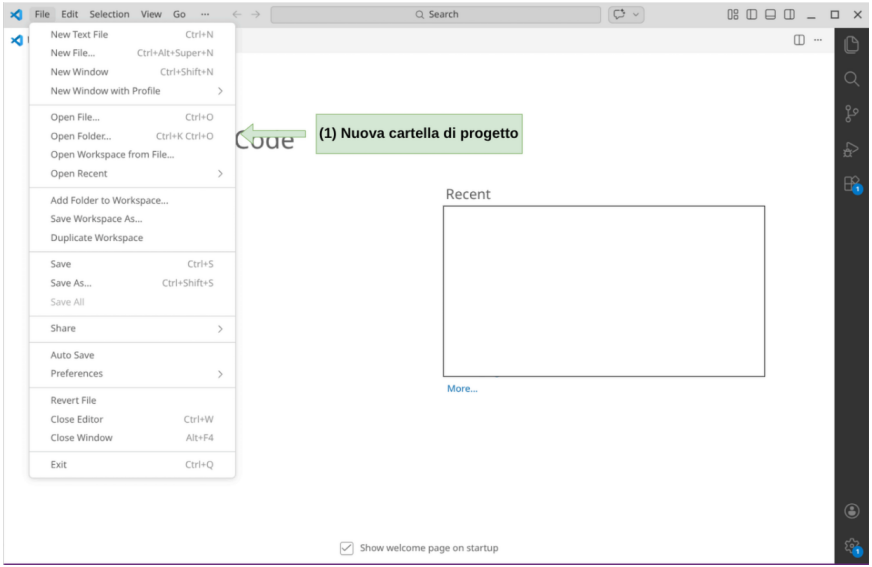
Progetto vs. Programma vs. File

- Un singolo Programma potrebbe essere molto grande, in tal caso sarà composto da molti File diversi;
- Gli IDE permettono di raggruppare un insieme di File correlati in un "Progetto":
  - In Visual Studio Code, un progetto corrisponde ad una cartella.
- Ogni volta che vogliamo creare un nuovo Programma, dovremo:
  - Creare un nuovo Progetto (con la relativa cartella);
  - Creare uno (o più) File Python (con estensione `.py`) all'interno del Progetto.

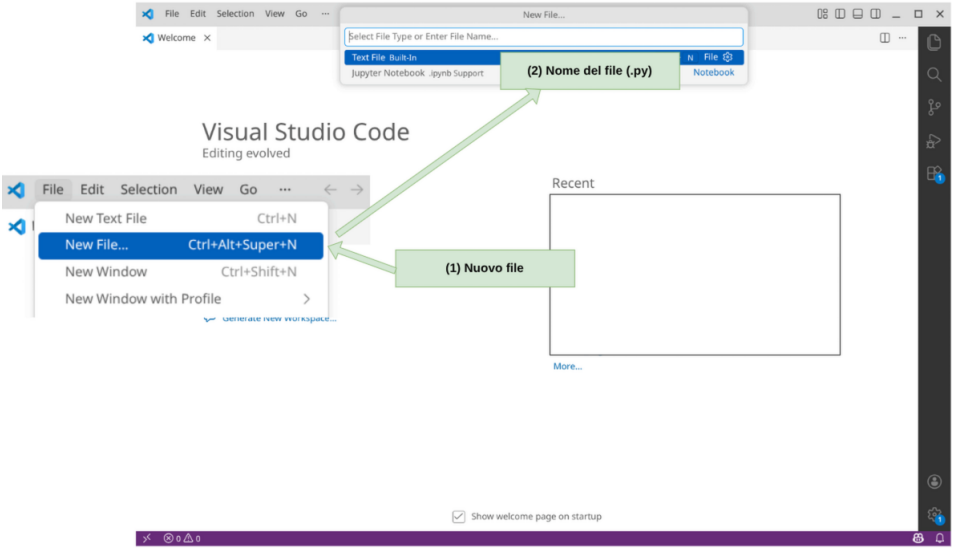
Progetto vs. Programma vs. File

- Un singolo Programma potrebbe essere molto grande, in tal caso sarà composto da molti File diversi;
- Gli IDE permettono di raggruppare un insieme di File correlati in un "Progetto":
  - In Visual Studio Code, un progetto corrisponde ad una cartella.
- Ogni volta che vogliamo creare un nuovo Programma, dovremo:
  - Creare un nuovo Progetto (con la relativa cartella);
  - Creare uno (o più) File Python (con estensione `.py`) all'interno del Progetto.

