Academie voor Technology, Innovation & Society Delft
Academie voor ICT & Media

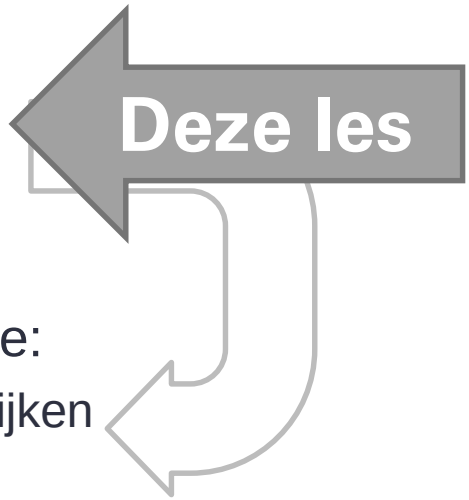
Gestructureerd programmeren in C

GESPRG: Vergelijken & herhalen

DE HAAGSE HOGESCHOOL

Programmeren

- Welke basisbewerkingen zijn er?
  - Lezen en schrijven (invoer en uitvoer)
  - Onthouden (variabelen)
  - Rekenen
  - Herhalen
  - Beslissen



Subcategorie:

- Vergelijken

DE HAAGSE HOGESCHOOL

Vergelijken

- Relationele operatoren:

Relationele operator	Betekenis	Teken in de wiskunde
>	Groter dan	>
<	Kleiner dan	<
>=	Groter dan of gelijk aan	≥
<=	Kleiner dan of gelijk aan	≤
==	Is gelijk aan	=
!=	Ongelijk aan	≠

Resultaat is een int (waar ➔ 1, onwaar ➔ 0)

Let op verschil in C tussen = en ==

DE HAAGSE HOGESCHOOL

Vergelijken met relationele operatoren

- OPDRACHT: pak pen en papier en schrijf voor elke onderstaande *printf()*, waarin een relationele vergelijking staat, wat de uitvoer moet zijn.

Code:

```
#include <stdio.h>

int main(void) {
 char a = 'a';
 char b = 'b';

 printf("VERGELIJKEN MAAR! \n");
 printf("5 > 3    -> %d \n", 5 > 3);
 printf("3 > 5    -> %d \n", 3 > 5);

 printf("5 >= 5   -> %d \n", 5 >= 5);
 printf("5 >= 3   -> %d \n", 5 >= 3 );
 printf("3 >= 5   -> %d \n", 3 >= 5);

 printf("5 == 5   -> %d \n", 5 == 5);
 printf("5 == 3   -> %d \n", 5 == 3);
 printf("a == a   -> %d \n", a == a);
 printf("b == a   -> %d \n", b == a);

 printf("5 != 5   -> %d \n", 5 != 5
);
 printf("5 != 3   -> %d \n", 5 != 3);
 printf("a != a   -> %d \n", a != a);
 printf("b != a   -> %d \n", b != a);

 fflush(stdin); getchar(); return 0;
}
```

Uitvoer:

```
5 > 3   -> 1
3 > 5   -> 0
5 >= 5  -> 1
5 >= 3  -> 1
3 >= 5  -> 0
5 == 5  -> 1
5 == 3  -> 0
a == a  -> 1
b == a  -> 0
5 != 5  -> 0
5 != 3  -> 1
a != a  -> 0
b != a  -> 1
```

Herhalen, herhalen, herhalen, he..

DE HAAGSE HOGESCHOOL

Herhalen

- Er zijn in C **3** herhalingsopdrachten
  - `for`
  - `do while`
  - `while`

DE HAAGSE HOGESCHOOL

for

- Gebruik een **for** als het **aantal** herhalingen bij het programmeren “bekend” is.

```
#include <stdio.h>

int main(void) {
    int i;
    for (i = 1; i != 10; i = i + 1) {
        printf("hallo %d\n", i);
    }
    getchar();
    return 0;
}
```

Zolang de voorwaarde WAAR is

initialisatie

doe telkens aan einde

DE HAAGSE HOGESCHOOL

Uitvoer van een for-lus

Code:

```c
#include <stdio.h>

int main(void) {
    int i;
    for (i = 1; i != 10; i = i + 1) {
        printf("hallo %d\n", i);
    }
    getchar();
    return 0;
}
```

Uitvoer:

```
hallo 1
hallo 2
hallo 3
hallo 4
hallo 5
hallo 6
hallo 7
hallo 8
hallo 9
```

Uitvoer van een for-lus

- Gaat het nu nog goed?

