Introduzione al Python

Python è un linguaggio di programmazione ad alto livello, interpretato, orientato agli oggetti e multipiattaforma. È stato creato da Guido van Rossum alla fine degli anni '80 ed è diventato uno dei linguaggi di programmazione più popolari al mondo.

Python è utilizzato in molte aree diverse, tra cui:

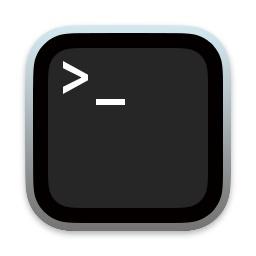
* **Sviluppo web:** Python viene utilizzato per sviluppare applicazioni web utilizzando framework come Django, Flask e Pyramid.
* **Data science:** Python è molto popolare nel campo della data science grazie alle librerie come NumPy, Pandas, Matplotlib e Scikit-learn.
* **Automazione e scripting:** Python è un'ottima scelta per l'automazione e lo scripting grazie alla sua facilità d'uso e alla sua ampia gamma di librerie.
* **Intelligenza artificiale e machine learning:** Python è un linguaggio molto popolare per l'AI e il machine learning grazie a librerie come TensorFlow, Keras e PyTorch.

Python è un linguaggio molto facile da imparare e leggere grazie alla sua sintassi semplice e intuitiva. Inoltre, la sua ampia gamma di librerie e strumenti lo rende molto potente e versatile.

Durante lo svolgimento dell’Hackathon noi consigliamo di utilizzare questo linguaggio di programmazione, in quanto molto semplice da utilizzare.

Siamo sicuri che potrà tornarvi utile anche nello sviluppo di tanti altri programmi nel futuro!

Installazione Python su MacOS e Linux

Su macchine con sistema operativo MacOS o con distribuzioni Linux Python è già presente ed è accessibile utilizzando il programma **“Terminale”** del computer.

Installazione Python su Windows

Per iniziare a programmare con Python, la prima cosa da fare è scaricare e installare il software. Vai sul sito ufficiale di Python: <https://www.python.org/downloads/>

Clicca sul pulsante "Download" e seleziona la versione più recente di Python. Inoltre, se stai utilizzando Windows, seleziona anche la versione a 64 bit o a 32 bit in base al tuo sistema operativo.

Installare un editor di testo

Ora che hai installato Python, hai bisogno di un editor di testo per scrivere il codice. Ci sono molte opzioni tra cui scegliere, ma vi consigliamo di utilizzare Visual Studio Code.

Puoi scaricare Visual Studio Code dal sito ufficiale: <https://code.visualstudio.com/>

Una volta installato, apri Visual Studio Code e seleziona "Extensions" dal menu laterale. Cerca e installa l'estensione "Python" per aggiungere funzionalità di sviluppo Python.

Scrivere e salvare il codice

Ora sei pronto per scrivere il tuo primo programma Python.

Apri Visual Studio Code e seleziona "File" > "Nuovo file".

Scrivi il seguente codice Python nel file:

Immagine che contiene testo, logo

Descrizione generata automaticamente

Salva il file con un nome di tuo gradimento e con l'estensione **.py** (ad esempio, "**hello.py**").

Ora per eseguire il codice fare click su “Esegui” > “Esegui senza eseguire il debug” oppure cliccando sul tasto play in alto a destra su Visual Studio Code.

A questo punto si aprirà una schermata del terminale che mostrerà il risultato dell’esecuzione del programma Python.

Le basi di Python

Python utilizza la sintassi indentata per definire i blocchi di codice. Questo significa che l'indentazione (i tab o gli spazi) sono importanti per separare le sezioni di codice. Di seguito troverai alcune basi di Python:

**La funzione print() viene utilizzata per visualizzare un output sullo schermo.**

Immagine che contiene testo, logo

Descrizione generata automaticamente

Output:

Hello, world!

**Le variabili vengono utilizzate per memorizzare dati. Una variabile è un nome che fa riferimento ad un valore specifico.**

Immagine che contiene testo, dispositivo, metro

Descrizione generata automaticamente

**Python supporta molti tipi di dati, come interi, float, stringhe e booleani.**

Immagine che contiene testo

Descrizione generata automaticamente

**Strutture di controllo di flusso**

L'istruzione **if** viene utilizzata per eseguire un blocco di codice se una determinata condizione è vera.

Immagine che contiene testo, dispositivo, metro

Descrizione generata automaticamente

Output:

x è maggiore di 5

Il ciclo **while** viene utilizzato per eseguire un blocco di codice finché una determinata condizione è vera.

Immagine che contiene testo

Descrizione generata automaticamente

Output:

1

2

3

4

5

Il ciclo **for** viene utilizzato per eseguire un blocco di codice per ogni elemento in un elenco.

Immagine che contiene testo

Descrizione generata automaticamente

Output:

mela

banana

kiwi

**Funzioni**

Una funzione è un blocco di codice riutilizzabile che esegue un'azione specifica. Di seguito troverai un esempio di funzione in Python:

**Immagine che contiene testo

Descrizione generata automaticamente**

Output:

8

Un Esempio Di Programma

Ecco un esempio di programma Python che utilizza i concetti spiegati in precedenza.