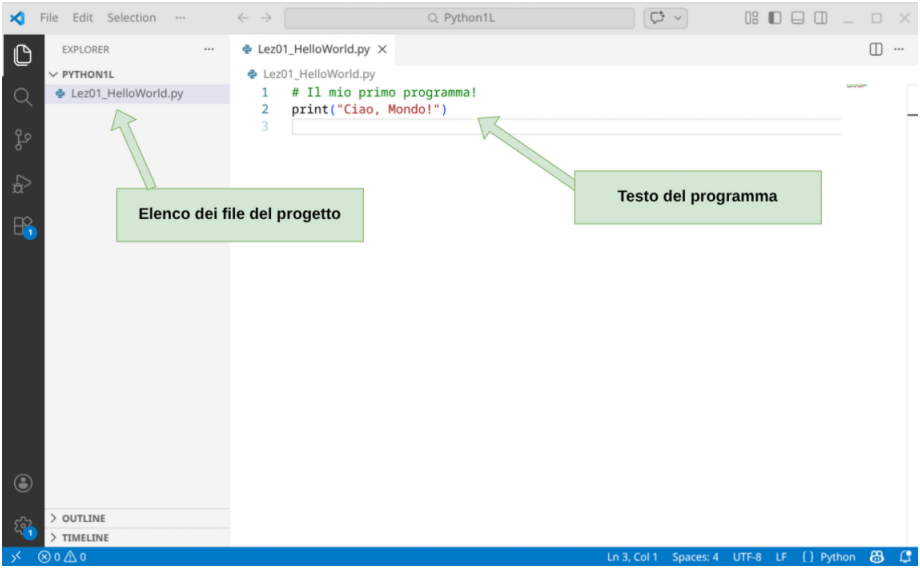
Creare un nuovo progetto



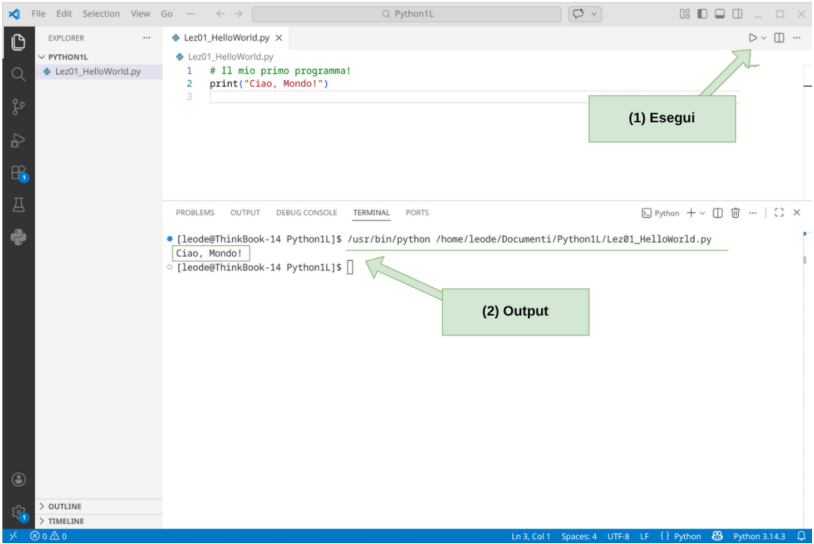
Creare un nuovo progetto



Creare un nuovo progetto



Creare un nuovo progetto



Creare un nuovo progetto

- Il codice sorgente è salvato nel file `.py`;
- Creiamo una cartella per il corso di Tecnologie informatiche;
- Creiamo una cartella di progetto per ciascun programma:
  - Un programma sarà composto da uno o più file `.py`
- Fare backup regolari e frequenti dei propri dati:
  - Su chiavetta USB (o più di una);
  - Su un disco di rete (servizio cloud) or hard disk esterno;
  - Fatelo. Davvero. Sempre.

Creare un nuovo progetto

- Il codice sorgente è salvato nel file `.py`;
- Creiamo una cartella per il corso di Tecnologie informatiche;
- Creiamo una cartella di progetto per ciascun programma:
  - Un programma sarà composto da uno o più file `.py`
- Fare backup regolari e frequenti dei propri dati:
  - Su chiavetta USB (o più di una);
  - Su un disco di rete (servizio cloud) or hard disk esterno;
  - Fatelo. Davvero. Sempre.

Creare un nuovo progetto

- Il codice sorgente è salvato nel file `.py`;
- Creiamo una cartella per il corso di Tecnologie informatiche;
- Creiamo una cartella di progetto per ciascun programma:
  - Un programma sarà composto da uno o più file `.py`
- Fare backup regolari e frequenti dei propri dati:
  - Su chiavetta USB (o più di una);
  - Su un disco di rete (servizio cloud) or hard disk esterno;
  - Fatelo. Davvero. Sempre.

Creare un nuovo progetto

- Il codice sorgente è salvato nel file `.py`;
- Creiamo una cartella per il corso di Tecnologie informatiche;
- Creiamo una cartella di progetto per ciascun programma:
  - Un programma sarà composto da uno o più file `.py`
- Fare backup regolari e frequenti dei propri dati:
  - Su chiavetta USB (o più di una);
  - Su un disco di rete (servizio cloud) or hard disk esterno;
  - Fatelo. Davvero. Sempre.

