Introduzione al Corso

Corso Programmazione Python 2024

Introduzione

Luca Di Pietro Martinelli

Parte del materiale deriva dai corsi dei proff. Paolo Caressa e Raffaele Nicolussi (Sapienza Università di Roma) e Giorgio Fumera (Università degli Studi di Cagliari)

Informazioni sul corso

Durata base: 3 giornate da 7 ore ciascuna, per un totale di 21 ore di corso complessive.

Contenuti: parte di teoria e parte di esercizi, questi ultimi da svolgere durante le lezioni oppure anche al di fuori del corso.

Materiali: slide suddivise per modulo e per teoria/esercizi che vi saranno fornite al termine di ogni giornata.

Obiettivi del corso

Il corso intende fornire agli studenti:

- un'introduzione generale al mondo della programmazione e dei linguaggi di programmazione
- la storia e le caratteristiche principali che caratterizzano il linguaggio Python
- le conoscenze basilari relative alla programmazione in Python per lo sviluppo di applicazioni.

Cos’è un linguaggio di programmazione?

È un linguaggio formale che specifica un insieme di istruzioni che possono essere usate per produrre dati in uscita.

Si basa su alcuni concetti fondamentali (1/2):

- **Istruzione:** comando oppure regola descrittiva che, ad ogni esecuzione, cambia lo stato interno del calcolatore (che sia lo stato reale della macchina oppure un ambiente virtuale, teorico, creato dal linguaggio).
- **Variabile e costante:** un dato o un insieme di dati, noti o ignoti, già memorizzati o da memorizzare. A una variabile corrisponde sempre, da qualche parte, un certo numero (fisso o variabile) di locazioni di memoria che vengono allocate, cioè riservate, per contenere i dati stessi.
- **Espressione:** una combinazione di variabili e costanti, unite da operatori.

4

Cos’è un linguaggio di programmazione?

È un linguaggio formale che specifica un insieme di istruzioni che possono essere usate per produrre dati in uscita.

Si basa su alcuni concetti fondamentali (2/2):

- **Struttura dati:** meccanismo che permette di organizzare e gestire dati complessi
- **Struttura di controllo:** che permette di governare il flusso di esecuzione del programma, alterandolo in base al risultato o valutazione di un'espressione
- **Funzionalità di input/output:** possibilità di inserire dati da tastiera e visualizzarli in output (stampa a video) attraverso i cosiddetti canali standard (standard input, standard output)
- **Commento:** possibilità di inserire un commento sul codice scritto, sintatticamente identificato e delimitato, che ne esplichi le funzionalità a beneficio della leggibilità o intelligibilità.

5

Lessico, sintassi e semantica

- **Lessico:** insieme di regole formali per la scrittura di "parole" in un linguaggio.
- **Sintassi:** insieme di regole formali per la scrittura di "frasi" (istruzioni) in un linguaggio, che stabiliscono cioè la grammatica del linguaggio stesso.
- **Semantica:** l'insieme dei significati da attribuire alle frasi (che devono essere sintatticamente corrette) costruite nel linguaggio.

Nota: una frase può essere sintatticamente corretta e tuttavia non avere significato!

Livelli di astrazione

Due tipologie di linguaggi di programmazione in funzione del livello di astrazione.

Linguaggi ad alto livello

Significativa astrazione dai dettagli del funzionamento di un calcolatore e dalle caratteristiche del linguaggio macchina. Richiedono un compilatore o un interprete che traduca le istruzioni di alto livello in linguaggio di basso livello (linguaggio macchina).

Vantaggi:

- Portabilità: possibilità di esecuzione su qualsiasi calcolatore
- Leggibilità: facilità di comprensione sia nello scrivere che nel leggere il codice
- Manutenibilità: semplicità nell'effettuare modifiche.

Svantaggi:

- Controllo limitato sulla compilazione del codice
- Possibile riduzione di efficienza.

7

Livelli di astrazione

Due tipologie di linguaggi di programmazione in funzione del livello di astrazione.

