

```
##### Funzioni (seconda parte)
# La volta scorsa ci eravamo lasciati col discorso delle funzioni. Avevamo visto
# come è possibile eliminare codice rindondante, e fare funzioni che in
# qualche modo si "adattano" alle richieste. Come nel caso in cui abbiamo una
# funzione che può accettare sia due o tre parametri. Nel primo caso, il terzo
# valore sarà un valore di default. Ma può anche accadere il contrario:
# Potremmo avere bisogno di una funzione che ignori alcuni parametri che gli
# vengono passati. In questo caso, usiamo "dummy":

# Uso di "dummy"

def funzione(var1, var2, dummy):
    return var1*var2

print funzione(2, 3, "a") # Il terzo argomento viene ignorato

##### Dizionari
# Ecco a voi il magnifico mondo dei dizionari. Molto spesso capita di dovere
# attribuire a un valore un'altro valore. Questo si potrebbe fare con una
# lista, ma sarebbe dispendioso. I dizionari ci permettono di fare questo
# lavoro in maniera pulita e semplice.

# Uso delle liste

lista1=["mamma", "papà", "figlio"]
lista2=[1954, 1952, 1990]

posizione=lista1.index("mamma") # Restituisce la posizione di "mamma"
valore=lista2[posizione] # Restituisce il valore
print valore # Visualizza il risultato

# Con l'uso dei dizionari

dizionario={"mamma":1954, "papà":1952, "figlio":1990}
print dizionario["mamma"] # Visualizza il valore di "mamma"

# E continuiamo a divertirci!
dizionario["mamma"]=1955 # Ora abbiamo cambiato il valore a mamma
dizionario["cugino"]=1987 # "cugino" non esiste... Viene creato!
print dizionario.keys() # Visualizza tutte le chiavi in una lista
print dizionario.values() # Visualizza tutti i valori in una lista
del dizionario["mamma"] # Cancelliamo la mamma!

##### Librerie
# Sarebbe troppo dispendioso programmare alcune parti di codice che si usano
# spessissimo... Per questo esistono le librerie. Python è "con le batterie
# incluse", ovvero ci viene dato con una vasta collezione di librerie.

# Importare ed usare una libreria

import sys # Importa sys
sys.exit(0) # Usa la funzione exit che fa uscire il programma con il segnale 0

# ...oppure...

from sys import * # Importa tutto quello che c'è in sys.*
exit(0)

# ...oppure...

from sys import exit # Importa solo exit da sys, non tutto sys
```

```
exit(0)
```

```
# ...oppure...
```

```
from sys import exit as chiuditi # Attenzione, non darle un nome già in uso
chiuditi(0)
```

```
# Questo qui sopra è un esempio scemo, perché status 0 è la norma e perché si
# può benissimo usare (senza importare nulla): raise SystemExit(0)
```

```
# Esempio pratico
```

```
from time import strftime as now
print now("%M:%H") # Stampa i minuti + ":" + l'ora corrente
print now("%D") # Visualizza la data completa come 08/13/07 (all'inglese)
print now("%d/%m/%y") # Visualizza la data come 13/08/07
```