

---

# Programmeermethoden

Controle-structuren

week 3: 19–23 september 2011

<http://www.liacs.nl/home/kosters/pm/>

Inleveren: digitaal iets.cc en print van mooi.pdf.

```
\begin{document}
\usepackage{listings}
% ... etcetera ...
```

mooi.tex

```
#include <iostream>
using namespace std;
// ... etcetera ...
```

iets.cc

→  
L<sup>A</sup>T<sub>E</sub>X

```
Mooi printen
Walter Kesters
15 september 2010

1 Uitleg
Hoe print je een C++ programma mooi? Bijvoorbeeld met LATEX-pakage listings. Let op de talloze opties, bijvoorbeeld voor de tab-grootte.

2 Code
Een voorbeeld:
1
2 // file iets.cc
3 // Dit is een simpel C++-programma, hello world
4
5 #include <iostream>
6 using namespace std;
7
8 const double pi = 3.14159; // een constante (of ceath)
9 int main ( ) {
10     double straal; // straal van de cirkel
11     cout << "Geef straal, daarna Enter .. ";
12     cin >> straal;
13     if ( straal > 0 )
14         cout << "Oppervlakte "
15             << pi * straal * straal << endl;
16     else
17         cout << "Niet zo negatief ..." << endl;
18     cout << "Einde van dit programma." << endl;
19     return 0;
20 } //main
```

mooi.pdf

Zie de uitleg bij het derde werkcollege.

C++ kent de volgende controle-structuren:

**keuze** `if` (met als variant: `switch`)

**onbekend (maar eindig?) aantal herhalingen**

`while` (met als variant `do ... while`)

**“vast” aantal herhalingen** `for`

We gebruiken geen labels/goto's!

Een if-statement gaat informeel als volgt:

```
als ( een of andere test )  
    als de test waar is: zus en zo  
anders  
    als de test onwaar is: dit en dat
```

En een while-statement gaat informeel als volgt:

```
zolang ( een of andere test waar is )  
    zus en zo
```

```
if ( temperatuur > 0 )
    fietsen (...);
else {
    parapluikopen (...);
    lopen (...);
} //else
```

als test  $\neq$  0

als test = 0

Let op de accolades om het “compound” (samengestelde) statement. Rond `fietsen (...);` *mogen* accolades.

De tests gebruiken **predicaten** als `==` (*gelijk?*), `!=` (*ongelijk?*), `<` (*kleiner dan?*), `>` (*groter dan?*), `<=` (*kleiner dan of gelijk aan?*) en `>=` (*groter dan of gelijk aan?*).

Waar hoort een “else” bij?

```
if ( x > 0 )
  if ( y > 0 )
    cout << "Beide groter dan nul.";
else
    // waar hoort deze bij?
    cout << "      ? ? ? ?      ";
```

Waar hoort een “else” bij?

```
if ( x > 0 )
    if ( y > 0 )
        cout << "Beide groter dan nul.";
    else
        // waar hoort deze bij?
        cout << " x positief, y negatief (of 0) ";
```

Bij de laatste nog “openstaande” if!

Zorg ervoor dat de layout klopt — de compiler kijkt daar niet naar.

```
if ( x == 0 ) x = 1;  
else if ( x == 2 ) x = 7;
```

OK

```
if ( x == 0 )  
    x = 1;  
else if ( x == 2 )  
    x = 7;
```

OK++

```
if ( x == 0 )  
    x = 1;  
else  
    if ( x == 2 )  
        x = 7;
```

OK

```
if ( x == 0 ) x = 1;  
else if ( x == 2 ) x = 7;
```

OK (zie ook switch)

```
if ( x == 0 ) x = 42;  
if ( x == 2 ) x = 7;
```

// wat als hier x = 2;?  
zwak (x wordt twee  
maal lastig gevallen  
als x toevallig 0 is)

```
if ( x == 0 ) {  
    cout << "...iets..." << endl;  
    return 1;  
} //if  
if ( x == 2 ) x = 7;
```

// geen else nodig  
OK

```
if ( code == 0 ) // code van type int
    doedit (...);
else if ( code == 1 )
    doedat (...);
else ...
```

is equivalent met:

```
switch ( code ) {
    case 0: doedit (...); break;
    case 1: doedat (...); break;
    ...
} //switch
```

Je kunt ook als geval default: benutten.

Typisch gebruik: menu-opties, met een char.