Indice dei contenuti

Modalità interattiva di Python

- L'interprete di Python normalmente carica un intero programma ed esegue le istruzioni in esso contenute:
  - Procedimento simile ad altri linguaggi (compilati).

- In alternativa: in modo interattivo, Python può eseguire un'istruzione per volta:
  - Permette di scrivere velocemente dei "programmini di test";
  - Permette di provare e sperimentare con le varie istruzioni;
  - Permette di scrivere istruzioni Python direttamente nella finestra di console.

Modalità interattiva di Python

- L'interprete di Python normalmente carica un intero programma ed esegue le istruzioni in esso contenute:
  - Procedimento simile ad altri linguaggi (compilati).

- In alternativa: in modo interattivo, Python può eseguire un'istruzione per volta:
  - Permette di scrivere velocemente dei "programmini di test";
  - Permette di provare e sperimentare con le varie istruzioni;
  - Permette di scrivere istruzioni Python direttamente nella finestra di console.

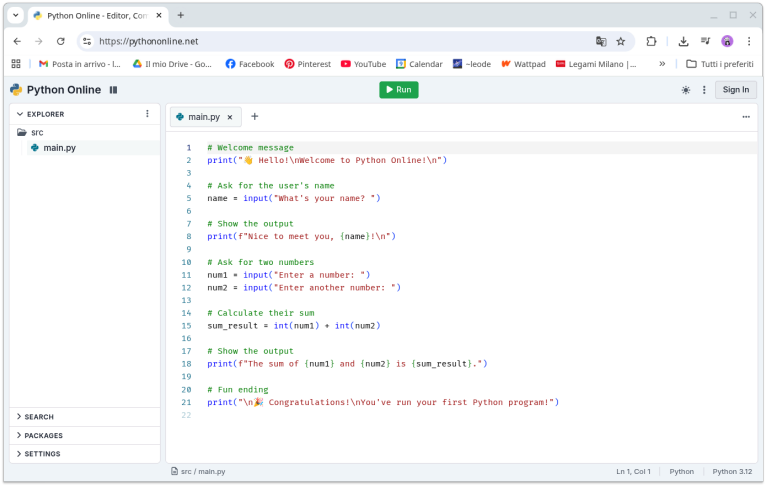
Modalità interattiva di Python

```
[leode@ThinkBook-14 Python1L]$ python Lez01_HelloWorld.py
Ciao, Mondo!
[leode@ThinkBook-14 Python1L]$ python
Python 3.14.3 (main, Feb 13 2026, 15:31:44) [GCC 15.2.1 20260209] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Ciao, Mondo!")
Ciao, Mondo!
>>> exit()
```

Modalità interattiva

IDE on-line (pythononline.net)

- Link: https://pythononline.net/



Indice dei contenuti

La funzione print()

- Usare la funzione `print()` in Python:
  - Una funzione è un insieme di istruzioni (con un nome) che svolge un compito (task) particolare (in questo caso, stampare un valore su schermo);
  - È codice che qualcun altro ha scritto per noi!

Definizione 1: La funzione print()

Sintassi:

```
print()
print(value1, value2, ...)
```

La funzione presenta come parametro opzionale un insieme di valori, se non viene stampato nessun valore verrà stampata una linea vuota. Altrimenti, i valori verrano stampati uno dopo l'altro separati da uno spazio.

Sintassi per le funzioni Python

- Per usare (o "chiamare") una funzione in Python, occorre specificare:
  - Il nome della funzione che vogliamo usare:
    - ⋆ Nell'esempio precedente, il nome era `print`.
  - Tutti i valori (argomenti, parametri) di cui la funzione ha bisogno per svolgere il proprio compito:
    - ⋆ In questo caso: `"Ciao, Mondo!"`
  - Gli argomenti sono racchiusi tra parentesi tonde;
  - Se vi sono più argomenti, sono separati da virgole.

Le stringhe

- Una sequenza di caratteri racchiusa tra apici o virgolette è chiamata Stringa:
  - Può essere racchiusa tra 'apici singoli';
  - Può essere racchiusa tra "apici doppi" o "virgolette".

Altri esempi della funzione print

- Stampare valori numerici:
  - `print(3 + 4)`
  - Valuta l'espressione $3 + 4$ e visualizza 7.

- Passare più valori alla funzione:
  - `print("Valore ottenuto ", 6 * 7)`
  - Visualizza: `Valore ottenuto 42`
  - Tutti i valori passati alla funzione vengono visualizzati, uno dopo l'altro, separati da uno spazio.

Altri esempi della funzione print

- Stampare valori numerici:
  - `print(3 + 4)`
  - Valuta l'espressione $3 + 4$ e visualizza 7.

