3. Operatoren

3.1.Arithmetische Operatoren

Hier ist ein Beispiel für die Grundrechenoperationen, die C++ zur Verfügung stellt:

Beispiel 1

```
int a, b, c;

a = 6;
b = 4;
c = a + b;      //Addition
c = a - b;      //Subtraktion
c = a * b;      //Multiplikation
c = a / b;      //Division
c = a % b;      //Modulo-Operation
```

Achtung! Bei einer ganzzahligen Division ensteht auch ein gahzzahliges Ergebnis.

3.2.Logische Operatoren und Vergleichsoperatoren

Die logischen Operatoren liefern nur zwei mögliche Antworten: true oder false.

Bemerkung: In C++ wird ein Wert ungleich 0 als true und ein Wert gleich 0 als false interpretiert.

Tabelle 1

OPERATOR	**AUSDRUCK**
kleiner	ergebnis = (ausdruck 1) < (ausdruck 2)
grösser	ergebnis = (ausdruck 1) > (ausdruck 2)
gleich	ergebnis = (ausdruck 1) == (ausdruck 2)
kleiner-gleich	ergebnis = (ausdruck 1) <= (ausdruck 2)
grösser-gleich	ergebnis = (ausdruck 1) >= (ausdruck 2)
und	ergebnis = (ausdruck 1) && (ausdruck 2)
oder	ergebnis = (ausdruck 1) || (ausdruck 2)
ungleich	ergebnis = (ausdruck 1) != (ausdruck 2)

Achtung! Vergleichen Sie niemals Fliesskommazahlen auf Gleichheit!

3.3.Bitweise Operatoren

Man kann zwei Variablen auch bitweise vergleichen. Dabei wird die bearbeitete Zahl als Bitfolge (binäre Darstellung) interpretiert. Hier sind ein paar Beispiele:

Tabelle 2

OPERATOR	AUSDRUCK
und	ergebnis = (zahl 1) & (zahl 2)
oder	ergebnis = (zahl 1) | (zahl 2)
exklusiv-oder	ergebnis = (zahl 1) ^ (zahl 2)
negation	ergebnis = ~(zahl 1)
shift-left	ergebnis = (ganzzahl) << (bitanzahl)
shift-right	ergebnis = (ganzzahl) >> (bitanzahl)

Ausserdem existieren Kurzschreibweisen für die Operationen:

Tabelle 3

X &= Y	X = X&Y
X |= Y	X = X|Y
X ^= Y	X = X^Y
X <<= Y	X = X<<Y
X >>= Y	X = X>>Y

3.4.Kurzschreibweise

Tabelle 4

X++	Erhöht X um 1 nach dessen Auswertung
++X	Erhöht X um 1 vor dessen Auswertung
X - -	Vermindert X um 1 nach dessen Auswertung
- - X	Vermindert X um 1 vor dessen Auswertung
X += Y	Addiert Y zu X (X=X+Y)
X -= Y	Subtrahiert Y von X (X=X-Y)
X *= Y	Multipliziert X mit Y (X=X*Y)
X /= Y	Dividiert X durch Y (X=X/Y)
X %= Y	Weist X den Rest der Division X/Y zu (X=X%Y)

Und hier ein Beispiel:

Beispiel 2

```
X = 10;      // Ausgangszustand
Y = 2;       // der Variablen

X++;         // X = X+1 = 11

Y = X++;     // Zuerst Y = X = 11
             // und dann X = X+1 = 12

Y = ++X;     // Jetzt zuerst X = X+1 = 13
             // und dann Y = X = 13

X += Y;      // X = X+Y = 13+13 = 26

X %= 5;      // X = X%5 = 26%5 = 1
```

```
X += (--Y * 2); // Zuerst Y = Y-1 = 12
                // dann X = X+12*2 = 1+24
```

