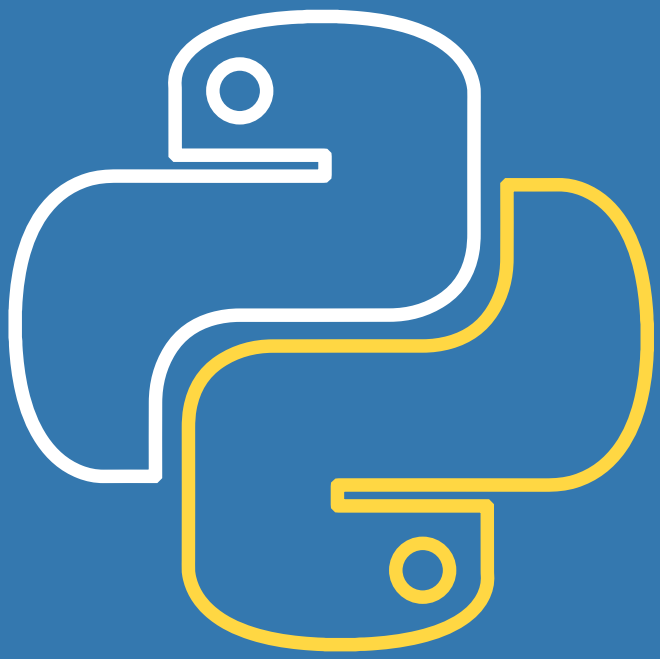
HOEPLI
TECNICA
PER LA SCUOLA

Quaderni di
TECNOLOGIE



Paolo Camagni
Riccardo Nikolassy

Python

Edizione **OPENSCHOOL**

1	**LIBRO**DITESTO
2	**E-BOOK+**
3	RISORSE**ONLINE**
4	**PIATTAFORMA**

HOEPLI

PAOLO CAMAGNI RICCARDO NIKOLASSY

PYTHON

Quaderni di tecnologie

EDITORE ULRICO HOEPLI MILANO

Via Hoepli 5, 20121 Milano (Italy)
tel. +39 02 864871 – fax +39 02 8052886
e-mail hoepli@hoepli.it

www.hoepli.it

Quaderni di tecnologie

La collana Quaderni di tecnologie è costituita da volumi monografici che trattano singolarmente argomenti specifici inerenti a tecnologie di ambito informatico, elettrico, elettronico, meccanico e meccatronico.
Lo scopo della collana è fornire al docente strumenti didattici specifici per singoli argomenti al fine di realizzare o integrare un percorso formativo adatto al proprio piano di lavoro.
L'approccio didattico vuole essere estremamente "semplificato" senza essere banale, mantenendo rigorosità nella terminologia ma essenzialità negli aspetti teorici privilegiando l'attività laboratoriale.
I Quaderni sono quindi uno strumento didattico realmente duttile, che consente al docente di costruirsi percorsi di insegnamento su misura, combinando argomenti/temi in funzione delle proprie specifiche esigenze.
L'elenco completo dei titoli disponibili è riportato all'indirizzo web www.hoepliscuola.it.

Presentazione del volume

Python è il linguaggio di programmazione che negli ultimi anni ha avuto una diffusione tale da contendere a Java il primato di linguaggio più utilizzato dai programmatori di tutto il mondo.
È stato creato da Guido Van Rossum, ricercatore di Amsterdam che avendo lavorato a un progetto di un linguaggio di programmazione con fini didattici di nome ABC, è riuscito a trasferire questa conoscenza in Python.
Viene definito un linguaggio di scripting orientato agli oggetti in quanto coniuga la flessibilità e la semplicità dei linguaggi di scripting con la potenza di elaborazione e la ricchezza di funzioni dei più tradizionali linguaggi di programmazione di sistema.
Le principali caratteristiche di Python:

- è free, quindi i programmatori sono liberi da tutti i problemi di licenza;
- è portabile: è stato scritto in ANSI C, quindi la sua portabilità deriva direttamente da quella del C;
- è veloce: pur essendo interpretato, "compila" il proprio codice in un bytecode molto efficiente e ... lo interpreta;
- gestisce la memoria automaticamente: esiste il meccanismo di "garbage collection";
- ha una sintassi chiara ed è ricco di librerie.

Tutte queste caratteristiche stanno convincendo molti grandi attori del mercato informatico a utilizzare Python, come ad esempio Red Hat, Infoseek, Yahoo! e la NASA.

Struttura dell'opera

PROVA ADESSO!

Per mettere in pratica, in itinere, quanto appreso nella lezione

OSSERVAZIONI

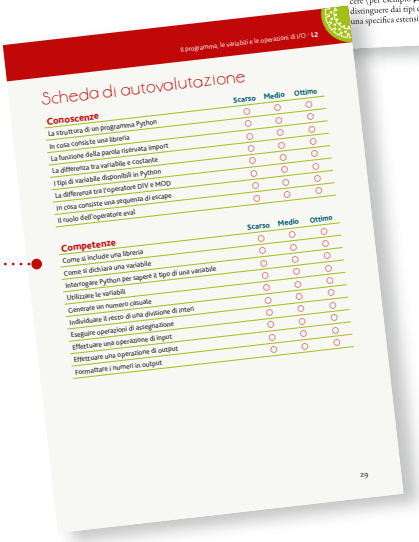
Un aiuto per comprendere e approfondire

DEFINIZIONI

La spiegazione di alcuni termini specifici

SCHEDA DI AUTOVALUTAZIONE

Alla fine di ciscuna lezione il lettore può valutare la qualità del suo apprendimento



ESERCIZI

Ampia sezione di esercizi per la verifica delle conoscenze e delle competenze

AreaDigitale

eBook+

L'eBook+ che completa il volume presenta l'intero testo in versione digitale, utilizzabile su tablet, LIM e computer, e offre alcuni contenuti aggiuntivi.