Immagine che contiene testo

Descrizione generata automaticamente

In questo esempio, abbiamo definito una funzione chiamata "trova\_massimo" che prende in input una lista di numeri e restituisce il massimo valore presente nella lista utilizzando un ciclo for e un'istruzione if.

Successivamente, abbiamo creato una lista di numeri casuali e l'abbiamo stampata utilizzando un ciclo for.

Infine, abbiamo utilizzato la funzione "trova\_massimo" per trovare il massimo valore della lista e abbiamo stampato il risultato.

Questo è solo un semplice esempio di come è possibile utilizzare i concetti di base della programmazione con Python per creare un programma funzionale.

Le Librerie In Python

In Python, una libreria è un insieme di moduli predefiniti che possono essere importati all'interno del tuo programma per fornirti funzionalità avanzate. Le librerie Python sono ampiamente utilizzate in molti campi, dalla scienza dei dati alla creazione di applicazioni web.

Per utilizzare una libreria in Python, devi innanzitutto installarla sul tuo computer. Ci sono diverse modalità per installare le librerie in Python, ma la più comune è utilizzare il gestore di pacchetti pip. Puoi installare una libreria specifica utilizzando il **comando** **da terminale** "pip install nome\_libreria".

Dopo aver installato la libreria, devi importarla nel tuo programma utilizzando la parola chiave "import". Ad esempio, se vuoi utilizzare la libreria **NumPy** per lavorare con i numeri, puoi utilizzare il seguente comando all'inizio del tuo programma:

Immagine che contiene testo

Descrizione generata automaticamente

In questo modo, la libreria NumPy sarà disponibile all'interno del tuo programma.

Puoi anche utilizzare l'istruzione "import" per importare un singolo modulo all'interno di una libreria. Ad esempio, se vuoi utilizzare solo la funzione "sqrt" per calcolare la radice quadrata, puoi utilizzare il seguente comando:

Immagine che contiene testo

Descrizione generata automaticamente

In questo modo, la funzione "sqrt" sarà disponibile all'interno del tuo programma senza dover importare l'intera libreria math.

Una volta importata la libreria, puoi utilizzare le sue funzionalità all'interno del tuo programma. Ad esempio, se hai importato la libreria NumPy, puoi utilizzare la funzione "numpy.array" per creare un array di numeri:

Immagine che contiene testo

Descrizione generata automaticamente

In sintesi, le librerie in Python sono un modo potente per estendere le funzionalità del tuo programma e sfruttare il lavoro che è stato già fatto dagli altri sviluppatori. Utilizzando le librerie giuste, puoi risparmiare molto tempo e concentrarti sulla risoluzione dei problemi specifici del tuo progetto.

Librerie: Un Esempio Di Programma

Immagine che contiene testo

Descrizione generata automaticamente

In questo esempio, abbiamo importato la libreria NumPy all'inizio del programma utilizzando l'istruzione "import numpy".

Successivamente, abbiamo creato un array di numeri casuali utilizzando la funzione "numpy.array".

Abbiamo poi utilizzato la funzione "numpy.mean" per calcolare la media dell'array e la funzione "numpy.std" per calcolare la deviazione standard dell'array.

Infine, abbiamo stampato i risultati utilizzando l'istruzione "print".

In questo modo, abbiamo utilizzato la libreria NumPy per eseguire operazioni matematiche avanzate su un array di numeri in modo semplice e veloce.

Richieste HTTP

Le richieste HTTP (Hypertext Transfer Protocol) sono il meccanismo utilizzato dal browser web per comunicare con i server web e richiedere informazioni o risorse dal server.

Quando un utente digita un URL nel browser web, il browser invia una richiesta HTTP al server che ospita il sito web corrispondente all'URL (es.: [www.google.it](http://www.google.it) , [www.python.org](http://www.python.org) ).

La richiesta HTTP contiene informazioni come il tipo di richiesta (GET, POST, PUT, DELETE, etc.), il tipo di contenuto richiesto (HTML, immagini, file audio o video, ecc.), e altri parametri, come ad esempio informazioni sui cookie.

Il server web riceve la richiesta HTTP e risponde con una **risposta HTTP** contenente il contenuto richiesto (o un messaggio di errore, se la richiesta non può essere soddisfatta). La risposta HTTP include informazioni come il codice di stato HTTP (che indica se la richiesta è stata completata con successo o meno), il tipo di contenuto restituito, e altre informazioni come i cookie.

Ci sono diversi tipi di richieste HTTP che il browser web può inviare al server web. Qui di seguito sono elencati i più comuni:

1. **GET:** è il tipo di richiesta HTTP più comune. Viene utilizzato per richiedere un documento o una risorsa dal server web, come ad esempio una pagina HTML, un'immagine o un file JSON.
2. **POST:** è utilizzato per inviare dati dal client al server web. Ad esempio, se si compila un modulo di registrazione su un sito web, i dati inseriti nel modulo vengono inviati al server tramite una richiesta HTTP POST.
3. **PUT:** viene utilizzato per aggiornare una risorsa esistente sul server web. Ad esempio, è possibile utilizzare una richiesta HTTP PUT per modificare un file su un server web.

