Kapitel 4

Ein- & Ausgabe

Ein- und Ausgabe

- Für die Ausgabe auf die Standardausgabe (üblicherweise der Monitor) kann das Ausgabe-Objekt *cout* verwendet werden.
- Für die Eingabe vom Standardeingabegerät (Tastatur) wird das Eingabe-Objekt *cin* verwendet.

Ausgabe - cout

- Beispiele:

```
cout << "Texteingabe";
cout << x;
cout << a << b;
cout << c << d << endl;
cout << "y = " << y << endl;
```

Ausgabe - cout

- Am Ende einer *cout*-Anweisung wird kein automatischer Zeilenvorschub eingefügt. Dies erreicht man durch Einfügen des Manipulators *endl* in den Ausgabestream.
- Bei der Ausgabe einer arithmetischen Größe werden so viele Stellen wie benötigt ausgegeben.
- Bei der Ausgabe von Gleitkommazahlen werden diese in Abhängigkeit von ihrem Wert mit oder ohne Exponenten ausgegeben.

Ausgabe-Manipulatoren

- Die Breite des Ausgabefeldes kann mit dem Manipulator *setw(n)* beeinflusst werden.
- *n* gibt die Anzahl der Stellen des Ausgabefeldes an.
- Dieser Manipulator wirkt nur auf die direkt folgende Ausgabe und gilt für alle Datentypen.

```
cout << "123456789012345678901";
cout << setw(15) << 1.602E-19
```

```
12345678901234567890
        1.602e-019
```

Ausgabe-Manipulatoren

- Die Anzahl der signifikanten Stellen bei der Ausgabe kann über den Manipulator *setprecision(n)* beeinflusst werden.
- Der Standardwert für *n* ist 6.
- Die Einstellung bleibt für alle weiteren Ausgaben bestehen!

```
cout << setprecision(10) << 10.0/3.0 << endl;
```

```
12345678901234567890
3.333333333
```

Ausgabe-Manipulatoren

- Zur Verwendung der Manipulatoren muss die Datei iomanip eingebunden werden:

```
#include <iomanip>
```

Ausgabe-Manipulatoren (kompakt)

Methode	Beschreibung
setw(n)	Gibt eine Mindestausgabebreite von n Zeichen vor. Die Ausgabe wird rechtsbündig angeschlagen und links mit Leerzeichen aufgefüllt. Die Ausgabebreite wird bei Bedarf kommentarlos überschritten. Gilt nur immer für die nächste Ausgabe!

Ausgabe-Manipulatoren (kompakt)

Methode	Beschreibung
setprecision(n)	Gibt an, wie viele Stellen einer Zahl ausgegeben werden sollen. (Standardwert für n ist 6.) Gilt für Vor- und Nachkommastellen. Gibt man im Ausgabestrom das Schlüsselwort "fixed" an, so gilt n nur für die Nachkommastellen. setprecision gilt für alle nachfolgenden Ausgaben.

Eingabe - cin

- Die Werte der Variablen werden in der Reihenfolge eingegeben, in der sie in der *cin*-Liste aufgeführt sind.
- Die Werte werden durch Leerzeichen oder durch Return getrennt. Die Eingabe wird mit Return abgeschlossen.

```
double x,y,z;
cin >> x >> y >> z;
```

Eingabe:

```
1.0
2.3
5
```

Eingabe - cin

- Eingabefehler kann man erkennen, indem man sie durch *fail* abfängt:

```
if (cin.fail())
        cout << "Eingabefehler!" << endl;
```

Escape-Sequenzen

- Graphisch nicht darstellbare Zeichen können mit Escape-Sequenzen dargestellt werden.
- Um sie darzustellen, werden sie als zwei Zeichen geschrieben.
- Eingeleitet werden sie mit \ .
- Sie benötigen aber trotzdem nur ein Byte.

Escape-Sequenzen

Zeichen	Bedeutung
\a	Klingelzeichen
\b	Backspace
\f	Seitenvorschub
\n	Neue Zeile
\r	Wagenrücklauf
\t	Tabulator
\0	String-Ende
\\	Backslash
\"	Anführungszeichen
\'	Hochkomma
\xhh	Numerischer Wert eines Zeichens hexadezimal

Beispiel

```
#include <iostream>
using namespace std;
void main ()
{
        cout << "\tDies ist \n\tein Beispiel \n\tmit ";
        cout << "Escape-Sequenzen." << "\n";
}
```

```
        Dies ist
        ein Beispiel
        mit Escape-Sequenzen.
```

Beispiel

```
#include <iostream>
#include <cmath>
using namespace std;
void main ()
{
        double a = 8.8, b = 7.24, c = 5.8;
        double y;
        y = a + sqrt(b - c);
        cout << "y = " << y << endl;
}
```

```
y = 10
```

Beispiel

```
#include <iostream>
#include <cmath>
#include <iomanip>
using namespace std;
void main ()
{
        double x = 1.357, y = 1.821, z = 2.092;
        cout << setprecision(4);
        cout << "Zahl" << "\t" << "Wurzel" << endl;
        cout << x << "\t" << sqrt(x) << endl;
        cout << y << "\t" << sqrt(y) << endl;
        cout << z << "\t" << sqrt(z) << endl;
}
```

Zahl	Wurzel
1.357	1.165
1.821	1.349
2.092	1.446