Indice

L'OFFERTA DIDATTICA **HOEPLI**

L'edizione **Openschool** Hoepli offre a docenti e studenti tutte le potenzialità di Openschool Network (ON), il nuovo sistema integrato di contenuti e servizi per l'apprendimento.

Edizione **OPENSCHOOL**




LIBRO DI TESTO

Il libro di testo è l'**elemento cardine** dell'offerta formativa, uno strumento didattico **agile** e **completo**, utilizzabile **autonomamente** o in combinazione con il ricco **corredo digitale** offline e online. Secondo le più recenti indicazioni ministeriali, volume cartaceo e apparati digitali **sono integrati in un unico percorso didattico**. Le espansioni accessibili attraverso l'eBook+ e i materiali integrativi disponibili nel sito dell'editore sono puntualmente richiamati nel testo tramite apposite icone.

eBOOK+

L'**eBook+** è la versione digitale e interattiva del libro di testo, utilizzabile su **tablet**, **LIM** e **computer**. Aiuta a comprendere e ad approfondire i contenuti, rendendo l'apprendimento più attivo e coinvolgente. Consente di leggere, annotare, sottolineare, effettuare ricerche e accedere direttamente alle numerose **risorse digitali integrative**.

➜ Scaricare l'eBook+ è molto **semplice**. È sufficiente seguire le istruzioni riportate nell'ultima pagina di questo volume.

RISORSE ONLINE

Il sito della casa editrice offre una ricca dotazione di **risorse digitali** per l'approfondimento e l'aggiornamento. Nella pagina web dedicata al testo è disponibile **MyBookBox**, il contenitore virtuale che raccoglie i materiali integrativi che accompagnano l'opera.

➜ Per accedere ai materiali è sufficiente registrarsi al sito **www.hoepliscuola.it** e inserire il codice coupon che si trova nella terza pagina di copertina.

Per il docente nel sito sono previste ulteriori risorse didattiche dedicate.

PIATTAFORMA DIDATTICA

La **piattaforma didattica** è un ambiente digitale che può essere utilizzato in modo duttile, a misura delle esigenze della classe e degli studenti. Permette in particolare di **condividere contenuti** ed **esercizi** e di partecipare a **classi virtuali**. Ogni attività svolta viene salvata sul **cloud** e rimane sempre disponibile e aggiornata. La piattaforma consente inoltre di consultare la versione online degli eBook+ presenti nella propria libreria.

➜ È possibile accedere alla piattaforma attraverso il sito **www.hoepliscuola.it**.

1

LEZIONE

PROGRAMMIAMO IN PYTHON

In questa lezione impareremo

- a installare e configurare l'ambiente di sviluppo Python
- a editare, testare e collaudare un programma in Python
- a disporre l'output sullo schermo

Il linguaggio Python

Python è un linguaggio di programmazione sviluppato da Guido Van Rossum negli anni Novanta con l'obiettivo di sintetizzare in un linguaggio semplicità e potenza: è un linguaggio ad alto livello che tenta in pratica di avvicinarsi, per quanto possibile, al ragionamento umano per cercare di semplificare al massimo la scrittura dei programmi.
È un linguaggio open source, moderno, semplice da imparare e comprensibile, gestito dal 2001 dalla Python Software Foundation, associazione indipendente, che annovera fra i suoi sponsor Sun, Canonical, O'Reilly, Microsoft, Zope.
Python è un linguaggio di programmazione interpretato, interattivo, orientato agli oggetti: combina una grande potenza con una sintassi molto chiara; inoltre si interfaccia bene con chiamate e librerie di sistema a sistemi operativi grafici ed è estendibile in linguaggio C o C++.

Il nome Python non ha nulla a che vedere con il serpente: Van Russum definì la prima release del 1991, che divenne la versione 1 solo nel 1994, prendendo parte del nome dello show comico trasmesso dalla BBC "Monty Python's Flying Circus", di cui era appassionato.

AreaDigitale

Versioni del linguaggio Python

DEFINIZIONE

Il bytecode è un codice intermedio tra il codice sorgente e il codice macchina che è svincolato dall'hardware in quanto è eseguibile su qualsiasi piattaforma e sistema operativo basta che per esso sia disponibile un interprete opportuno.

Python è un linguaggio molte facile da apprendere anche per chi lo utilizza come primo linguaggio di programmazione, data la semplicità della sintassi delle sue istruzioni: è ricco di librerie, sia native che sviluppate da terze parti, che vengono rese disponibili gratuitamente in Internet e, nonostante sia interpretato, ha ottime performance in quanto il suo codice bytecode è molto efficiente.

Come si scrive un programma in Python

Lo sviluppo di un programma avviene generalmente in due fasi distinte.
La prima fase è quella di progetto, dove si passa dal problema al programma:

- il problema viene studiato in modo da capire cosa si deve risolvere: questa operazione prende il nome di analisi del problema;
- si procede quindi con la ricerca dell'idea risolutiva, si deve cioè individuare come è possibile risolvere il problema definendo la strategia risolutiva;
- quando sono disponibili tutti gli elementi per scrivere il programma, esso viene scritto dapprima in linguaggio di progetto, poi in linguaggio di programmazione.

