Kontrollstrukturen

Die for-Schleife

Die for-Schleife ist eine „Zählschleife“, da eine int-Zahl (hier int i) jeden Durchlauf mitzählt.

```
cout << "\nAnton als ASCII-Strom" << endl;
string anton = "|||||||\n|     |\n| o o |\n|  L  |\n|  ~  |";

for(int i=0;i < anton.length();i++){
      cout <<  anton[i];
}
```

Versuche auch

```
cout << "\nAnton als ASCII-Strom" << endl;
string anton = "|||||||\n|     |\n| o o |\n|  L  |\n|  ~  |";

for(int i=0;i < anton.length();i++){
      cout <<  (int)anton[i];   // Typ int erzwingen
}
```

for rückwärts

Und wie wäre es mit ein wenig Sport?

```
cout << "\nAnton's Kopfstand" << endl;
string s = "|||||||\n|     |\n| o o |\n|  L  |\n|  ~  |";
for(int i=s.length()-1;i>=0;i--){
      cout <<  s[i];
}
```

Die while-Schleife

Syntax:

```
while (ausdruck)
{
    // Schleifenrumpf
}
```

Der Schleifenrumpf wird solange ausgeführt, wie der "ausdruck" wahr ist, d.h. solange er einen Wert ungleich 0 besitzt. Da der Ausdruck for dem Eintritt in die Schleife geprüft wird, wird die while-Schleife evtl. auch gar nicht ausgeführt.

`while (1) { ... }`
ist eine Endlosschleife, da der Ausdruck 1 stets wahr ist.

Beispiel: Aufsummieren

```
int i=1, sum=0, n;
while(i<=10)
{
    cin >> n;
    sum+=n;
    i++;
}
```

Die do-while-Schleife

Syntax:

```
do
{ ...
    // Schleifenrumpf
} while (ausdruck);
```

Der Schleifenrumpf wird auf jeden Fall einmal ausgeführt, alle weiteren Male nur, solange der Ausdruck einen Wert ungleich 0 hat.

Beachte:
Die while-Zeile wird mit ; abgeschlossen.

Übung

a) Welche Ausgabe erzeugt dieses Programmstück?

```
int i = 1;
do
{
   cout << i << "\t";
} while (i++ < 5);
```

b) Welche Ausgabe erzeugt `while (++i < 5)` ?

c) Abfrage am Programmende, ob nochmal oder nicht:

```
char c;
do
{
   /* eigentliches Programm */
   cout << "E --> Ende, andere Taste --> nochmal ...";
   cin >> c;
} while (c!='e' && c!='E');
```

d) Eingabefilter (Plausibilitätsprüfung):
Eine Eingabe soll solange wiederholt angefordert werden, bis ein gültiger Wert eingegeben wurde.

```
char a;
do
{
    cout << "geben Sie eine ganze Zahl von 0 bis 9 ein -->";
    cin >> a;
} while ((a<48) || (a>57));    // ASCII 48 - 57 ist 0..9
```

Logische Operatoren:

`||`	logisches ODER
`&&`	logisches UND
!	logisches NICHT

Die Verzweigung mit if

Syntax:

```
if(ausdruck)
{
    /* Anweisungsteil1 */
}
else
{
    /* Anweisungsteil2 */
}
```

Anweisungsteil1 wird ausgeführt, falls der Ausdruck nach if einen Wert ungleich 0 (logisch wahr) liefert, falls er 0 ergibt, wird Anweisungsteil2 durchlaufen. Besteht ein Anweisungsteil aus nur einem Befehl, so können die Klammern {} weggelassen werden. Der else-Zweig darf fehlen (einseitige Bedingung).