Code:

```c
#include <stdio.h>

int main(void) {
    int i;
    for (i = 1; i != 10; i = i + 2) {
        printf("hallo %d\n", i);
    }
    getchar();
    return 0;
}
```

Uitvoer:

```
hallo 1
hallo 3
hallo 5
hallo 7
hallo 9
hallo 11
hallo 13
hallo 15
hallo 17
…
```

Alternatieve voorwaarde

```c
#include <stdio.h>

int main(void) {
    int i;
    for (i = 1; i < 10; i = i + 1) {
        printf("hallo %d\n", i);
    }
    getchar();
    return 0;
}
```

Is dit beter ?

DE HAAGSE HOGESCHOOL

De andere kant op

- De lus kan ook “de andere kant op gaan”. We beginnen “hoog” en eindigen “laag”.

```
#include <stdio.h>

int main(void) {
    int i;
    for (i = 9; i >= 0; i = i - 1) {
        printf("hallo %d\n", i);
    }
    getchar();
    return 0;
}
```

DE HAAGSE HOGESCHOOL

{ bla; bla; }

- Compound statement
  - Als een compound statement uit slechts 1 statement bestaat dan word je niet verplicht accolade punctuatoren te gebruiken. Je kunt dan alleen dat ene statement in de for-lus gebruiken.

```
#include <stdio.h>

int main(void) {
    int i;
    for (i = 1; i < 10; i = i + 1)
        printf("hallo %d\n", i);
    getchar();
    return 0;
}
```

Is dit aan te raden?

DE HAAGSE HOGESCHOOL

Inspringen

- Maak je programma leesbaar door netjes in te springen.

```
#include <stdio.h>
int main(void)
{
int i;
for (i = 1; i < 10; i = i + 1)
{
printf("hallo %d\n", i);
}
getchar();
return 0;
}
```

DE HAAGSE HOGESCHOOL

http://en.wikipedia.org/wiki/Indent_style

Inspringen

- Maak je programma leesbaar door netjes in te springen.

```
#include <stdio.h>

int main(void)
{
    int i;
    for (i = 1; i < 10; i = i + 1)
    {
        printf("hallo %d\n", i);
    }
    getchar();
    return 0;
}
```

Er zijn verschillende veel gebruikte manieren. Kies zelf maar blijf wel consequent!

http://en.wikipedia.org/wiki/Indent_style

DE HAAGSE HOGESCHOOL

Voorbeeld: 1+2+3+…+100 =?

```
#include <stdio.h>

int main(void) {
    int i, som = 0;
    for ( ???                    ) {
        som = som + i;
    }
    printf("som = %d\n", som);
    getchar();
    return 0;
}
```

Kan dit slimmer?
http://nl.wikipedia.org/wiki/Somformule_van_Gauss

DE HAAGSE HOGESCHOOL

1+2+3+…+100 =?

- Met behulp van de somformule van Gauss

Code:

```c
#include <stdio.h>

int main(void) {
  int i, som = 0;
  printf("Geef een getal n op waarbij de som moet worden\n");
  printf("bepaald van alle getallen van 1 tot n : ");
  scanf("%d", &som);
  printf("som volgens somformule van Gauss = %d\n", (som*(som+1))/2);
  fflush(stdin);
  getchar();
  return 0;
}
```

Uitvoer:

```
Geef een getal n op waarbij de som moet worden
bepaald van alle getallen van 1 tot n : 111
som volgens somformule van Gauss = 6216
```

do while

- Gebruik een **do while** als het aantal herhalingen bij het programmeren “onbekend” is en ≥1.

```
#include <stdio.h>

int main(void)
    int getal;
    do {
        printf("Geef een positief getal: ");
        scanf("%d", &getal);
    } while (getal <= 0);
    printf("Het ingevoerde getal = %d\n", getal);
    fflush(stdin);
    getchar();
    return 0;
}
```