La fase di progetto viene effettuata "a computer spento", cioè senza l'ausilio di strumenti elettronici ma solo con l'utilizzo del "cervello umano". Tutte le fasi sono scritte rigorosamente su un foglio di carta e solo alla fine della codifica si passa al PC per compiere le operazioni della fase successiva, il collaudo vero e proprio del programma.

La seconda fase è quella di collaudo sul PC, che segue questi semplici passi:

- per prima cosa si scrivono le istruzioni del linguaggio Python in formato elettronico, utilizzando un elaboratore di testi (editing del programma);
- il programma che scriviamo in Python viene salvato in un file con un nome a piacere (per esempio **prova1**): prende il nome di programma sorgente, e per potersi distinguere dai tipi di file presenti nel sistema operativo è necessario aggiungergli una specifica estensione:

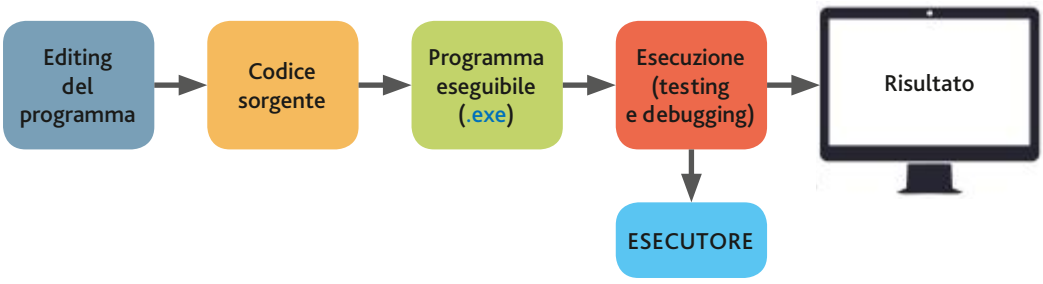
– il programma che scriviamo in Python viene salvato in un file con estensione **.py (prova.py)**.

- dato che l'elaboratore capisce solo il linguaggio binario (linguaggio macchina), abbiamo bisogno di un traduttore che prenda il programma sorgente e lo traduca in 0 e 1 per inviarlo al processore.

Compilatori e interpreti

Abbiamo tre possibili percorsi per "far arrivare" le istruzioni binarie al processore:

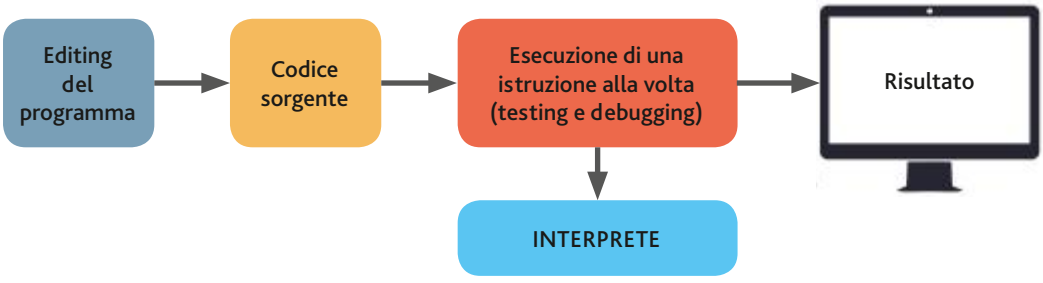
1. mediante il compilatore: questo è un programma che prende come ingresso il file sorgente che contiene il programma e, istruzione per istruzione, la trasforma e memorizza in un file, chiamato programma eseguibile (e scritto in codice binario) riconoscibile perché ha come suffisso .exe: basterà mandare in esecuzione il programma **prova1.exe** per controllare se il nostro algoritmo risponde correttamente alle richieste del problema da risolvere (test del programma) e per apportare le correzioni ed eliminare gli eventuali errori commessi (fase di debugging).



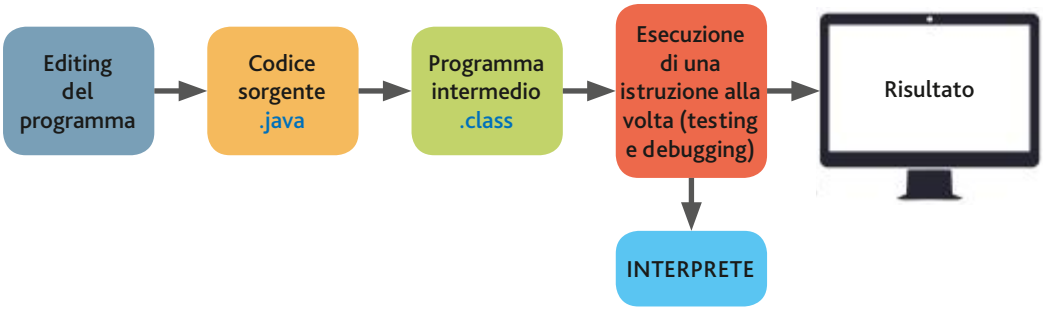
Questa procedura è il meccanismo utilizzato nei linguaggi C, C++, Visual Basic, Pascal, Fortran ...

2. mediante l'interprete: questo è un programma che prende come ingresso il file sorgente che contiene il programma e, istruzione per istruzione, la trasforma in codice binario e la manda direttamente in esecuzione, senza quindi memorizzarlo in un file eseguibile.

Questa procedura è il meccanismo utilizzato nei linguaggi Python e Javascript.