- Passare più valori alla funzione:
  - `print("Valore ottenuto ", 6 * 7)`
  - Visualizza: `Valore ottenuto 42`
  - Tutti i valori passati alla funzione vengono visualizzati, uno dopo l'altro, separati da uno spazio.

Altri esempi della funzione print

- Per default, la funzione print crea una nuova linea (va ≪a capo≫) ogni volta che stampa i suoi argomenti:
  - `print("Ciao, ")`
  - `print("Mondo!")`
- Stampa due linee di testo:
  - `Ciao,`
  - `Mondo!`

Il nostro secondo programma

```
# Calcola e stampa 3 + 4 = 7
print(3 + 4)

# Stampa "Ciao, Mondo!" in due righe
print("Ciao, ")
print("Mondo!")

# Stampa valori multipli con un'unica chiamata a print
print("I miei numeri preferito sono ", 3 + 4, " e ", 3 + 10)

# Stampa tre righe di testo di cui una vuota
print("Arrivederci")
print()
print("Spero di rivederti presto!")
```

Indice dei contenuti

Errori di compilazione e di esecuzione

Errori a tempo di compilazione (quando si fa click su "esegui"):

- Scrittura, maiuscole, punteggiatura, ...
- Ordine delle istruzioni, corrispondenza delle parentesi, virgolette, indentazione, ...
- Il compilatore non crea alcun programma eseguibile;
- Correggere il primo errore evidenziato, poi ri-compilare:
  - Ripetere finché tutti gli errori non sono corretti.
- Solitamente rivelati ed evidenziati direttamente dall'IDE.

Errori di compilazione e di esecuzione

Errori a tempo di esecuzione (in inglese: *run-time*):

- Il programma viene eseguito, ma non produce il risultato corretto;
- Il programma può andare in "crash";
- Sono i più difficili da trovare e correggere...
  - ... anche per programmatori più esperti!

Errori di sintassi

- Gli errori di sintassi vengono catturati dal compilatore;

- Verificare cosa succede se:
  - Sbagliamo una maiuscola: `Print("Hello World!")`
  - Dimentichiamo le virgolette: `print(Hello World!)`
  - Virgolette non corrispondenti: `print("Hello World!')`
  - Parentesi non corrispondenti: `print('Hello'`

- Proviamo ciascun esempio nell'IDE:
  - Nel sorgente del programma;
  - Nella console Python interattiva.

Q: Quali messaggi di errore vengono generati?

Errori di sintassi

- Gli errori di sintassi vengono catturati dal compilatore;
- Verificare cosa succede se:
  - Sbagliamo una maiuscola: `Print("Hello World!")`
  - Dimentichiamo le virgolette: `print(Hello World!)`
  - Virgolette non corrispondenti: `print("Hello World!')`
  - Parentesi non corrispondenti: `print('Hello'`
- Proviamo ciascun esempio nell'IDE:
  - Nel sorgente del programma;
  - Nella console Python interattiva.

Q: Quali messaggi di errore vengono generati?

Errori di sintassi

- Gli errori di sintassi vengono catturati dal compilatore;
- Verificare cosa succede se:
  - Sbagliamo una maiuscola: `Print("Hello World!")`
  - Dimentichiamo le virgolette: `print(Hello World!)`
  - Virgolette non corrispondenti: `print("Hello World!')`
  - Parentesi non corrispondenti: `print('Hello'`
- Proviamo ciascun esempio nell'IDE:
  - Nel sorgente del programma;
  - Nella console Python interattiva.

Q: Quali messaggi di errore vengono generati?

Errori logici

- Verificare cosa succede se:
  - Dividiamo per zero: `print(1/0)`
  - Sbagliamo il testo: `print("Hello, Word!")`
- Il programma compila ≪normalmente≫ e viene eseguito:
  - L'output però non è quello che ci aspettiamo!
- Proviamo ciascun esempio nell'IDE:
  - Quali errori vengono generati?

Errori logici

- Verificare cosa succede se:
  - Dividiamo per zero: `print(1/0)`
  - Sbagliamo il testo: `print("Hello, Word!")`
- Il programma compila ≪normalmente≫ e viene eseguito:
  - L'output però non è quello che ci aspettiamo!
- Proviamo ciascun esempio nell'IDE:
  - Quali errori vengono generati?

Errori logici

- Verificare cosa succede se:
  - Dividiamo per zero: `print(1/0)`
  - Sbagliamo il testo: `print("Hello, Word!")`
- Il programma compila ≪normalmente≫ e viene eseguito:
  - L'output però non è quello che ci aspettiamo!
- Proviamo ciascun esempio nell'IDE:
  - Quali errori vengono generati?

Riferimenti e approfondimenti

[1] F. Corno, 14BHD - Informatica, Politecnico di Torino, 2025.