Beispiele:

```
if (a >= b)
    max = a;
else
    max = b;
```

`if`-Anweisungen können geschachtelt werden. Fehlen dabei die Klammern, gehört ein `else` stets zum vorhergehenden `if`.

```
if (a < 0)
    cout << "Negativ";
else if (a == 0)
    cout << "Null";
else
    cout  << "Positiv";
```

Warnung:

Häufig wird in einem Vergleich auf Gleichheit statt dem Vergleichsoperator `"=="` versehentlich der Zuweisungsoperator `"="` geschrieben.

```
x = 0;
if (x = 1) cout << "Eins"; /* V O R S I C H T !!! */
```

Dieses Programmstück ist syntaktisch korrekt! C++ führt die Zuweisung `x = 1` aus, der Wert des Ausdruckes ist bei einer Zuweisung gleich dem Wert der rechten Seite, hier also 1, und dies ist logisch wahr, also wird der cout-Befehl ausgeführt, und das ist natürlich etwas völlig anderes als ursprünglich beabsichtigt war.
Der C++-Compiler gibt hier keine Fehlermeldung, vielleicht aber eine Warnung aus, ob wirklich Zuweisung statt Vergleich gemeint war.

Die Sprunganweisungen break und continue

Mit dem Schlüsselwort **`break`** wird die innerste Schleife vorzeitig verlassen, bei dem case-Statement (s.u.) zum nächsten case gesprungen.

```
if(found)
{
      break;
}
```

`continue` sorgt dafür, dass der **Rest der innersten Schleife übersprungen** und gleich die Bedingung für weiteren Schleifendurchlauf geprüft wird. Es wird relativ selten verwendet.

```
for (i=0; i<10; i++)
{
      if (!zaehlen) continue;
      count++;
}
```

`break` und `continue` zählen zu den erlaubten Elementen eines guten Programmierstiles, die Verwendung des `goto`-Befehls dagegen ist zwar möglich, aber umstritten. Prinzipiell kann jeder Algorithmus ohne goto formuliert werden.

Die Mehrfachverzweigung mit switch

Syntax:

```
switch (ausdruck)
{
    case wert1: anweisungsteil1;break;
    case wert2: anweisungsteil2;break;
.........................
    case wertn: anweisungsteiln;break;
    default: defaultanweisung;
}
```

Der Ausdruck muß leider von einem ganzzahligen Typ sein, es wird der case-Zweig mit dem passenden Wert angesprungen, falls kein Wert passt, wird der (optionale) default-Zweig ausgeführt. Um die einzelnen Anweisungsteile müssen auch bei mehreren Anweisungen keine Klammern {} gesetzt werden.

Ein String als Zahlenfolge

Im Computer sind Texte und Zeichen auch nur Zahlen. Anton ist demnach eine Zahlenfolge:
124 124 124 124 124 124 124 10 124 32 32 32 32 32 124 10 124 32 111 32 111 32 124 10 124 32 32 76 32 32 124 10 124 32 32 126 32 32 124
Und welche Information steckt in dieser Zahlenfolge?

65 114 109 101 114 32 65 110 116 111 110

— — — — — — — — — — —

Anton löst ein Rätsel

Vom Reis auf dem Schachbrett

Ein Wesir erfand das Schachspiel für seinen Herrscher, den Sultan. Dieser war so erfreut, dass er dem Wesir jeden Wunsch erfüllen würde. Der Wesir dachte nach und sagte: "Eure Majestät - lassen Sie ein Reiskorn auf das erste Feld des Schachbrettes, zwei Reiskörner auf das Zweite, vier Reiskörner auf das Dritte usw. legen. Geben Sie mir jene Reismenge, die auf dem letzten Feld liegt." Wie viele Reiskörner hätte der Wesir bekommen?

Anton ist schlau. Er weiß, dass auf dem ersten Feld 2^0, auf dem 2. Feld 2^1, auf dem letzten Feld 2^{63} Reiskörner liegen müssten. Das geht noch mit dem unsigned long Datentyp.

