

# Grundlagen der Programmierung in C++

## Kontrollstrukturen

Wintersemester 2005/2006  
 G. Zachmann  
 Clausthal University, Germany  
[zach@in.tu-clausthal.de](mailto:zach@in.tu-clausthal.de)

## Block

- Keine Kontrollstruktur im eigentlichen Sinn
- Dient zur Zusammenfassung mehrerer Anweisungen
- Bsp.:
 

```
{
  a = 1;
  b = a*a;
}
```
- Überall, wo eine Anweisung stehen kann, kann auch ein Block stehen
- Wird fast ausschließlich für Kontrollstrukturen gebraucht
- Kann man schachteln ("innerer" und "äußerer" Block)

## If

- Einfachstes Konstrukt zum Kontrollfluß
- Definition:
 

```
if ( boolean-expr )
  statement;
```

 oder
 

```
if ( boolean-expr )
  statement1;
else
  statement2;
```
- Mehr braucht man eigtl. nicht

## Beliebte Fallen:

```
if ( i == 1 ) // this is always true !!
{
  ...
}
```

```
if ( i == 0 )
  if ( a < 0 )
  {
    ...
  }
else
{
  ... // will be executed only
  // if a < 0 !!
}
```

## Geschachtelte If's

- Anweisung(en) innerhalb `if` oder `else` können wieder `if`'s enthalten
- "Inneres" und "äußeres" If
- Folge von Tests:


```
if ( punkte >= 90 )
  note = 'A';
else if ( punkte >= 85 )
  note = 'B';
else if ( punkte >= 70 )
  note = 'C';
else if ( punkte >= 50 )
  note = 'D';
else
  note = 'E';
```

## Switch

- Definition:
 

```
switch ( selector )
{
  case value1 :
    statement-list 1 ;
    break;
  case value2 :
    statement-list 2 ;
    break;
  default:
    statement-list n
}
```
- Selector muß Integer-Expression sein

- Achtung: **break** nicht vergessen!
- Sonst "fall-through"
- Ist ein "Feature" (ich behaupte: Bug)
- Zusammenfassung von Cases: **case 1: case 2:**



G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 7

Beispiel:

```
// int month, numDays;
switch ( month )
{
  case 1: case 3:
  case 5: case 7:
  case 8: case 10:
  case 12:      numDays = 31;
                break;
  case 4: case 6:
  case 9: case 11: numDays = 30;
                break;
  case 2:      if ( ((year % 4 == 0) && !(year % 100 == 0))
                || (year % 400 == 0) )
                numDays = 29;
                else
                numDays = 28;
                break;
  default:    fprintf( stderr, "BUG: month = %d !\n", month );
}

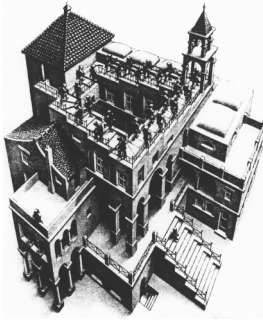
```

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 8

### Schleifen

"Life is just one damn thing after another."  
-- Mark Twain

"Life isn't just one damn thing after another ... it's the same damn thing over and over and over again."  
-- Edna St. Vincent Millay



G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 9

### While-Schleife

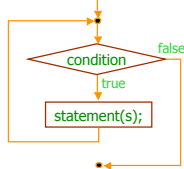
Definition (Syntax & Semantik):

```
while ( condition )
  statement ;
```

Beispiele:

```
// int b;
int a = 1;
while ( a < b )
{
  a *= 2;
}

```



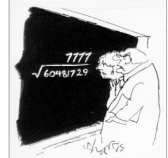
G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 10

### Beispiel: Quadratwurzeln (Newton-Raphson)

- Ziel: Berechnung der Quadratwurzel einer Floatingpoint-Zahl  $c$
- Initialisiere  $t = c$
- Ersetze  $t$  durch den Mittelwert von  $t$  und  $c/t$
- Wiederhole dies, bis  $t=c$

```
c = 2.0;
t = c;
while ( t - c/t > 0.0000000001 )
  t = ( c/t + t ) / 2.0;
printf("%f\n", t );

```



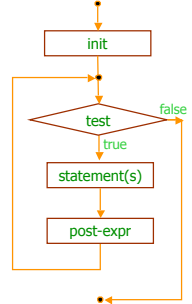
"A wonderful square root. Let's hope it can be used for the good of mankind."