DE HAAGSE HOGESCHOOL

Voorbeeld: do-while

Code:

```c
#include <stdio.h>

int main(void) {
    int getal;
    do {
        printf("Geef een positief getal: ");
        scanf("%d", &getal);
    } while (getal <= 0);
    printf("Het ingevoerde getal = %d\n", getal);
    fflush(stdin);
    getchar();
    return 0;
}
```

Uitvoer:

```
Geef een positief getal: -100
Geef een positief getal: 0
Geef een positief getal: 0
Geef een positief getal: -1
Geef een positief getal: 1
Het ingevoerde getal = 1
```

while

- Gebruik een **while** als het aantal herhalingen bij het programmeren “onbekend” is en ≥0 is.

```
#include <stdio.h>
int main(void) {
    int getal;
    printf("Geef een positief getal: ");
    scanf("%d", &getal);
    while (getal <= 0) {
        printf(“Helaas! Geef een positief getal: ");
        scanf("%d", &getal);
    }
    printf("Het ingevoerde getal = %d\n", getal);
    fflush(stdin);
    getchar();
    return 0;
}
```

zolang...

DE HAAGSE HOGESCHOOL

Voorbeeld: while

Code:

```c
#include <stdio.h>
int main(void) {
    int getal;
    printf("Geef een positief getal: ");
    scanf("%d", &getal);
    while (getal <= 0) {
        printf("Helaas! Geef een positief getal: ");
        scanf("%d", &getal);
    }
    printf("Het ingevoerde getal = %d\n", getal);
    fflush(stdin);
    getchar();
    return 0;
}
```

Uitvoer:

```
Geef een positief getal: -1
Helaas! Geef een positief getal: 0
Helaas! Geef een positief getal: 1
Het ingevoerde getal = 1
```

For & While equivalentie

```
for (expressie1; expressie2; expressie3) {
      statements
}
```

is equivalent met:

```
expressie1
while (expressie2) {
      statements
      expressie3
}
```

DE HAAGSE HOGESCHOOL

Hoelang is een…

- Hoelang wordt de onderstaande code uitgevoerd?

```
#include <stdio.h>

int main(void) {
  while (1) {
   printf("Hallo");
  }
}
```

- Hoe ziet de for-lus versie van deze code eruit?

DE HAAGSE HOGESCHOOL

Huiswerk

- Schrijf een programma dat de som bepaalt van 5 ingevoerde integers. De uitvoer moet er als onder uitzien. Invoer door de gebruiker is vet gedrukt. Let op het gebruik van de extra lege regels. Maak gebruik van een `for`-lus.

```
Som van vijf getallen

Geef getal 1: 3
Geef getal 2: 5
Geef getal 3: 7
Geef getal 4: 6
Geef getal 5: 9

De som is: 30
```

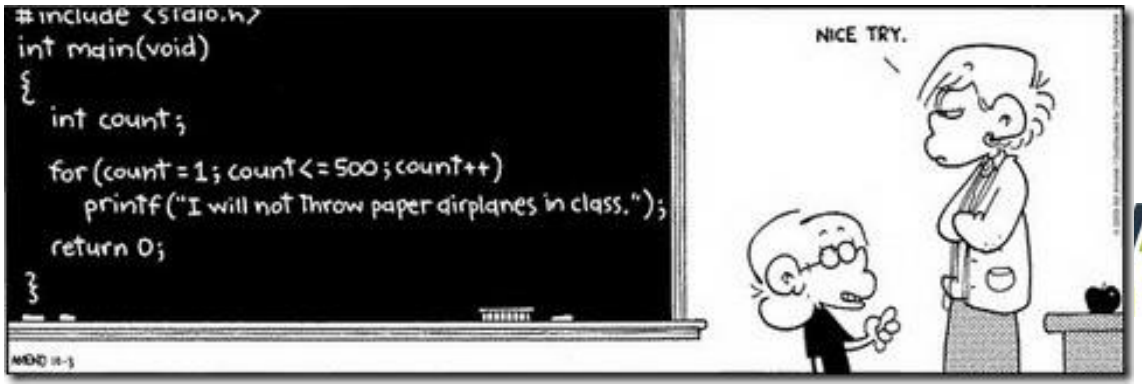
DE HAAGSE HOGESCHOOL

Huiswerk

- Schrijf een programma dat de tafels van 1 t/m 5 netjes naast elkaar afdrukt.

