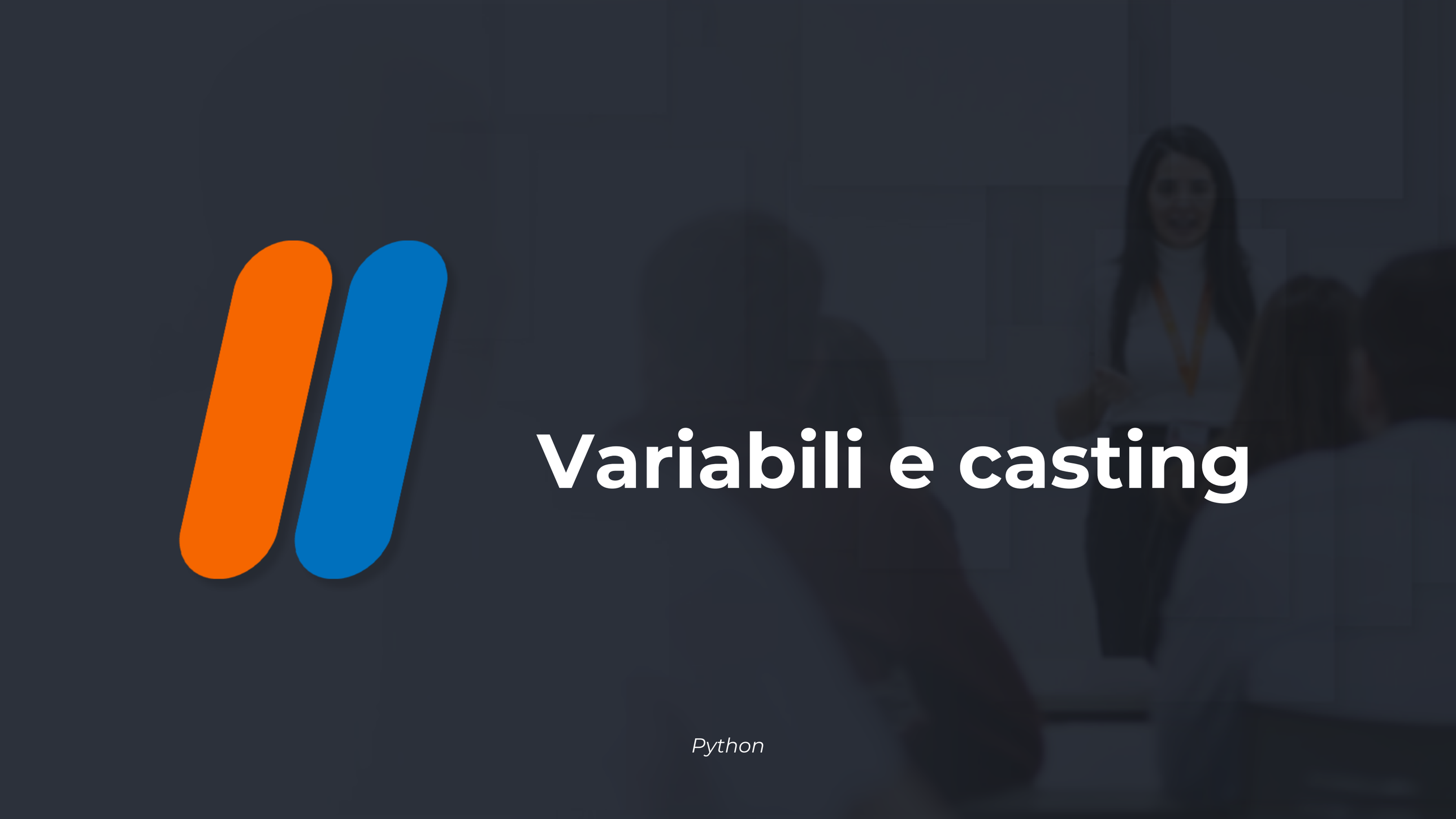


Variabili e casting

Python

In Python, una variabile è un nome che si riferisce a un valore memorizzato nella memoria del computer.

Le variabili sono fondamentali per programmare in Python poiché consentono di memorizzare, modificare e recuperare dati mentre il programma viene eseguito.

