

Ein/Ausgabe in C++ (iostream,iomanip)

- **Formatierte Ausgabe über cout**

Syntax: `cout << Argument1 << Argument2 << ...`

Die Formatierung der Ausgabe kann durch sog. Manipulatoren beeinflusst werden. (*Auszug*)

Manipulator	Wirkung	Bemerkungen
<code>setw(n)</code> ¹	Min. Feldbreite (Voreinst.: 0)	<code>#include <iomanip></code>
<code>setprecision(n)</code>	Ziffern nach Dezimalpunkt (Voreinst.: 6)	<code>#include <iomanip></code>
<code>setfill(c)</code>	Füllzeichen c (Voreinst.: '␣')	<code>#include <iomanip></code>
<code>left</code>	linksbündig	
<code>right</code>	rechtsbündig (Voreinst.)	
<code>internal</code>	Vorzeichen linksbündig, Zahl rechtsbündig	
<code>showpos</code>	pos. Vorzeichen ausgeben	
<code>noshowpos</code>	pos. Vorzeichen nicht ausgeben (Voreinst.)	
<code>fixed</code>	Festpunktformat	
<code>scientific</code>	Exponentialformat	
<code>dec</code>	Dezimalausgabe (Voreinst.)	
<code>oct</code>	Oktalausgabe	
<code>hex</code>	Hexadezimalausgabe	
<code>showbase</code>	führende 0 (oktal), 0x, 0X (hex.) ausg.	
<code>noshowbase</code>	keine Zahlssystemangabe (Voreinst.)	
<code>uppercase</code>	Großbuchst. in der Zahlausg.	
<code>nouppercase</code>	Kleinbuchst. in der Zahlausg. (Voreinst.)	
<code>endl</code>	Zeilenvorschub	
<code>flush</code>	Ausgabepuffer leeren	
<code>ends</code>	Zeichenkettenendezeichen ausg.	

Statt durch Manipulatoren können Formatflags auch durch `cout.setf` bzw. durch Komponentenfunktionen gesetzt werden.

<code>cout.width(n)</code> ¹	Min. Feldbreite
<code>cout.precision(n)</code>	Ziffern nach Dezimalpunkt
<code>cout.fill(c)</code>	Füllzeichen c
<code>cout.setf(ios::left,ios::adjustfield)</code>	linksbündig
<code>cout.setf(ios::right,ios::adjustfield)</code>	rechtsbündig
<code>cout.setf(ios::internal,ios::adjustfield)</code>	Vorz. linksbündig, Zahl rechtsbündig
<code>cout.setf(ios::showpos)</code>	pos. Vorzeichen ausgeben
<code>cout.unsetf(ios::showpos)</code>	pos. Vorzeichen nicht ausgeben
<code>cout.setf(ios::fixed,ios::floatfield)</code>	Festpunktformat
<code>cout.setf(ios::scientific,ios::floatfield)</code>	Exponentialformat
<code>cout.setf(ios::dec,ios::basefield)</code>	Dezimalausgabe
<code>cout.setf(ios::oct,ios::basefield)</code>	Oktalausgabe
<code>cout.setf(ios::hex,ios::basefield)</code>	Hexadezimalausgabe
<code>cout.setf(ios::showbase)</code>	führende 0 (oktal), 0x, 0X (hex.) ausg.
<code>cout.unsetf(ios::showbase)</code>	keine Zahlssystemangabe
<code>cout.setf(ios::uppercase)</code>	Großbuchstaben in der Zahlausgabe
<code>cout.unsetf(ios::uppercase)</code>	Kleinbuchstaben in der Zahlausgabe
<code>cout.flush()</code>	Ausgabepuffer leeren

¹Wirkt nur auf die nächste Ausgabeoperation

- **Formatierte Eingabe über cin**

Syntax: cin >> *Argument1* >> *Argument2* >> ...

Mit Manipulatoren kann der Eingabevorgang verändert werden. (*Nicht g++-2.95.2!*)

skipws führenden Zwischenraum überlesen (Voreinst.)
noskipws führenden Zwischenraum nicht überlesen
dec oct hex Dezimal/Oktal/Hexadezimaleingabe

Die entsprechenden Formatflags können auch mit `cin.setf` gesetzt werden.

cin.setf(ios::skipws) führenden Zwischenraum überlesen (Voreinst.)
cin.unsetf(ios::skipws) führenden Zwischenraum nicht überlesen
cin.setf(ios::dec,ios::basefield) Dezimaleingabe
cin.setf(ios::oct,ios::basefield) Oktaleingabe
cin.setf(ios::hex,ios::basefield) Hexadezimaleingabe

Wird neben der C++-Ein/Ausgabe auch die C-Ein/Ausgabe benutzt, dann sollte die Funktion `ios::sync_with_stdio()` vor der ersten C++-Ein/Ausgabeoperation aufgerufen werden.

- **Beispiele für die formatierte Ausgabe in C++**

Programm:

Ausgabe:

```
#include <iomanip>
#include <iostream>
using namespace std;

int main()
{ int i=123; double x=12.345; char s[]="Ausgabe";

  cout << "|" << setw(6) << i << "|" << endl;
  cout << "|" << setw(6) << left << i << "|" << endl;
  cout << "|" << setw(2) << right << i << "|" << endl;

  cout << "|" << fixed << x << "|" << endl;
  cout << "|" << setw(8) << setprecision(4)
    << x << "|" << endl;
  cout << "|" << setw(8) << showpos << setprecision(2)
    << x << "|" << endl;
  cout << "|" << noshowpos << right << setw(1)
    << setprecision(1) << x << "|" << endl;
  cout << "|" << scientific << setprecision(6)
    << x << "|" << endl;
  cout << "|" << setw(14) << x << "|" << endl;
  cout << "|" << setw(14) << setprecision(7)
    << uppercase << x << "|" << endl;

  cout << "|" << setw(10) << s << "|" << endl;
  cout << "|" << setw(10) << left << s << "|" << endl;
  return 0;
}
```