Anton wird gebeamt

```
string anton = "|||||||\n|     |\n| o o |\n|  L  |\n|  ~  |";

cout << "\nAnton gebeamt" << "\n";

for(int i=0;i < anton.length();i++)
{
 int a = (int)anton[i];
 string str;
 for(int k=0; k<8; k++)
 {
   int h = a>>k;
   if(h%2==0)
   {
     str += '0';
   }
   else
   {
     str += '1';
   }
 }

 string verdreht;
 for(int k=7;k>=0;k--)
   verdreht+=str[k];
 cout << verdreht.c_str();
}
```

```
01111100011111000111110001111100011111000111110001111100000001010011111000010000
00100000001000000010000000100000011111000000010100111110000100000011010111100100000
01101111001000000111110000001010011111100001000000010000000100110000100000000100000
01111100000001010011111000010000000100000011111100010000000100000011111100
```

Transistorwiderstand

Den Gesamtlastwiderstand Z eines Transistors berechnet man mit der Formel $Z = \frac{R_C \cdot R_L}{R_C + R_L}$.

Ein Programm soll nach der Eingabe des Kollektorwiderstands R_C und des Lastwiderstands R_L den Wert Z ausgeben.

Technische Hinweise:

Berechne zuerst $h = R_C + R_L$ und dann $Z = \frac{R_C \cdot R_L}{h}$.

Testdaten

Berechne Z, wenn der Kollektorwiderstand R_C=156 Ohm und der Lastwiderstand R_L= 96.7 Ohm ist.
Geben Sie die Ergebnisse auf 6 Stellen nach dem Koma genau an.

Lösung:

```
double rc = 156, rl = 96.7;
h = rc + rl;
double z = rc * rl / h;
```

Lösungen einer quadratischen Gleichung

Die "Mitternachtsformel" ist wohl jedem Schüler in Erinnerung. Mit dieser Formel können die Lösungen einer quadratischen Gleichung $ax^2 + bx + c = 0$ berechnet werden.

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \quad und \quad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

Schreibe ein Programm, das die Lösungen, falls vorhanden, berechnet:

Technische Hinweise:

Die Quadratwurzel berechnet die Formel Math.`Sqrt( )`

Ein negativer Wert unter der Quadratwurzel erzeugt einen Laufzeitfehler.
Deshalb soll das Ergebnis von $d = b^2 - 4ac$ berechnet werden und dann eine Fallunterscheidung für die 3 Möglichkeiten (d == 0) , (d > 0) oder (d < 0) gemacht werden. Verwende if(d == 0) {} else if(d > 0) { } else {}

Die Quadratwurzel berechnet die Funktion

C++
#include <Math.h> sqrt(d)

Testdaten

Schreibe ein Programm, das die Lösungen der folgenden Gleichungen berechnet:
$5x^2 + 19x + 14 = 0$, $2x^2 - 3x + 6 = 0$, $10x^2 + 2x + 15 = 0$

$5x^2 + 19x + 14 = 0$ 1. Lösung: -1.0 2. Lösung: -2.8
$2x^2 - 3x + 6 = 0$
$10x^2 + 2x + 15 = 0$

Schrittweise die Differenz bilden

Ein Auto kostet heute 40000 Euro. Es verliert aber jedes Jahr 15% des Wertes.
Wieviel ist es nach 10 Jahren noch wert?

Wieviel ist es nach 10 Jahren noch wert, wenn Sie im 3. Jahr einen Auffahrunfall haben, der zusätzlich 50% Wertverlust bedeutet?

Wieviel wäre das Auto nach 10 Jahren wert, wenn der Unfall erst im 9. Jahr ist?

Schrittweise die Summe bilden: while

Lass Dein Programm rechnen:
1+2+3+4+ … +99+100
Wie lautet das Ergebnis?

oder

$$1+\frac{1}{2^2}+\frac{1}{3^2}+\frac{1}{4^2}+\frac{1}{5^2}+\frac{1}{6^2}+..$$

Wie lautet das Ergebnis? Was ergibt sich wenn die Zahl im Nenner bis 100 oder bis 10000 geht?
Beachte: Wenn in einer Formel mit Komma-Ergebnis ganze Zahlen verwendet werden, ist "casting" angesagt.

Parallelschwingkreis

Die Frequenz eines Parallel-Schwingkreises berechnet sich mit der Formel $f = \frac{1}{2\pi\sqrt{L \cdot C}}$

Die Induktivität der Spule L und die Kapazität des Kondensators C sind angegeben. Das Programm berechnet f.