3. mediante due fasi successive: nella prima fase il compilatore genera un codice intermedio passando in "rassegna" tutto il programma sorgente e traducendolo in un nuovo formato che "non è proprio un formato binario", ma, come è intuibile dal suo nome, "una via di mezzo" preparatoria a essere eseguito in una seconda fase da un interprete: in questa terza modalità è quindi necessaria la presenza sia di un compilatore sia di un interprete.
   Questa procedura è il meccanismo utilizzato in Java.



Ambienti di sviluppo

Per effettuare la fase di collaudo di un programma sono presenti in commercio software che prendono il nome di ambienti di sviluppo e che integrano tutti gli strumenti necessari alla fase di collaudo in modo da agevolare le operazioni che un programmatore deve eseguire. Con una singola applicazione il programmatore può così editare il codice, compilarlo, eseguirlo e fare il debug semplicemente con "un clic del mouse".

Per scrivere programmi in linguaggio Python utilizzeremo la shell che viene proposta direttamente al termine dell'istallazione del pacchetto software python.

È possibile scaricarlo gratuitamente dall'indirizzo http://www.python.it/download/

È anche possibile trasformare il programma .py in formato eseguile dal calcolatore, cioè .exe, in modo che non sia anche necessaria la presenza dell'interprete Python sulla macchina.

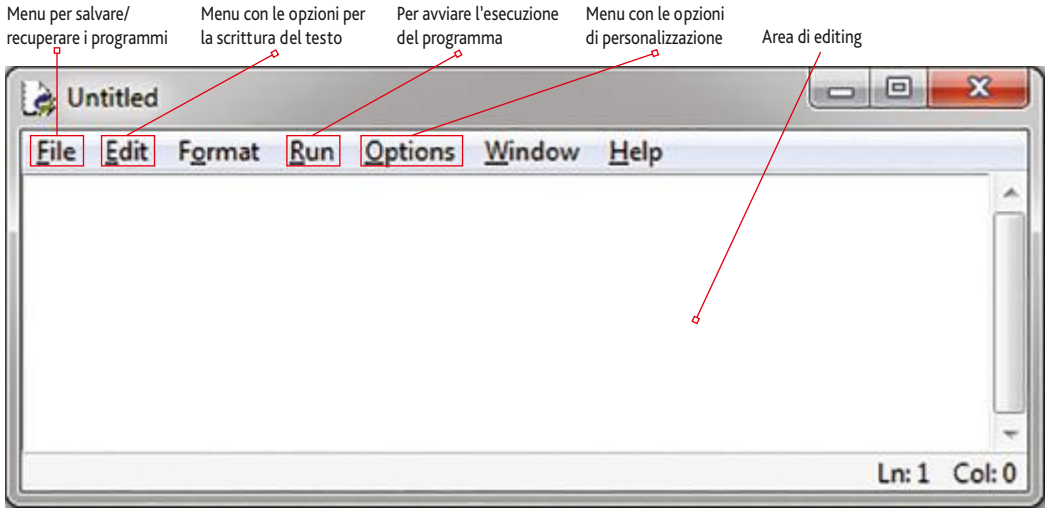
AreaDigitale

Rendere eseguibile un programma Python

Scriviamo il nostro primo programma

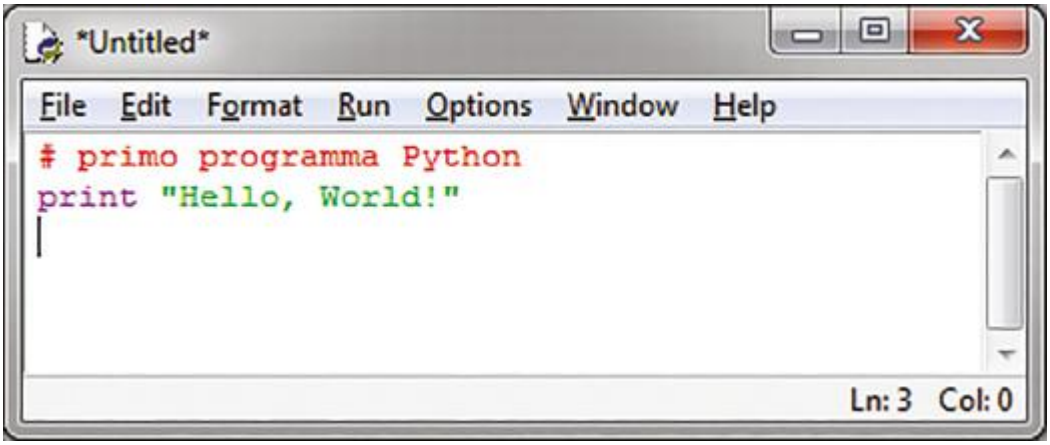
Dopo aver istallato il programma sul nostro PC, per comodità operativa creiamo un collegamento sul desktop in modo da visualizzare la finestra dell'interfaccia IDLE di Python cliccando direttamente sulla seguente icona  .

La finestra visualizzata è la seguente, nella quale possiamo individuare i principali strumenti che abbiamo a disposizione.



Editazione del codice

Scriviamo un primo programma: è divenuta un'usanza comune "di buon auspicio" scrivere come primo programma il codice che saluta "il mondo dell'informatica" per avvisare che ... "un nuovo programmatore sta arrivando!".
Trascriviamo queste due righe nell'area di testo.



Automaticamente l'editor attribuisce un colore alle diverse parole non appena vengono riconosciute e individuate come componenti del linguaggio: in questo modo il programmatore ha un immediato riscontro visivo e può direttamente verificare la correttezza di quanto sta scrivendo.