Stel je wilt de eerste  $n$  positieve gehele getallen en hun kwadraten afdrukken:

```
int i, n;  cin >> n;                                1--1
                                                    2--4
                                                    3--9
i = 1;                                              4--16
while ( i <= n ) {                                  5--25
    cout << i << "--" << i * i << endl;            6--36
    i++;
}//while                                           ...
```

Het kan ook zo:

```
for ( i = 1; i <= n; i++ )
    cout << i << "--" << i * i << endl;
```

Nog enkele voorbeelden van **while-loops**:

```
while ( n != 0 )    // zolang n niet 0 is
    n = n - 1;      // laag hem met 1 af
```

Dit is overigens hetzelfde als `while ( n ) n--; !?`

Let er op dat je beter `( n > 0 )` als test kunt gebruiken:

```
x = 1;
while ( x < 100 ) { cout << x << endl; x = x + 2; }
```

drukt de oneven getallen  $< 100$  af, maar niet als de test `( x != 100 )` zou luiden — dan stopt het niet.

Bij een while-loop is het aantal “doorgangen” van te voren vaak onbekend of lastig te bepalen:

```
int x = 1;
while ( x < 1000 ) x = 2 * x;
// nu is x gelijk aan 1024
```

Het **Collatz-probleem** ( $3x + 1$  vermoeden) zegt dat het volgende programma voor ieder positieve gehele  $x$  stopt:

```
while ( x != 1 ) // 13->40->20->10->5->16->8->4->2->1
    if ( x % 2 == 0 ) // is x even?
        x = x / 2;
    else
        x = 3 * x + 1;
```

*A/*s dit stopt, is  $x$  na afloop gelijk aan 1.

Na

```
while ( x > 1 ) x = x / 2;      // of x /= 2;
```

geldt not (  $x > 1$  ), oftewel ! (  $x > 1$  ), dus (  $x \leq 1$  ).

En na

```
while ( ! ( x % 2 ) ) x = x / 2;
```

zijn alle factoren 2 uit de “oude”  $x$  gehaald;  $x$  is nu oneven.

Doe de test liever als: (  $x \% 2 == 0$  ).

Bij een **for-loop** als

```
for ( i = 3; i <= 17; i = i + 2 ) cout << i << "-";
```

wordt eerst (één maal) de initialisatie `i = 3;` gedaan, en daarna herhaald de cyclus test (geldt `i <= 17?`), statement (body) `cout << i << "-";` en teller-aanpassing `i = i + 2.`

We krijgen uitvoer 3-5-7-9-11-13-15-17-, en na afloop heeft `i` de waarde 19.

Opgave: probeer het streepje na 17 kwijt te raken.

Een for-loop kan in principe hetzelfde als een while-loop: “for (A B C) D” komt overeen met “A while (B) { D C }”. Bij een for-loop is het aantal “doorgangen” vaak bekend.

Een rare for-loop:

```
for ( ; ! ( x % 2 ); x /= 2 ) ;
```

Dit is hetzelfde als

```
while ( ! ( x % 2 ) ) x = x / 2;
```

En wat te denken van `for ( ; ; ) ;`? Dat is een **oneindige loop**, met een “empty statement”, net als `while ( true ) ;`.

Een **dubbele for-loop**:

```
int i, j;
for ( i = 1; i <= 5; i++ ) { // buitenste loop
    cout << i << ": ";
    for ( j = 1; j <= i; j++ ) // binnenste loop
        cout << i * j << " ";
    cout << endl;
} //for
```

geeft:

```
1: 1
2: 2 4
3: 3 6 9
4: 4 8 12 16
5: 5 10 15 20 25
```

Het **do-while statement** is een variant op de while-loop, waarbij de “body” in ieder geval één maal wordt uitgevoerd:

```
do {
    cout << "Geef positief getal .. ";
    cin >> getal;
} while ( getal < 0 );

do
    cin >> kar;
while ( ( 'a' <= kar && kar <= 'z' )
        || ( 'A' <= kar && kar <= 'Z' ) );
// nu bevat kar de eerste "niet-letter"
```

NB Pas op met `cin >> ...`, dit slaat whitespace over.

- maak de eerste programmeeropgave — de deadline is op vrijdag 23 september 2011
- lees de tweede programmeeropgave:  
`http://www.liacs.nl/home/kosters/pm/op2pm.php`
- lees Savitch Hoofdstuk 2
- lees dictaat Hoofdstuk 3, tot en met 3.5, en 3.7
- maak opgaven 6/10 uit het opgavendictaat