Technischer Hinweis:

Verwende den Datentyp double für alle Variablen.

Testdaten

C = 240 Pico Farad (1 Farad = 1 Ampere*Sekunde/Volt)
L = 0,4 Milli Henry (1 Henry = 1 Volt*Sekunde/Ampere)
Das Ergebnis liefert eine Frequenz in Kilo Hertz Bereich.

```
#include  <math.h>


double f, l, c;
l = 0.4/1000;     // 0.4 Milli Henry
c = 0.24/1000000000; // 240 Pico Farad

double h = 2 * M_PI * sqrt(l * c);
// Hilfsvariable h

f = 1.0 / h;
cout << "Frequenz f=" << f/1000 << " Kilo Hertz";
```

Zins

Sie legen bei einer Bank 1000€ zu einem Zinssatz von 5% an.

a) Wie groß ist Ihr Kapital nach 12 Jahren?
b) Nach wie vielen Jahren hat sich Ihr Kapital mehr als verdoppelt?

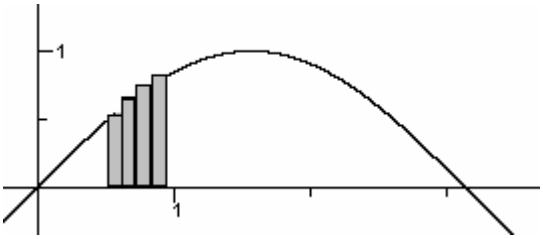
```
int i;
for(i=0;i<12;i++)
{
    kapital = kapital*1.05;    // Das Kapital wird 12 mal um 5% erhöht.
}
```

Berechnung einer Umdrehungszahl

Ein Gewicht von 2 g wird an einer Schnur von 25 cm geschleudert. Bei welcher Umdrehungszahl reißt die Schnur, wenn sie 50 Newton (entspricht 5 kg Gewicht) aushält?
Die Reißkraft ist Masse * Radius * (2 * PI * Frequenz)² , wenn die Werte in kg, m, s eingesetzt werden.

Flächenberechnung

Mathe



Die Fläche unter der Sinuskurve wird in 100000 Streifen mit der Breite PI/100000 zerlegt. Die Streifenhöhe des n-ten Streifens ist sin(n*PI/100000).

Berechne den Flächeninhalt. (Das exakte Ergebnis ist 2.0)

Kopfrechnen - Wettbewerb

Der Computer zeigt eine Rechenaufgabe: Zum Beispiel 7 * 8.
Vom Benutzer wird das Ergebnis eingegeben. Ist es richtig, geht der Computer zur nächsten Aufgabe. Ist es falsch wird solange wiederholt, bis das Ergebnis richtig ist. Das Programm zählt die richtigen Ergebnisse.
Nach 20 Sekunden ist Schluss und der Benutzer erhält die Meldung, wie viel Rechnungen er geschafft hat: Z.B.
5 Aufgaben in 20 Sekunden geschafft.

Ein Näherungswert für die natürliche Zahl e

Die Zahl e ist der Grenzwert von $(1+\frac{1}{n})^n$ für sehr große Zahlen n. Berechne e auf 8 Stellen nach dem Komma genau. Wie groß muss dann n sein?

Ein Optimierungsproblem

Für die Verpackung eines Weihnachtsgeschenks basteln Sie eine Schachtel.
Sie verwenden ein rechteckiges Stück Pappe mit der Länge 30 cm und der Breite 20 cm.
An den Ecken werden Quadrate mit der Kantenlänge x cm ausgeschnitten.
Der Rest wird zu einer Schachtel mit der Höhe x hochgebogen.
Schreiben Sie ein Programm, das das Schachtelvolumen für 0 < x < 10 in kleinen Schritten (0.1) berechnet.
Wie groß muss x sein, damit die Schachtel das größtmögliche Volumen erhält und am meisten Weihnachtsplätzchen darin Platz finden?