La prima istruzione, quella che inizia con "#", non è un comando che deve essere eseguito dal linguaggio: il "cancelletto" è un carattere convenzionale che indica che quanto viene scritto di seguito è un commento, cioè una frase che serve a spiegare il programma e/o a dare informazioni al programmatore.

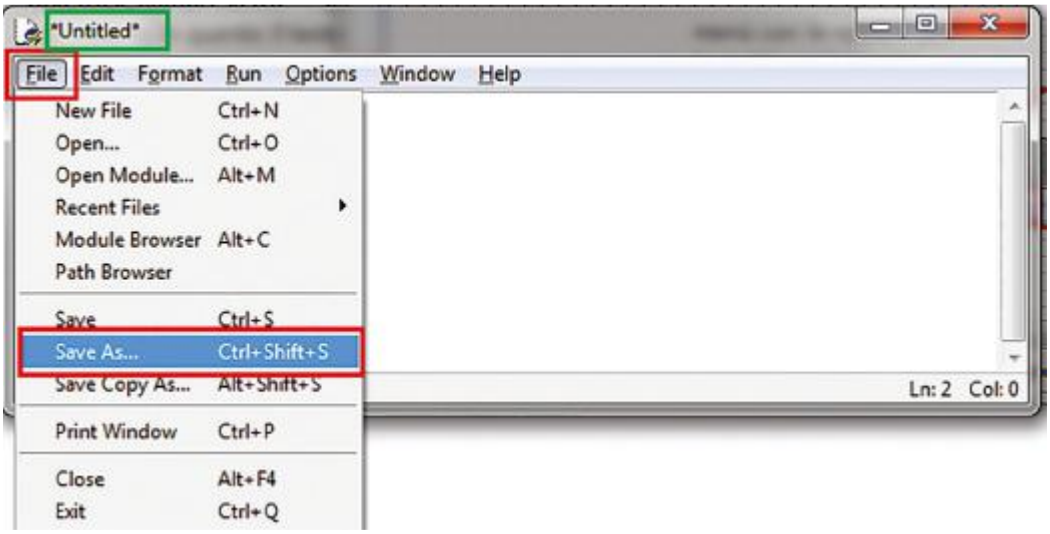
Di default in Python i commenti sono colorati in rosso, ma il programmatore può scegliere un colore personalizzato tramite il menu Options .

Nella seconda riga è presente un'istruzione che viene individuata e colorata in viola: la parola riservata è print e indica all'esecutore di stampare quanto viene scritto di seguito.

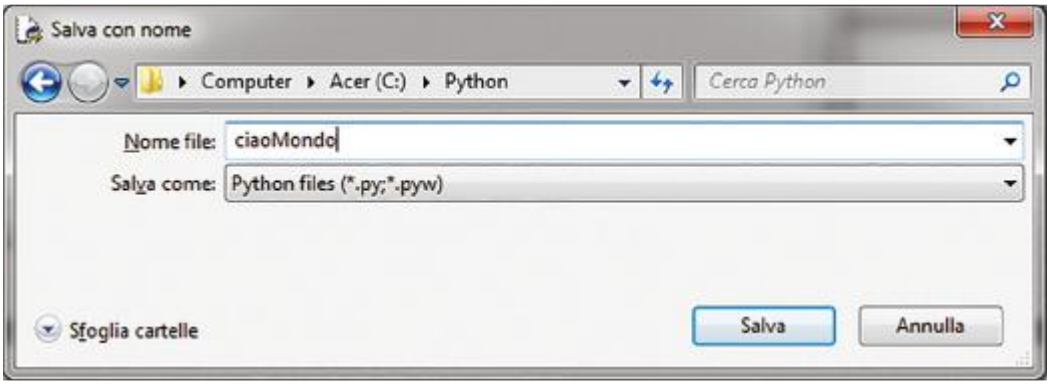
Nel nostro caso abbiamo inserito una frase compresa tra " ": anche le virgolette sono specifici indicatori di sintassi che vengono riconosciuti e colorati in verde dall'editor, colore applicato anche a tutto ciò che contengono.

Una frase compresa tra " " prende il nome di stringa di caratteri: l'interprete non cerca istruzioni al suo interno in quanto il testo tra virgolette non deve essere eseguito ma solo visualizzato.

Prima di mandare in esecuzione il programma lo memorizziamo su disco, cioè lo salviamo in un file mediante l'apposita opzione presente nel primo menu:



Cliccando l'opzione evidenziata viene visualizzata la seguente finestra nella quale possiamo scegliere la directory dove salvare il file e il nome che vogliamo assegnargli:

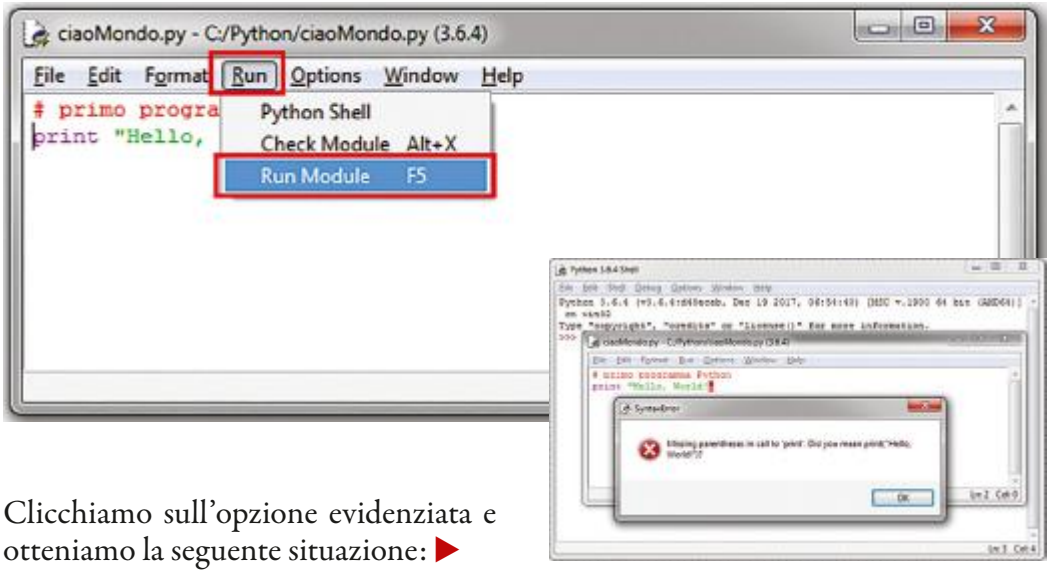


Dopo aver salvato il nostro codice possiamo osservare che viene cambiata l'intestazione della finestra e al posto di "Untitled" ora troviamo il nome che abbiamo assegnato.

Siamo pronti per testarne la correttezza mandandolo in esecuzione.

Esecuzione del codice

L'esecuzione di un programma Python viene effettuata in diversi modi ma quello che noi utilizzeremo è quello automatico collegato direttamente al nostro ambiente IDLE di sviluppo.

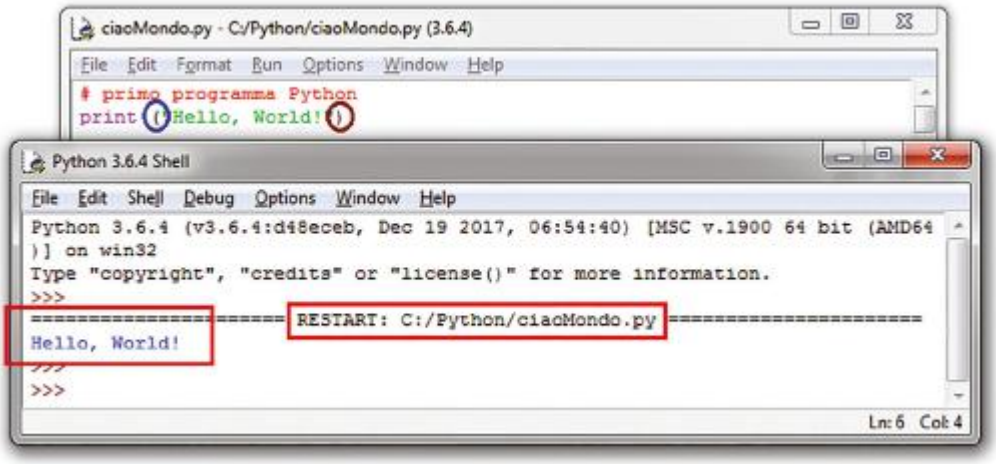


Clicchiamo sull'opzione evidenziata e otteniamo la seguente situazione: ▶

L'esecuzione del programma viene effettuata in una nuova finestra, la finestra di shell, che, come vedremo in seguito, ci offrirà un insieme di altre possibilità e modalità di esecuzione del programma. Nel nostro esempio il programma che abbiamo scritto non è andato in esecuzione in quanto ci viene segnalato un errore.

L'interprete ha "provato" a eseguirlo ma ha riscontrato un errore sintattico (SyntaxError) e ce lo indica con un messaggio in una ulteriore finestra.

Interveniamo modificando l'istruzione errata, cioè aggiungendo le parentesi, come ci viene suggerito nella finestra di errore!



Quindi mandiamo in esecuzione il programma e ora otteniamo il risultato, che viene visualizzato nella finestra di shell.

Abbiamo effettuato due fasi importanti della codifica di un programma:

- il testing: l'insieme delle operazioni che vengono effettuate per verificare la correttezza di un programma, cioè che questo produca risultati corretti;
- il debugging: la programmazione è un processo complesso e dato che esso è fatto da esseri umani spesso comporta errori, chiamati bug, e il processo della loro ricerca e correzione è chiamato debug.

AreaDigitale

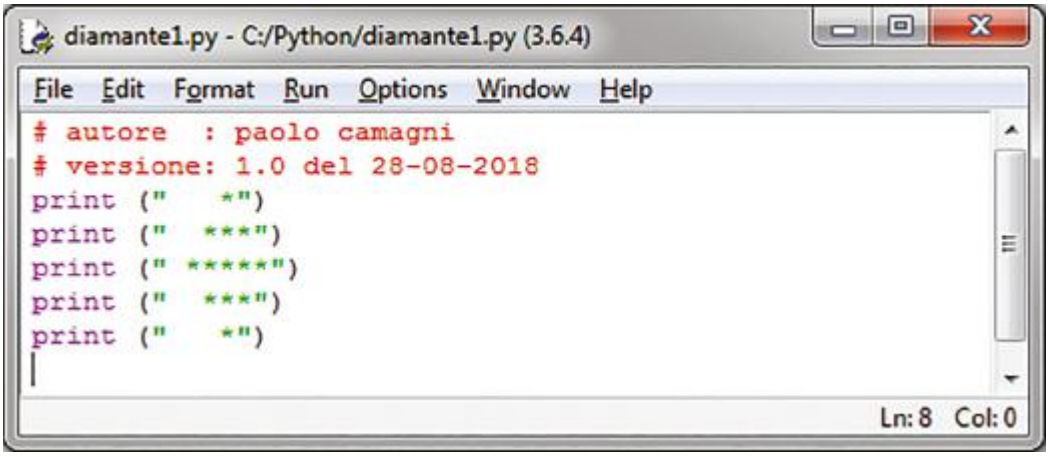
Origine del termine debugging e tipologie di errori

Un primo programma completo

Realizziamo ora un programma completo che visualizza un semplice disegno sullo schermo, o meglio, nell'area inferiore della finestra dello shell, come quello di figura.