Linguaggio a basso livello

Coincidono o si avvicinano molto al linguaggio macchina fornendo poca o nessuna astrazione dai dettagli del funzionamento fisico del calcolatore.

Vantaggi:

- Controllo totale delle istruzioni che verranno eseguite dalla macchina.

Svantaggi:

- Dipendenza dalla macchina su cui il linguaggio viene eseguito.
- Difficoltà di lettura e scrittura del codice.

8

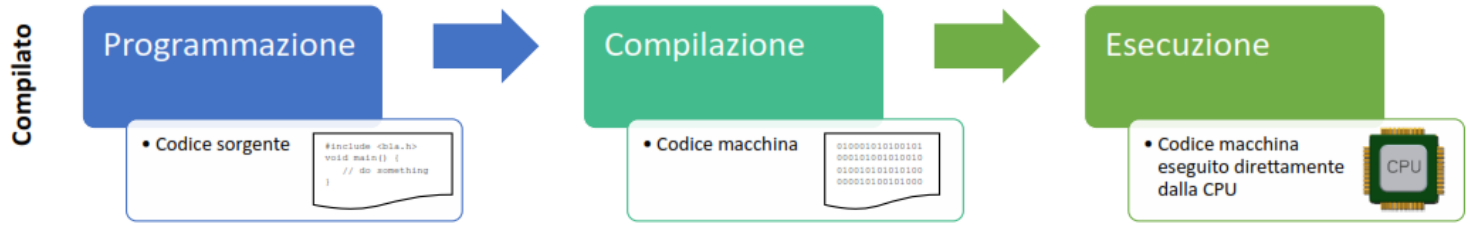
Linguaggi compilati vs. interpretati

Linguaggio compilato

Compilazione: in un linguaggio compilato il codice sorgente viene tradotto in linguaggio macchina (binario) da un compilatore prima dell'esecuzione del programma. Il risultato della compilazione è un file eseguibile che può essere eseguito direttamente dal sistema operativo.

Velocità di esecuzione: i programmi scritti in linguaggi compilati tendono ad essere più veloci rispetto a quelli scritti in linguaggi interpretati poiché il codice è già tradotto in linguaggio macchina e non richiede traduzione in tempo reale durante l'esecuzione.

Portabilità: i programmi compilati possono essere meno portabili poiché l'eseguibile generato dipende dall'architettura della macchina su cui è stato compilato.



9

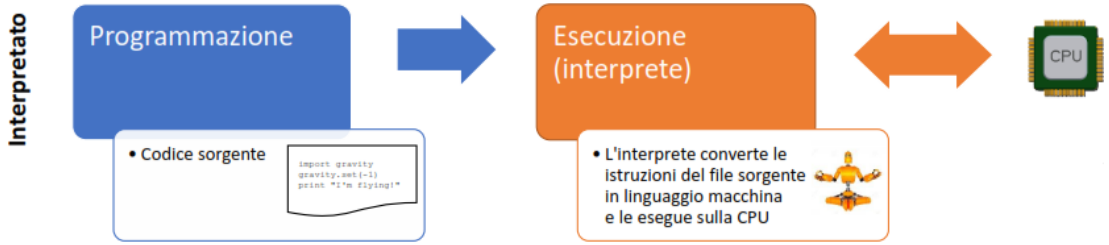
Linguaggi compilati vs. interpretati

Linguaggio interpretato

Esecuzione: in un linguaggio interpretato il codice sorgente viene eseguito direttamente da un interprete senza essere trasformato in codice macchina. L'interprete legge ed esegue le istruzioni una alla volta durante l'esecuzione del programma.

Velocità di esecuzione: i linguaggi interpretati tendono generalmente ad essere più lenti rispetto ai linguaggi compilati poiché le istruzioni vengono tradotte in tempo reale durante l'esecuzione anziché essere tradotte in linguaggio macchina prima dell'esecuzione.

Portabilità: i linguaggi interpretati sono più portabili perché l'interprete può essere implementato su diverse piattaforme senza dover modificare il codice sorgente.



10