

Indentazione

Informatica@DSS 2019/2020 — Il canale

Massimo Lauria <massimo.lauria@uniroma1.it>
<http://massimolauria.net/courses/informatica2019/>

Indentazione

L'inserimento di spazio vuoto all'inizio della riga, per

- ▶ identificare blocchi logici di codice
- ▶ rendere il codice più leggibile

Esempio di indentazione annidata

```
def scontato(prezzo,sconto):      1
    if sconto < 0 or sconto > 100: 2
        print("Errore nell'input") 3
        print("Lo sconto deve essere tra 0 e 100") 4
        return 5
    6
    percentuale = 100 - sconto 7
    prezzo_scontato = prezzo * percentuale / 100 8
    return prezzo_scontato 9
10
print(scontato(1000,20)) 11
print(scontato(500,15)) 12
```

In Python l'indentazione è importante

Le istruzioni nello stesso blocco devono essere allineate

```
print("Prima riga")      1  
    print("seconda riga") 2
```

```
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
    File "/tmp/babel-YszcCQ/python-VERkEL", line 2  
        print("seconda riga")  
        ^  
IndentationError: unexpected indent
```

```
print("Prima riga")      1  
print("seconda riga")    2
```

```
Prima riga  
seconda riga
```

Tab vs Spazi

Il carattere “tabulazione” (TAB) indica “aggiungi un livello di indentazione”. Sfortunatamente

- ▶ è visivamente uguale a una sequenza di spazi
- ▶ la sequenza ha lunghezza differente (2,3,4,8... spazi) a seconda della visualizzazione.

```
<spazio><spazio><spazio><spazio>istruzione1  
<tabulazione>istruzione2
```

In editor o terminali diversi si ottiene:

```
istruzione1  
istruzione2
```

```
istruzione1  
istruzione2
```

```
istruzione1  
    istruzione2
```

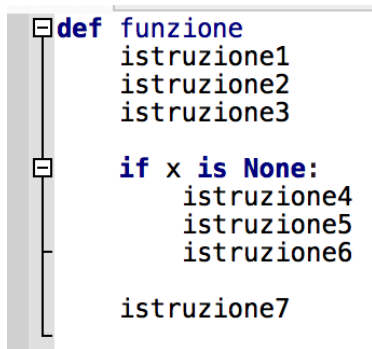
Tab vs Spazi in python3

In python3

- ▶ è vietato mischiare Tab e Spazi nell'indentazione
- ▶ si consiglia di usare solo spazi (tipicamente 4 per livello)
- ▶ è possibile impostare l'editor così che inserisca 4 spazi ogni volta che si preme TAB.

Tab vs Space nell'editor Geany (I)

Geany evidenzia i livelli di indentazione. Linee spurie possono indicare che Tab e Spazi si sono mischiati.

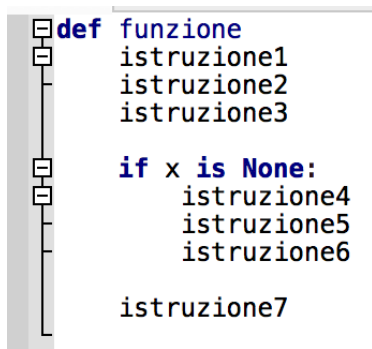


The image shows a screenshot of the Geany text editor. On the left, a vertical grey bar contains a bracket that groups the first three lines of code ('def funzione', 'istruzione1', 'istruzione2', 'istruzione3') together, indicating they are at the same indentation level. Another bracket groups the next three lines ('if x is None:', 'istruzione4', 'istruzione5', 'istruzione6') together, indicating they are at a higher indentation level. The line 'istruzione7' is at the base indentation level. This visual feedback confirms that only spaces were used for indentation.

```
def funzione
    istruzione1
    istruzione2
    istruzione3

    if x is None:
        istruzione4
        istruzione5
        istruzione6

istruzione7
```



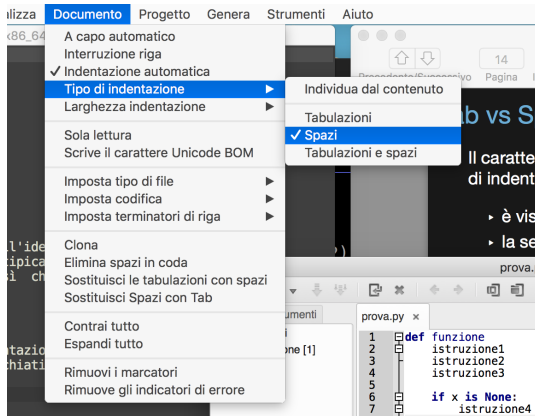
The image shows a screenshot of the Geany text editor with the same code as the previous image. However, the vertical bracket on the left is misaligned. It correctly groups the first three lines, but the bracket for the 'if' block is shifted to the right, indicating that the indentation of those lines is not consistent with the others. This misalignment is a visual cue that a mix of tabs and spaces was used for indentation, which Geany is highlighting as a problem.

```
def funzione
    istruzione1
    istruzione2
    istruzione3

    if x is None:
        istruzione4
        istruzione5
        istruzione6

istruzione7
```

Tab vs Space nell'editor Geany (II)



Quanto indentare

Io suggerisco 4 spazi.

- ▶ la lunghezza dell'indentazione è facoltativa
- ▶ non compromettete la leggibilità

```
x = 12 1
print("Primo livello di indentazione, 0 spazi") 2
if x > 0: 3
    print("Secondo livello di indentazione, 2 spazi") 4
    if x < 100: 5
        print("Terzo livello di indentazione, 1 spazio") 6
    else: 7
        print("Secondo livello di indentazione, 5 spazi") 8
```

De-indentare

Ridurre l'indentazione comunica al Python che la nuova istruzione fa parte di un blocco di codice più esterno, al quale questa deve essere allineata.

```
x = 10 1
2
def gruppo_istruzioni(): 3
    print("Tizio") 4
    print('Caio') 5
    print("Sempronio") 6
7
gruppo_istruzioni() 8
```

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/var/folders/kf/p7km5ptj1p52hvv5nl6j9gr80000gn/T/babel-oj12Dc/python-u
    gruppo_istruzioni()
    ^
IndentationError: unindent does not match any outer indentation level
```

Commenti e righe vuote

Ignorati da python. L'indentazione è sempre considerata rispetto alla precedente riga contenente vero codice.

```
x = 10 1
2
def gruppo_istruzioni(): 3
    print("Tizio") 4
    # commento mal indentato. Brutto ma corretto 5
    print('Caio') 6
    print("Sempronio") 7
8
    # altro commento mal indentato 9
gruppo_istruzioni() 10
```

```
Tizio
Caio
Sempronio
```