



2. Manipolatori per l'Input/Output formattato

Con gli operatori di I/O `>>` e `<<` è possibile impostare le formattazioni che consentano di migliorare la visualizzazione dei dati.

Nel linguaggio C++ esistono alcuni strumenti definiti nel file delle librerie standard **`iomanip`**. Per usare le formattazioni descritte in seguito occorre quindi introdurre nel programma la dichiarazione di inclusione:

```
#include <iomanip>
```

I principali manipolatori di I/O sono riassunti nella seguente tabella.

Manipolatore	Descrizione
<code>endl</code>	carattere di fine riga (simile a <code>'\n'</code>)
<code>ends</code>	carattere di fine stringa (simile a <code>'\0'</code>)
<code>dec</code>	imposta la base decimale
<code>oct</code>	imposta la base ottale
<code>hex</code>	imposta la base esadecimale
<code>setw(b)</code>	imposta il numero di battute (default <code>b=0</code> , senza formattazione)
<code>setfill(c)</code>	imposta il carattere di riempimento (default <code>c=spazio</code>)
<code>setprecision(d)</code>	imposta il numero di decimali (default <code>d=6</code>)

I manipolatori `dec`, `oct`, `hex` possono essere utilizzati anche con l'operatore di input (`>>`). Per i manipolatori `dec`, `oct`, `hex`, `setfill`, `setprecision`, l'impostazione del manipolatore permane per tutto il codice, fino a nuova impostazione.

Il manipolatore `setw` è invece valido solo per l'output successivo.

Lo schema seguente illustra alcuni esempi di utilizzo di questi manipolatori:

Codice	Output
<pre>int i; cout << "Introdurre un numero decimale: "; cin >> dec >> i; cout << "dec/oct/hex:\t"; cout << dec << i << '\t' << oct << i << '\t' << hex << i << endl;</pre>	<pre>Introdurre un numero decimale: 10 dec/oct/hex: 10 12 a</pre>
<pre>cout << "Introdurre un numero esadecimale: "; cin >> hex >> i; cout << "dec/oct/hex:\t"; cout << dec << i << '\t' << oct << i << '\t' << hex << i << endl;</pre>	<pre>Introdurre un numero esadecimale: 10 dec/oct/hex: 16 20 10</pre>
<pre>cout << dec << endl;</pre>	(ripristina la rappresentazione decimale)



Codice	Output
<pre>cout << setfill('*') << setw(4) << 25 << '\t' << 25 << endl;</pre>	<pre>**25 25</pre>
<pre>cout << setfill(' ');</pre>	(ripristina lo spazio di riempimento)
<pre>cout << 1./3. << '\t' << setprecision(3) << 1./3. << endl;</pre>	<pre>0.333333 0.333</pre>
<pre>int valore1 = 100; int valore2 = 200; int totale; cout << setw(20) << "Milano" << setw(6) << valore1 << endl; cout << setw(20) << "Roma" << setw(6) << valore2 << endl; totale = valore1 + valore2; cout << setfill('-') << setw(21) << ' ' << setw(5) << '-' << setfill(' ') << endl; cout << setw(20) << "Totale" << setw(6) << totale << endl;</pre>	<pre> Milano 100 Roma 200 ----- Totale 300</pre>

Si noti l'uso della sequenza di escape `\t` per indicare la tabulazione sulla riga.

La tabella seguente presenta l'elenco completo delle **sequenze di escape** utilizzate nel linguaggio C++:

Sequenza di escape	Descrizione	Terminologia inglese
<code>\n</code>	A capo riga	<i>new line</i>
<code>\t</code>	Tabulazione	<i>tab</i>
<code>\r</code>	Ritorno a capo della stessa riga	<i>carriage return</i>
<code>\"</code>	Doppi apici	<i>double quote</i>
<code>\\</code>	Barra contraria	<i>backslash</i>
<code>\b</code>	Una battuta indietro	<i>backspace</i>
<code>\'</code>	Apice singolo	<i>single quote</i>
<code>\?</code>	Punto di domanda	<i>question mark</i>
<code>\a</code>	Segnalazione acustica	<i>bell</i>
<code>\0</code>	Fine stringa	<i>end of string</i>
<code>\f</code>	Salto pagina	<i>form feed</i>



ESERCIZIO

Quali delle seguenti affermazioni sono vere (V) e quali false (F)?

Nell'istruzione

```
cout << setw(n)
      << setprecision(m)
      << dato
      << endl;
```

- a) n indica il valore che è stato assegnato alla variabile *dato*
- b) n indica il numero totale dei caratteri con cui verrà presentato il valore di *dato*
- c) n indica il numero delle cifre intere con cui verrà presentato il valore di *dato*
- d) dopo il valore di *dato* viene scritto il numero decimale $n.m$
- e) *dato* deve essere di tipo *float* o *double*
- f) *dato* deve essere di tipo *int*
- g) m indica il numero di cifre decimali con cui viene scritto *dato*
- h) il contenuto di m è stampato con molta precisione