Python è un linguaggio di programmazione dinamico, il che significa che non è necessario dichiarare il tipo di una variabile quando la si crea. Il tipo di una variabile viene determinato automaticamente quando le si assegna un valore.

Ecco una panoramica dei tipi di variabili più comuni in Python:

- Numeri Intero (int): Rappresenta numeri interi, sia positivi che negativi, senza parti decimali. Esempio: numero = 10.

- Numero con Virgola Mobile (float): Rappresenta numeri reali, inclusi quelli con parti decimali. Esempio: altezza = 1.75.

- Stringhe (str): Rappresentano testo o sequenze di caratteri. Devono essere racchiuse tra apici singoli (' ') o doppi (" "). Esempio: nome = "Mario".

- Booleani (bool): Rappresentano verità con due possibili valori: True (vero) o False (falso). Esempio: is_online = True.

- Liste (list): Rappresentano una raccolta ordinata di elementi che possono essere di tipo diverso. Le liste sono mutabili, il che significa che i loro contenuti possono essere modificati. Esempio: numeri = [1, 2, 3, 4].

- Tuple (tuple): Simili alle liste, ma immutabili. Una volta creato, il contenuto di un tuple non può essere modificato. Esempio: coordinate = (10.0, 20.0).

- Dizionari (dict): Rappresentano coppie chiave-valore. Sono collezioni non ordinate e le chiavi sono uniche all'interno di un dizionario. Esempio: studente = {"nome": "Luca", "età": 22}.

- Set (set): Rappresentano una collezione non ordinata di elementi unici. Sono utili per rimuovere duplicati da una sequenza e per eseguire operazioni matematiche come unione, intersezione, differenza. Esempio: colori = {"rosso", "verde", "blu"}.

Python supporta anche tipi di dati più complessi, come gli oggetti, che sono fondamentali nel paradigma della programmazione orientata agli oggetti (OOP).

Nel contesto dell'OOP, le variabili che appartengono a un oggetto o a una classe sono spesso chiamate "attributi" o "proprietà", e le funzioni associate a un oggetto sono chiamate "metodi".

Il casting in Python si riferisce al processo di conversione di una variabile da un tipo all'altro, ad esempio da un intero a un float, o da una stringa a un intero.

Questa operazione è comune in molti linguaggi di programmazione e può essere particolarmente utile in Python dato il suo tipaggio dinamico.

Python fornisce un set di funzioni integrate per eseguire il casting tra tipi di dati.

Ecco una panoramica delle funzioni di casting più comuni:

int(): Converte il suo argomento in un intero.

Può essere usato per convertire float in interi (truncato, non arrotondando), o per convertire stringhe che rappresentano numeri interi in veri e propri interi.

1. `n = int(2.8)  # n sarà 2`
2. `m = int("3")  # m sarà 3`

float(): Converte il suo argomento in un float. È utile per convertire interi o stringhe (che rappresentano numeri decimali o interi) in numeri a virgola mobile.

```
1. x = float(5)     # x sarà 5.0
2. y = float("5.5") # y sarà 5.5
```

str(): Converte il suo argomento in una stringa. Può essere utilizzato per convertire quasi ogni tipo di dato in una rappresentazione stringa.

```
1. a = str(20)    # a sarà '20'
2. b = str(10.01) # b sarà '10.01
```

bool(): Converte il suo argomento in un booleano. Restituisce **False** se l'argomento è in qualche modo "vuoto" (come **0**, **0.0**, **''**, **[]**, **()**), altrimenti restituisce **True**.

```
1. c = bool(0)   # c sarà False
2. d = bool(123) # d sarà True
```

list(), **tuple()**, **set()**, **dict()**: Queste funzioni convertono i loro argomenti nei corrispondenti tipi di collezioni. Sono particolarmente utili per convertire tra diversi tipi di sequenze o per creare una nuova collezione da un iteratore.