- Bestudeer C boek (Deitel en Deitel. *C How to Program*):
  - paragraaf 2.6
  - paragraaf 3.7
  - paragrafen 4.1 t/m 4.6

- Maak opdrachten:
  - 2 en 4 van https://www.w3resource.com/c-programming-exercises/for-loop/index.php



Academie voor Technology, Innovation & Society Delft
Academie voor ICT & Media

Gestructureerd programmeren in C

GESPRG: Beslissingen

DE HAAGSE HOGESCHOOL

Beslissen

- Er zijn in C 3 beslisopdrachten:
  - `if`
  - `if else`
  - `switch`

DE HAAGSE HOGESCHOOL

if

- Als de expressie WAAR is, wordt *statement1* uitgevoerd.

```
if (expressie) {
    statement1;
}
```

- De haakjes om *expressie* zijn verplicht.

DE HAAGSE HOGESCHOOL

if

- Lees 2 gehele getallen in en druk de grootste af

```
#include <stdio.h>

int main(void) {
    int max, getal;
    printf("Geef een getal: ");
    scanf("%d", &max);
    printf("Geef nog een getal: ");
    scanf("%d", &getal);
    if (getal > max) {
        max = getal;
    }
    printf("Het maximum is: %d\n", max);
    fflush(stdin);
    getchar();
    return 0;
}
```

DE HAAGSE HOGESCHOOL

if else

- Als de expressie WAAR is, wordt *statement1* uitgevoerd.
- Als de expressie ONWAAR is, wordt *statement2* uitgevoerd.

```
if (expressie) {
    statement1;
}
else {
    statement2;
}
```

DE HAAGSE HOGESCHOOL

if else

- Lees 2 gehele getallen in en druk de grootste af

```
#include <stdio.h>

int main(void) {
    int a, b, max;
    printf("Geef een getal: ");
    scanf("%d", &a);
    printf("Geef nog een getal: ");
    scanf("%d", &b);
    if (a > b) {
        max = a;
    }
    else {
        max = b;
    }
    printf("Het maximum is: %d\n", max);
    fflush(stdin); getchar(); return 0;
}
```

Welke van deze twee programma's vind jij **beter**?

DE HAAGSE HOGESCHOOL

Bij welke `if` hoort `else` ?

```
if (a > 3)
    if (a < 2)
        printf("a\n");
else
    printf("b\n");
```

```
if (a > 3)
    if (a < 2)
        printf("a\n");
    else
        printf("b\n");
```

Geven beide programmadelen dezelfde uitvoer als a = 1?

Welke uitvoer?

DE HAAGSE HOGESCHOOL

Bij welke `if` hoort `else` ?

```
if (a > 3) {
    if (a < 2) {
        printf("a\n");
    }
}
else {
    printf("b\n");
}
```

```
if (a > 3) {
    if (a < 2) {
        printf("a\n");
    }
    else {
        printf("b\n");
    }
}
```

Geven beide programmadelen dezelfde uitvoer als a = 1?

Welke uitvoer?

DE HAAGSE HOGESCHOOL

switch

- Zet Nederlands toetscijfer om naar Amerikaans resultaat. Ga uit van:

Nederlands	Amerikaans
8, 9 of 10	A
7	B
6	C
5	D
0, 1, 2, 3 of 4	F

DE HAAGSE HOGESCHOOL

switch

```
switch (cijfer) {
    case 10:
    case 9:
    case 8:
        letter = 'A'; break;
    case 7:
        letter = 'B'; break;
    case 6:
        letter = 'C'; break;
    case 5:
        letter = 'D'; break;
    default:
        letter = 'F'; break;
}
```

DE HAAGSE HOGESCHOOL

if else

```
if (cijfer == 10 || cijfer == 9 || cijfer == 8) {
    letter = 'A';
}
else if (cijfer == 7) {
    letter = 'B';
}
else if (cijfer == 6) {
    letter = 'C';
}
else if (cijfer == 5) {
    letter = 'D';
}
else {
    letter = 'F';
}
```

Is deze else nodig?

if

```
if (cijfer == 10 || cijfer == 9 || cijfer == 8) {
    letter = 'A';
}
if (cijfer == 7) {
    letter = 'B';
}
if (cijfer == 6) {
    letter = 'C';
}
if (cijfer == 5) {
    letter = 'D';
}
if (cijfer < 5 || cijfer > 10) {
    letter = 'F';
}
```

Wat is het verschil met de vorige sheet?