3.5.Rangfolge der Operatoren

Tabelle 5

Gruppe	Operator	Bedeutung	Auswertung
1	::	Zugriffsoperator	links->rechts
2	(...)	Klammer	links->rechts
	(...)	Funktionsaufurf	
	[...]	Indizierung	
	->	Indirekter Zugriff auf Klassenelement	
	.	Direkter Zugriff auf Klassenelement	
	++ und --	Post-Inkrement/Dekrement	
3	! und ~	NOT und 1er-Komplement	rechts->links
	+ und -	Vorzeichen	
	++ und --	Pre-Inkrement/Dekrement	
	&	Adressoperator	
	*	Dereferenzierungsoperator	
	sizeof(...)	Größenoperator	
	new und delete	dynamische Speicherverwaltung	
	( )	Typkonvertierung	
4	.* und ->*	Indirekter Zugriff auf Klassenelement	links->rechts
5	* und / und %	Arithmetische Operatoren	links->rechts
6	+ und -	Arithmetische Operatoren	links->rechts
7	<< und >>	Schiebeoperatoren	links->rechts
8	< und <= und > und >=	Vergleichsoperatoren	links->rechts
9	== und !=	Vergleichsoperatoren	links->rechts
10	&	UND Bitoperator	links->rechts
11	^	EXCLUSIV-ODER Bitoperator	links->rechts
12	|	ODER Bitoperator	links->rechts
13	&&	logischer UND Operator	links->rechts
14	||	logischer ODER Operator	links->rechts
15	?:	Bedingungsoperator	rechts->links
16	= und *= und /= und %= und += und -= und &= und ^= und |= und <<= und >>=	Zuweisungsoperatoren	rechts->links
17	throw	Exception auslösen	
18	,	Komma-Operator	links->rechts

3.6.Operatoren für die string-Klasse: Erweiterung

3.6.1. Vergleichen

Man kann zwei Strings mit den bekannten Vergelichsoperatoren ><, ==, != vergleichen. Der Vergleich erfolgt hierbei lexikalisch, d.h. '*Aaaa*' ist kleiner als '*Bb*'. Hier ein Beispiel:

Beispiel 3

```
// Zwei String definieren
string s1("Aaaa"), s2("Bb");
// Strings vergleichen
if (s1 < s2)
    cout << "s1 kommt vor s2" << endl;
else
    cout << "s2 kommt vor s1" << endl;
```

3.6.2. Indexoperator

Um die einzelnen Zeichen des Strings zu erreichen, verwendet man den Indexoperator []. Hier ein Beispiel:

Beispiel 4

```
// String definieren
string s1("Ein Text");
// Alle Zeichem im String einzeln ausgeben
for (int i=0; i<s1.size(); i++)
    cout << s1[i] << ',';
```

Die Methode *size()* liefert die Anzahl der Zeichen im string (die Länge des Strings – d.h. die absolute Länge; die Indices gehen dann von 0 bis *size()*-1).

Alle Operatoren werden in diesem Beispiel angewendet (Sortierung einer Liste mit Namen nach bubblesort):

Beispiel 5

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

// Funktion zum Tauschen von zwei strings
void swap(string *pa,string *pb)
{
   string tmp=*pa;
   *pa=*pb;
   *pb=tmp;
};

// Funktion zur Sortierung von einem array nach bubblesort
void bubbleSort(string *array,int size)
{
   for(int obergrenze=size-1;obergrenze>0;--obergrenze)
   {
      for(int pos=0;pos<obergrenze;++pos)
      {
         if(array[pos]>array[pos+1])
            swap(&array[pos],&array[pos+1]);
      };
   };
};
```

```
void main()
{
   int i;
   const int ANZAHL_NAMEN=5;
   string namen[ANZAHL_NAMEN];

   // Ausgabe der Aufforderung
   cout<<"Bitte geben Sie "<<ANZAHL_NAMEN<<" Namen ein!"<<endl;

   // Einlesen der Namen
   for(i=0;i<ANZAHL_NAMEN;++i)
      cin>>namen[i];

   // Sortieren der Namen mittels Bubblesort
   bubbleSort(namen,ANZAHL_NAMEN);

   // Ausgabe der sortierten Namen
   cout<<"Die Namen sortiert:"<<endl;

   for(i=0;i<ANZAHL_NAMEN;++i)
      cout<<namen[i]<<endl;
};
```

3.7.Aufgaben

3.7.1. Erweitern Sie den Taschenrechner vom Aufgabenblatt 1. Das neue Programm soll folgende Schritte ausführen:

- Eingabe der gewünschten Operation
- Eingabe des gewünschten Datentyps
- Eingabe der Operanden
- Rückgabe des Ergebnisses

Die zur Verfügung stehenden Typen sollen: *int*, *float* und *bool* sein. Die zur Verfügung stehenden Operation alle aus dem Kapitel sein, ausser die bitweisen Operationen.

3.7.2. Schreiben Sie ein kleines Tool, dass Strings verwaltet. Es soll als Eingabe zwei Strings bekommen und mittels Menü eine der folgenden Operationen machen:

- string 1 und string 2 addieren
- string 2 und string 1 addieren
- string 1 < string 2 berechnen
- string 1 > string 2 berechnen
- string 1 == string 2 berechnen

3.7.3. Schreiben Sie ein kleines Tool, dass bitfields verwaltet. Es soll als Eingabe zwei bitfelder (array von *true* und *false* mit 8 Feldern) bekommen und mittels Menü eine der bitweisen Operationen aus dem Kapitel ausführen.