Un esempio di richiesta GET per una pagina web:

**GET /index.html HTTP/1.1** <- la richiesta di tipo GET

**Host:** [**www.example.com**](http://www.example.com)

**User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:90.0) Gecko/20100101 Firefox/90.0**

**Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8**

**Accept-Language: en-US,en;q=0.5**

**Accept-Encoding: gzip, deflate, br**

**Connection: keep-alive**

**Cookie: session\_id=1234567890**

Risposta:

**HTTP/1.1 200 OK**  <- il codice di risposta

**Date: Fri, 29 Apr 2023 10:00:00 GMT**

**Server: Apache/2.4.10 (Debian)**

**Last-Modified: Mon, 24 Apr 2023 10:00:00 GMT**

**ETag: "123456-abcdefg-1234567890"**

**Content-Type: text/html**

**Content-Length: 1234**

**Connection: close**

<!-- La pagina web di risposta -->

**<html>**

**<head>**

**<title>Example</title>**

**</head>**

**<body>**

**<p>Welcome to the example website.</p>**

**</body>**

**</html>**

I cinque codici di risposta HTTP più comuni che un server web può restituire:

1. **200 OK**: Indica che la richiesta HTTP è stata elaborata correttamente dal server e la risorsa richiesta è stata trovata e restituita al client.
2. **404 Not Found**: Indica che il server non è stato in grado di trovare la risorsa richiesta. Questo potrebbe accadere se il client ha richiesto una pagina che non esiste sul server.
3. **401 Unauthorized**: Indica che il server richiede una autenticazione prima di fornire l'accesso alla risorsa richiesta.
4. **403 Forbidden**: Indica che il server rifiuta la richiesta perché il client non ha le autorizzazioni per accedere alla risorsa richiesta.
5. **500 Internal Server Error**: Indica che si è verificato un errore interno sul server durante l'elaborazione della richiesta HTTP. Questo potrebbe accadere a causa di un errore di configurazione del server o di un problema con il codice sorgente dell'applicazione web.

Ci sono molti altri codici di risposta HTTP possibili, ognuno dei quali ha un significato specifico, ma questi sono i più comuni che si possono incontrare.

Richieste HTTP in Python

In Python, puoi fare richieste HTTP utilizzando la libreria "requests":

1. **Installare la libreria "requests":** La prima cosa da fare è assicurarsi che la libreria "requests" sia installata sul tuo computer. Puoi farlo utilizzando il comando "pip install requests" nel terminale del tuo sistema operativo.
2. **Importare la libreria "requests":** Dopo aver installato la libreria "requests", devi importarla nel tuo programma utilizzando la parola chiave "import".
3. **Fare una richiesta GET:** Una volta importata la libreria "requests", puoi fare una richiesta HTTP utilizzando la funzione "requests.get(url)". Ad esempio, se vuoi fare una richiesta GET all'URL "<https://jsonplaceholder.typicode.com/posts>":

Immagine che contiene testo

Descrizione generata automaticamente

In questo modo, la funzione "requests.get" farà una richiesta HTTP GET all'URL specificato e salverà la risposta nella variabile "response".

1. **Analizzare la risposta:** Dopo aver fatto una richiesta HTTP, puoi analizzare la risposta utilizzando le funzionalità della libreria "requests". Ad esempio, puoi utilizzare la proprietà "status\_code" della risposta per verificare se la richiesta è stata eseguita correttamente. Puoi anche utilizzare la proprietà "text" per ottenere il contenuto della risposta in formato testo:

Immagine che contiene testo

Descrizione generata automaticamente

In questo modo, stiamo controllando se lo status code della risposta è uguale a 200 (il che significa che la richiesta è stata eseguita correttamente) e, in tal caso, stamperemo il contenuto della risposta. Se lo status code non è 200, stamperemo un messaggio di errore.

Di seguito invece un esempio completo di una richiesta HTTP di tipo POST (utilizzata per inviare dati come, ad esempio, username e password quando facciamo il login) con python.

Immagine che contiene testo

Descrizione generata automaticamente

In questo esempio, stiamo facendo una richiesta HTTP POST all'URL "<https://jsonplaceholder.typicode.com/posts>" con il payload specificato. Stiamo anche specificando l'intestazione della richiesta POST utilizzando il parametro "headers". Inoltre, stiamo utilizzando il metodo "json" della libreria "requests" per inviare il payload come JSON.

Dopo aver inviato la richiesta POST, stiamo verificando lo status code della risposta utilizzando la proprietà "status\_code" dell'oggetto "response". Se lo status code è 201, stamperemo il contenuto della risposta utilizzando la proprietà "content". Se lo status code non è 201, stamperemo un messaggio di errore.

Per un esempio di tool automatico che faccia richieste GET e POST in sequenza fare riferimento al file **esempio-tool-automatico-richieste-GET-POST.py**