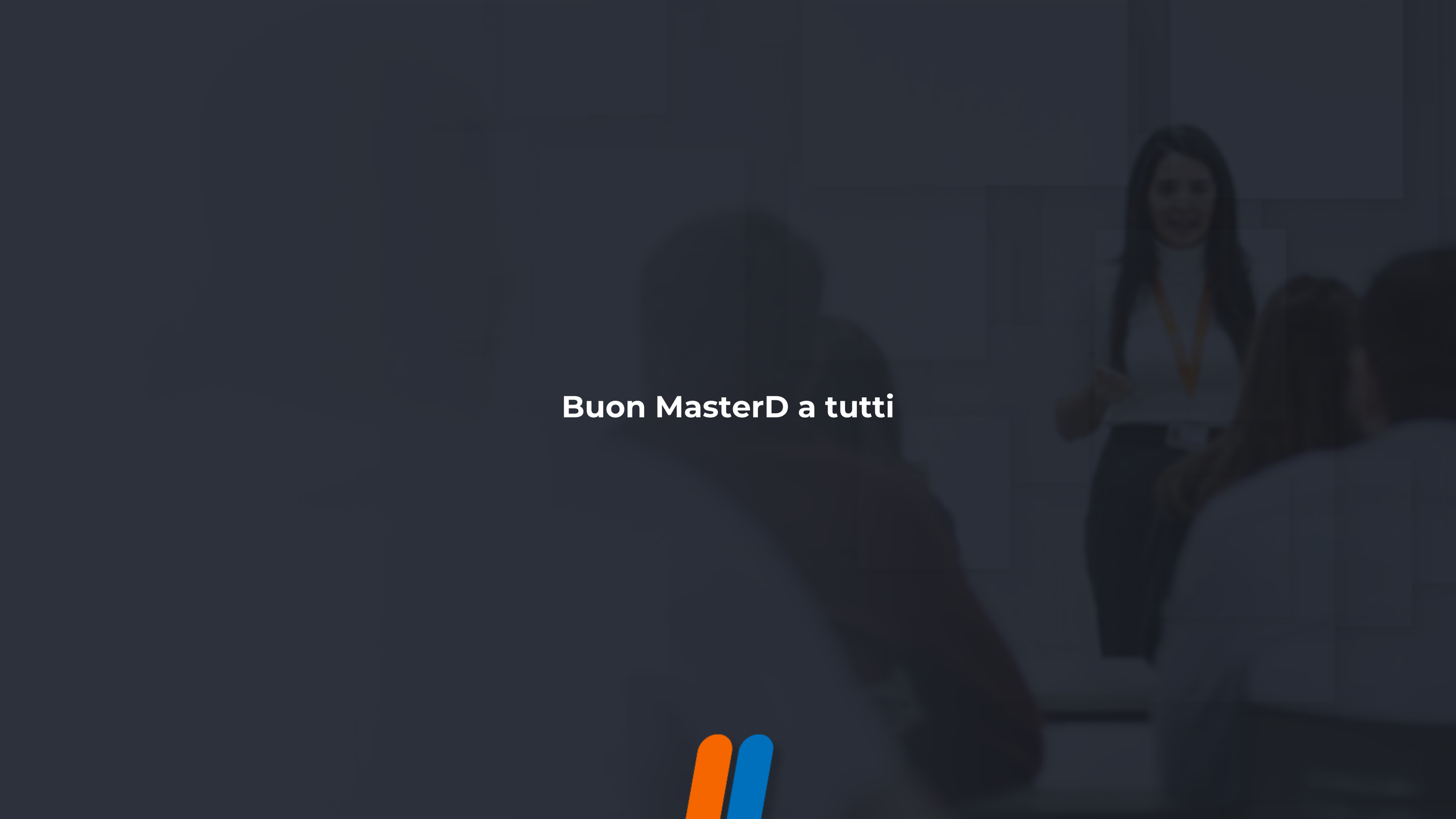
```
1. lista_da_tuple = list((1, 2, 3))       # Sarà [1, 2, 3]
2. tuple_da_lista = tuple([1, 2, 3])    # Sarà (1, 2, 3)
3. set_da_lista = set([1, 2, 2, 3, 3])    # Sarà {1, 2, 3}
```

Il casting è importante per diversi motivi:

- Validazione dei dati: Assicura che i dati siano del tipo corretto prima di procedere con operazioni che potrebbero altrimenti fallire o comportarsi in modo inaspettato.

- Conversione di dati: Utile quando si ricevono dati in un formato (ad esempio, da input utente o file) e si necessita di convertirli in un altro formato per elaborazioni successive.

- Interoperabilità: Consente a parti di codice che si aspettano tipi di dati diversi di lavorare insieme in modo più armonioso.


Buon MasterD a tutti