Kapitel 3

Datentypen und Variablen

Datentypen - Einführung

- Für jede Variable muss ein Datentyp festgelegt werden.
- Hierdurch werden die Wertemenge und die verwendbaren Operatoren festgelegt.
- Für jeden Datentyp gibt es eine Festlegung, in welcher Weise Konstanten eindeutig dargestellt werden können.
- In C++ gibt es eine Reihe vordefinierter Datentypen.
- Aus den vordefinierten Datentypen können weitere Datentypen zusammengesetzt werden.

Datentypen - Einführung

- In dieser Tabelle sind die arithmetischen Datentypen aufgelistet:

Datentyp	Länge	Wertebereich	
		von	bis
char	1 Byte	-2^7	2^7-1
int	4 Byte	-2^{31}	$2^{31}-1$
float	4 Byte	$\pm 1.2\ 10^{-38}$	$\pm 3.4\ 10^{+38}$
double	8 Byte	$\pm 2.2\ 10^{-308}$	$\pm 1.8\ 10^{+308}$

Ganzzahlige Konstanten

- Ganze Zahlen (ganzzahlige Konstanten) werden unter den Datentyp *int* gespeichert.
- Der Computer speichert die Zahlen aber nicht in der Form, wie wir sie benutzen. (Also z.B. die Zahl 6 wird nicht als "6" gespeichert.)
- Die Speicherung erfolgt als Binärzahl. (z.B. 0110 für die Zahl 6)
- Um auch negative Zahlen speichern zu können, wird die Zahl bei den meisten Rechenwerken in der Zweierkomplementdarstellung gespeichert.

Ganzzahlige Konstanten

Beispiel: 4 wird im Computer so gespeichert:

1. Vorzeichen ignorieren und ins Binärsystem umrechnen
   4 => 0000 0100

Ganzzahlige Konstanten

Beispiel: -4 wird im Computer so gespeichert:

1. Vorzeichen ignorieren und ins Binärsystem umrechnen
   4 => 0000 0100
2. Invertieren, da negativ
   1111 1011
3. Eine 1 addieren, da negativ
   1111 1011 + 0000 0001 => 11111100

Ganzzahlige Konstanten

- Durch die Speicherung der Zahlen in Zweierkomplementdarstellung können wir auf dem vorgesehenen Platz zwar nur halb so viele Zahlen speichern.
- Jedoch ermöglicht uns dies, dass die Verarbeitung von positiven und negativen Zahlen im Rechenwerk gleich bleiben kann. Ansonsten müsste der Computer bei Berechnungen positive und negative Zahlen intern unterschiedlich behandeln.

Ganzzahlige Konstanten - Eingabe

- Ganzzahlige Konstanten können dezimal, oktal oder hexadezimal in einem C-Programm angegeben werden.
- Die Unterscheidung wird anhand der ersten Ziffer vorgenommen.

Darstellung	1. Ziffer	Beispiel
dezimal	1 bis 9	4 +4 -67 101
oktal	0	012 +0167 -013
hexadezimal	0x	0xC2 -0x3D

Beispiel

```
#include <iostream>
using namespace std;
void main ()
{
	cout << 10;
	cout << 012;
	cout << 0xa;
}
```

```
10
10
10
```

Ganzzahlige Konstanten - Ausgabe

- Ganzzahlige Konstanten können dezimal, oktal oder hexadezimal ausgegeben werden.
- Die Darstellung wird bestimmt durch den Befehl *setbase().*

Beispiel

```
#include <iostream>    #include <iomanip>    using namespace std;
void main () {
        int var = 10;
        cout << "Die Zahl " << var << " ist" << endl
        cout << "hexadezimal: " << setbase(16) << var << endl;
        cout << "oktal: " << setbase(8) << var << endl;
        cout << "dezimal: " << setbase(10) << var << endl;
}
```

```
Die Zahl 10 ist
hexadezimal: a
oktal: 12
dezimal: 10
```

Zahlensysteme

- Umwandlung einer Dualzahl in eine Dezimalzahl:
  $(010111)_2 = 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = (23)_{10}$

- Umwandlung einer Oktalzahl in eine Dezimalzahl:
  $(167)_8 = (1 \cdot 8^2 + 6 \cdot 8^1 + 7 \cdot 8^0) = (119)_{10}$

- Umwandlung einer Hexadezimalzahl in eine Dezimalzahl:
  $(C2)_{16} = (12 \cdot 16^1 + 2 \cdot 16^0) = (194)_{10}$

Gleitkomma Konstanten

- Der Standarddatentyp für Gleitkomma Konstanten ist double.
- Die Abspeicherung erfolgt im IEEE-Format (Institute of Electrical and Electronics Engineers) durch Verwendung: eines Vorzeichenbits, 11 Exponentenbits und 52 Mantissenbits.
- Die Genauigkeit der Darstellung ist auf ca. 15 Stellen begrenzt.

Gleitkomma Konstanten

- Beispiele für die Ein- und Ausgabe von Gleitkomma Konstanten (Gleitkommazahlen)

1.2345
-0.012
41.0
1.1E+02
-25.0E+15

Definition einer Variablen

- Bei der Definition einer Variablen wird der Variablen eine Speicherstelle zugewiesen. Diese kann beschrieben und ausgelesen werden.

```
int anzahl;
double kraft;
double volumen, radius;
```

Gültigkeitsbereich von Variablen

- Variablen können innerhalb und außerhalb von Blöcken deklariert werden.
- Wird eine Variable innerhalb eines Blocks deklariert, so hat sie nur hier Gültigkeit. Man nennt sie deshalb eine lokale Variable.
- Wird eine Variable außerhalb eines Blocks, d.h. außerhalb von Funktionsrümpfen deklariert, so hat sie von ihrer Position an Gültigkeit. Sie wird globale Variable genannt.

Initialisierung von Variablen

- Man kann Variablen auch definierte Anfangswerte zuordnen.
- Eine Variable sollte vor ihrer ersten Verwendung initialisiert werden.

```
int anzahl = 1000;
double PI = 3.14159;
int a = 2, b = 3;
```

Benannte Konstanten

- Konstanten kann ein Name zugeordnet werden.
- Wird bei der Initialisierung einer Variablen vor den Datentyp das Schlüsselwort *const* geschrieben, so handelt es sich um eine benannte Konstante.

```
const double PI = 3.14159;
```