Si devono effettuare le seguenti operazioni:

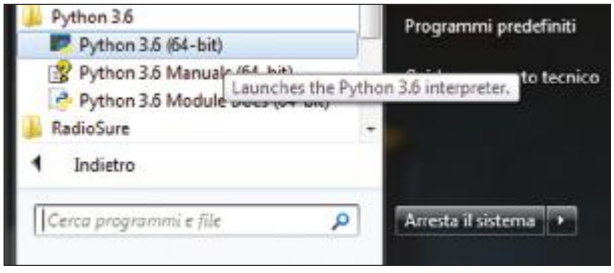
1. creare un nuovo file sorgente;
2. dargli il nome diamante1 e scrivere il codice mostrato nella figura più sotto;
3. salvandolo in un file con il nome diamante1.py;
4. mandarlo in esecuzione e correggere gli eventuali errori che ci vengono segnalati.



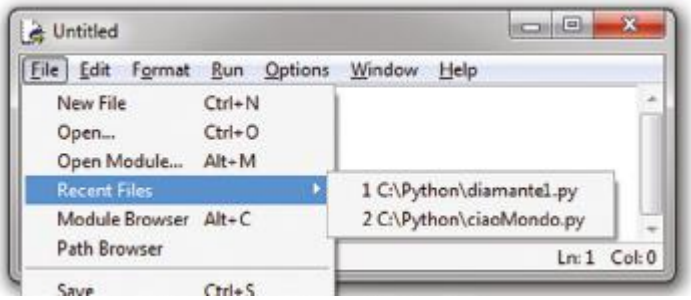
Abbiamo modificato i commenti iniziali inserendo dati utili come il nome del programmatore e la versione del programma: è importante conoscere quando e da chi è stato sviluppato un programma in modo da sapere a chi far riferimento in caso di necessità di modifiche e/o integrazioni successive alla sua creazione.

"Rientriamo" nell'ambiente Python

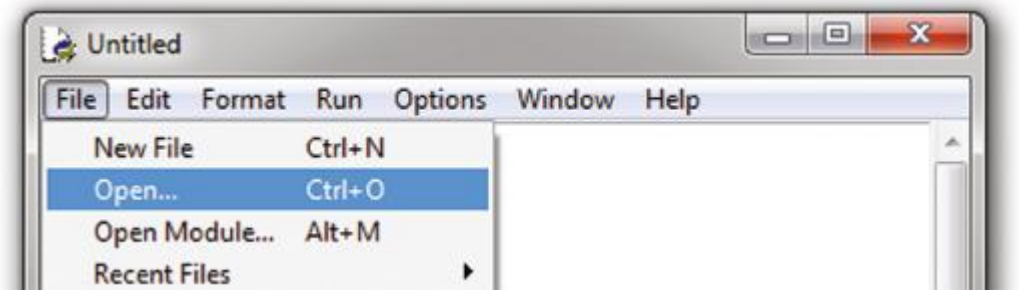
Per tornare nuovamente nell'ambiente Python è necessario mandare in esecuzione il programma facendo clic sull'icona corrispondente che abbiamo collocato sul desktop oppure su quella che automaticamente è stata aggiunta, in fase di installazione, nel menu Programmi, come mostrato nella figura. ▶



Per riprendere e modificare un programma già scritto abbiamo due possibilità. Facciamo clic sul menu File e ricerchiamo nel disco il file mediante l'opzione File-Recent Files che ci elenca gli ultimi lavori che abbiamo realizzato:



Oppure mediante l'opzione Open presente nel menu File, che visualizza la cartella che contiene tutti i nostri programmi.

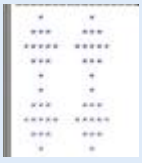


Prova adesso!

- Realizzare un programma
- Istruzioni di output

Apri il file diamante1.py

1. Modifica il programma in modo che vengano visualizzati 4 diamanti sullo schermo. Salva il programma con il nome diamante2.py e confronta il tuo codice con quello presente nelle risorse online sul sito **hoepliscuola.it** nella sezione riservata a questo volume.



2. Ripeti le stesse operazioni con il programma diamante1.py visualizza 5 diamanti sullo schermo. Salva il programma con il nome diamante3.py e confronta il tuo codice con quello con quello presente nelle risorse online sul sito **hoepliscuola.it** nella sezione riservata a questo volume.

Domande a risposta multipla

Indica la risposta corretta barrando la casella relativa.

1 L'analista:

a. scrive il programma
b. codifica un problema
c. studia il problema
d. scrive in Python

2 Il codice macchina:

a. viene scritto dal produttore della macchina
b. è un linguaggio ad alto livello
c. deve essere compilato dal compilatore
d. è il risultato della compilazione

3 Il compilatore:

a. traduce il programma sorgente in linguaggio macchina
b. traduce il programma macchina in linguaggio sorgente
c. traduce il problema in programma
d. traduce l'algoritmo in linguaggio macchina

4 Il linguaggio ad alto livello:

a. è un linguaggio come il Python
b. serve per i problemi di alto livello concettuale
c. è un linguaggio formale come l'italiano
d. è più completo del linguaggio naturale

5 Il linguaggio Python:

a. è un linguaggio di programmazione
b. è un'evoluzione del linguaggio C
c. serviva per implementare i primi sistemi operativi
d. deriva dal linguaggio Pascal

6 Un ambiente integrato di sviluppo:

a. è un linguaggio di sviluppo
b. è sempre molto costoso
c. serve all'analista per scrivere il programma
d. agevola le operazioni del programmatore

Test Vero/Falso

Indica, barrando la relativa casella, se le seguenti affermazioni sono vere o false.

