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

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
`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;`
	`Introdurre un numero decimale: 10`
	`dec/oct/hex:  10  12  a`

`cout << "Introdurre un numero esadecimale: ";`
`cin >> hex >> i;`
`cout << "dec/oct/hex:\t";`
`cout << dec << i<< '\t'`
`<< oct << i<< '\t'`
`<< hex << i<< endl;`
	`Introdurre un numero esadecimale: 10`
	`dec/oct/hex:  16  20  10`

`cout << dec << endl;`	(ripristina la rappresentazione decimale)

Codice	Output

`cout << setfill('*')`
`<< setw(4)`
`<< 25`
`<< '\t'`
`<< 25`
`<< endl;`
	`**25  25`

`cout << setfill(' ');`	(ripristina lo spazio di riempimento)

`cout << 1./3.`
`<< '\t'`
`<< setprecision(3)`
`<< 1./3.`
`<< endl;`
	`0.333333  0.333`

`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;`
	`Milano  100`
	`Roma  200`
	`--------------------  -----`
	`Totale  300`

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

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