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 11

### For-Schleife

- while** genügt eigtl., **for** ist aber praktisch
- Definition:
 

```
for ( init-expr; test-expr; post-expr )
  statement ;
```
- Test-expr** muß **bool** liefern
- In den **expr**'s ist alles erlaubt, was in normalen Ausdrücken erlaubt ist:
  - Operatoren
  - Zuweisungen
  - Variablen-Deklarationen ...



G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 13

Beispiele:

```

// float q; unsigned int n;
float s = 0;
// s = geom. Reihe 1 + q + q^2 + q^3 + ... + q^n
float qq = 1;
for ( uint i = 0; i < n; i ++ )
{
    s += qq;
    qq *= q;
}

```

Heißt "Schleifenvariable" (loop variable)

Loop body (Schleifenrumpf)

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 14

For-Deklaration kann auch etwas komplexer sein:

```

for ( uint i = 0, j = 17; i ++, j ++ )
{
    ...
}

```

```

#include <stdio.h>
int main(void)
{
    int count;
    for (count = 1; count <= 500; count++)
        printf("I will not throw paper airplanes in class.");
    return 0;
}

```

NICE TRY.

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 15

Scope von Deklarationen innerhalb des for-init:

- Nur For-Klammer und Rumpf der For-Schleife (action)!
- Unbedingt nutzen! (natürlich nicht default in MS-VS-Studio ☹)
- Hat mir schon einige Stunden Bug-Suche erspart!
- Unterstützt das Prinzip der Daten-Lokalität bzw. des "information hiding"

```

for ( uint i = 0; ... )
{
    ...
}
// ab hier ist i wieder unbekannt!

```

Scope von i

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 16

Geschachtelte Schleifen (nested loops)

Schleifenrumpf kann wieder Schleife enthalten:

```

for ( uint i = 0; ... )
{
    for ( uint j = 0; ... )
    {
        ...
    }
}

```

Andere Schleifenvariable nehmen!

Beispiel:

```

for ( unsigned int i = 0; i < 5; i ++ )
{
    for ( unsigned int j = 0; j < i; j ++ )
    {
        putchar('*');
    }
    putchar('\n');
}

```

Ausgabe

```

*
**
***
****
*****

```

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 17

Äquivalenz von for und while

```

for ( init ; test ; expr )
    action ;

```

```

init ;
while ( test )
{
    action ;
    expr ;
}

```

Init und test werden also immer mind. 1x ausgeführt

Expr und action werden nicht notw. ausgeführt!

Wann for und wann while?

- For wenn Schleife über Integer-Bereich mit festen Grenzen
- While wenn Anzahl Durchläufe nicht klar (max. Anz. begrenzen!)

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 20

Beispiel: Mittelwert

```

uint list_size = 3;
uint n_values = 0;
float sum = 0.0;
while ( n_values < list_size )
{
    float value;
    scanf("%f", &value);
    sum += value;
    n_values ++;
}
float average = sum / n_values;
printf("Average = %f\n", average);

```

Angenommen, wir wüssten das

Schleifen in C laufen fast immer von 0 ... n-1

Lokale Variable innerhalb der Schleife

float/int: int wird konvertiert zu float

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 22

Eingabe: 1 5 3

```

uint list_size = 3;
uint n_values = 0;
float sum = 0.0;
while ( n_values < list_size )
{
    float value;
    scanf("%f", &value );
    sum += value;
    n_values ++ ;
}
float average = sum / n_values;
printf( "Average = %f\n", average );