1 I linguaggi di programmazione sono di diverso livello. V F

2 L'algoritmo risolve uno specifico problema. V F

3 I linguaggi di programmazione usati dai programmatori servono per scrivere codice in binario. V F

4 I linguaggi di programmazione descrivono gli algoritmi. V F

5 L'esecutore umano utilizza il linguaggio naturale. V F

6 Il linguaggio di programmazione è un linguaggio formale con una sintassi e una semantica. V F

7 Il linguaggio orientato alla macchina è composto da istruzioni estremamente semplici. V F

8 I programmi scritti in linguaggio Python devono avere suffisso .py. V F

9 Prima di mandare un programma in esecuzione questo deve essere compilato. V F

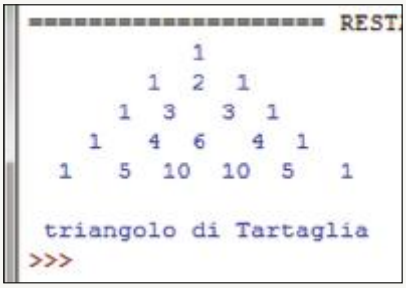
10 L'ambiente IDLE di sviluppo integrato non ha nessuna licenza. V F

Esercitiamoci

Problemi

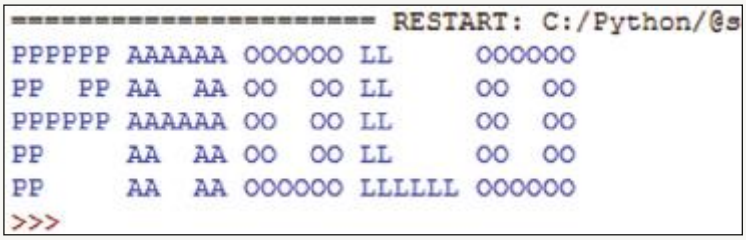
Per ciascuna delle seguenti situazioni descrivi l'algoritmo risolutivo in linguaggio Python.

1 Scrivi un programma che produca il seguente output sullo schermo e confronta la tua soluzione con quelle presenti nelle risorse online sul sito www.hoepliscuola.it nella sezione riservata a questo volume (tartaglia_solux.py).



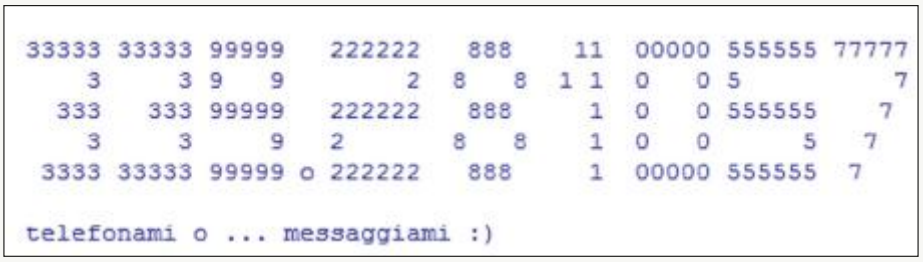
2 Scrivi un programma che produca il seguente output sullo schermo e confronta la tua soluzione con quelle presenti nelle risorse online sul sito www.hoepliscuola.it nella sezione riservata a questo volume (natale_solux.py).
Osservazione: per poter visualizzare la \, dato che è un carattere "riservato", devi metterne due affiancate, cioè print("\\");

3 Scrivi un programma che riproduca il tuo nome sullo schermo, come riportato nell'esempio seguente.



Rientra quindi nel programma e aggiungi una cornice attorno al nome.
Confronta la tua soluzione con quelle presenti nelle risorse online sul sito www.hoepliscuola.it nella sezione riservata a questo volume (nome_solux.py).

4 Scrivi un programma che visualizzi sullo schermo il tuo numero di cellulare, dove ogni cifra è formata graficamente dal numero che rappresenta, come riportato nella figura.
Confronta la tua soluzione con quelle presenti nelle risorse online sul sito www.hoepliscuola.it nella sezione riservata a questo volume (telefono_solux.py).



Scheda di autovalutazione

Conoscenze	**Scarso**	**Medio**	**Ottimo**
Cos'è un ambiente visuale	○	○	○
Differenza tra compilazione e interpretazione	○	○	○
Motivazioni sull'utilizzo di Python	○	○	○
Procedura di installazione di Python	○	○	○
Composizione dell'ambiente di lavoro	○	○	○
I contenuti della finestra dell'interfaccia IDLE	○	○	○
Le opzioni del menu principale	○	○	○
I colori utilizzati dell'editor di Python	○	○	○
La differenza tra test e debug	○	○	○
Il ruolo di un interprete	○	○	○

Competenze	**Scarso**	**Medio**	**Ottimo**
Salvare un nuovo programma	○	○	○
Recuperare e modificare un programma	○	○	○
Mandare in esecuzione un programma	○	○	○
Inserire una operazione di output	○	○	○
Inserire un commento	○	○	○
Utilizzare la finestra di shell	○	○	○