```

list_size	3
n_values	1
sum	1.0
average	3.0

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 23

### Exkurs: korrekte Programme durch vollständige Fallunterscheidung

```

uint list_size = 3;
uint n_values = 0;
float sum = 0.0;
while ( n_values < list_size )
{
    float value;
    scanf("%f", &value );
    sum += value;
    n_values ++ ;
}
float average = sum / n_values;
printf( "Average = %f\n", average );

```

Kennt man i.A. nicht!

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 24

```

float value;
uint n_values = 0;
float sum = 0.0;
while ( scanf("%f", &value) == 1 )
{
    sum += value;
    n_values ++ ;
}
float average = sum / n_values;
printf( "Average = %f\n", average );

```

Diese Funktion liefert Anzahl erfolgreich konvertierter Eingaben

Neues Problem: was, wenn 0 Werte gelesen wurden?!

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 25

```

float value;
uint n_values = 0;
float sum = 0.0;
while ( scanf("%f", &value) == 1 )
{
    sum += value;
    n_values ++ ;
}
if ( n_values > 0 )
{
    float average = sum / n_values;
    printf( "Average = %f\n", average );
}
else
{
    fprintf( stderr, "No values on stdin!\n" );
}

```

Noch ein Problem: was, wenn der input stream gar nicht mehr aufhört?!

Oder Fehlercode zurückgeben, oder Exception werfen, oder ...

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 26

```

const uint max_n_values = 0;
float value;
uint n_values = 0;
float sum = 0.0;
while ( scanf("%f", &value) == 1 && n_values < max_n_values )
{
    sum += value;
    n_values ++ ;
}
if ( n_values > 0 )
{
    float average = sum / n_values;
    printf( "Average = %f\n", average );
}
else
{
    fprintf( stderr, "No values on stdin!\n" );
}

```

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 27

### Break und continue

- Problem: zusätzliche Sprünge innerhalb der Schleife
- Zurück an Anfang
- Aus Schleife raus

```

bool go_on = true;
while ( (...) && go_on )
{
    if ( ... )
        go_on = false;
    else
    {
        if ( ... )
        {
            if ( ... )
                go_on = false;
            else
                go_on = false;
        }
    }
}

```

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 28

## Lösung

- Zusätzliche Flußkontrolle innerhalb Schleifen
- Können nur innerhalb Schleifen verwendet werden
- break**: springt aus Schleife heraus (nur 1 Level!)
- continue**: springt ans Ende des Schleifenblocks

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 29

Beispiel von vorhin:

```
while (...)
{
    if (...)
        break;
    if (...)
        continue;
    if (...)
        break;
    // continue jumps here
} // break jumps here
```

```
bool go_on = true;
while ( (...) && go_on )
{
    if (...)
        go_on = false;
    else
    {
        if (...)
        {
            if (...)
                go_on = false;
        }
        else
            go_on = false;
    }
}
```

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 30

## Exkurs: Idiome

- Jede Sprache hat ihre eigenen Idiome, so auch jede Programmiersprache
- Man sollte diese Idiome unbedingt verwenden

Richtig	Nicht C++-like
<pre>i ++ ;  for ( i=0; i &lt; n; i ++ )     ...  if ( a &amp;&amp; !b )  printf("Resultat: %.3f = %d%%\n",        x, i );</pre>	<pre>i += 1; // oder ... i = i + 1;  i = 0; while ( i &lt; n ) {     ...     i ++ ; }  if ( a == true &amp;&amp; b==false )  cout &lt;&lt; "Resultat: " &lt;&lt; setprec(3) &lt;&lt; x &lt;&lt; i;</pre>

Hier streiten sich die Geister

G. Zachmann Grundlagen der Programmierung in C - WS 05/06 Kontrollstrukturen, 31