Booleaanse operatoren

- And &&
- Or ||
- Not !

```
do {
    printf("Geef je cijfer: ");
    scanf("%d", &cijfer);
} while (cijfer < 0 || cijfer > 10);
```

Of:

```
} while (!(cijfer >= 0 && cijfer <= 10));
```

Veel gemaakt fout:

```
!(0 <= cijfer <= 10)
```

DE HAAGSE HOGESCHOOL

Short-circuit evaluation

- Bij het uitvoeren van de booleanse operatoren || en && wordt gestopt zodra de uitkomst bekend is.

Als a deelbaar is door b dan …

```
if (a % b == 0) …
```

Gaat fout als b == 0

```
if (b != 0 && a % b == 0) …
```

Gaat goed als b == 0
Dankzij short-circuit evaluation

DE HAAGSE HOGESCHOOL

Conversie van WAAR en ONWAAR

- Een expressie wordt in C als dat nodig is impliciet (automatisch) omgezet naar WAAR of ONWAAR.
  - Een expressie met de waarde 0 wordt ONWAAR.
  - Een expressie met een waarde ongelijk aan 0 wordt WAAR.

Dus:

```
scanf(("%d", &i);
if (i) {
    printf("Hallo");
}
```

Is hetzelfde als:

```
scanf(("%d", &i);
if (i != 0) {
    printf("Hallo");
}
```

Het is beter om expliciet te zeggen wat je bedoelt.

DE HAAGSE HOGESCHOOL

Vergelijken met operatoren

Code :

```c
#include <stdio.h>

int main(void) {
char a = 'a',b = 'b', g = 4;

    printf("BOOLEAANSE EN RELATIONELE OPERATOREN, VERGELIJKEN MAAR! \n");
    printf("5>3 && a!=b    -> %d \n", 5>3 && a!=b);
    printf("5>3 && a==b    -> %d \n", 5>3 && a==b);
    printf("!(5>3 && a!=b) -> %d \n", !(5>3 && a!=b));
    printf("5<3 && a==b    -> %d \n", 5<3 && a==b);

    printf("5>3 || a!=b    -> %d \n", 5>3 || a!=b);
    printf("5>3 || a==b    -> %d \n", 5>3 || a==b);
    printf("!(5>3 || a==b) -> %d \n", !(5>3 || a==b));
    printf("!(5<3) || a==b -> %d \n", !(5<3) || a==b);

    printf("!(!(5<3) || a==b)     -> %d \n", !(!(5<3) || a==b));
    printf("!((5<3 || a==b) && 1) -> %d \n", !((5<3 || a==b) && 1));
    printf("(!(5<g) || a==b)      -> %d \n", (!(5<g) || a==b));

    fflush(stdin);
    getchar();
    return 0;
}
```

Vergelijken met operatoren

Uitvoer:

```
BOOLEAANSE EN RELATIONELE OPERATOREN, VERGELIJKEN MAAR!
5>3 && a!=b    -> 1
5>3 && a==b    -> 0
!(5>3 && a!=b) -> 0
5<3 && a==b    -> 0
5>3 || a!=b    -> 1
5>3 || a==b    -> 1
!(5>3 || a==b) -> 0
!(5<3) || a==b -> 1
!(!(5<3) || a==b)     -> 0
!((5<3 || a==b) && 1) -> 1
(!(5<g) || a==b)      -> 1
```

DE HAAGSE HOGESCHOOL

Programmeren == Moeilijk ?

- Schrijf een programma dat…
  - Hoe bedenk je een programma?
- Stap voor stap…
  - Vastleggen structuur algoritme
  - Stapgewijze verfijning

Vastleggen structuur algoritme: Flow chart

- Een flow chart gebruik je om visueel het algoritme (reeks instructies) vast te leggen.
- Symbolen flow chart:
  - Begin/eind:

  - Actie:

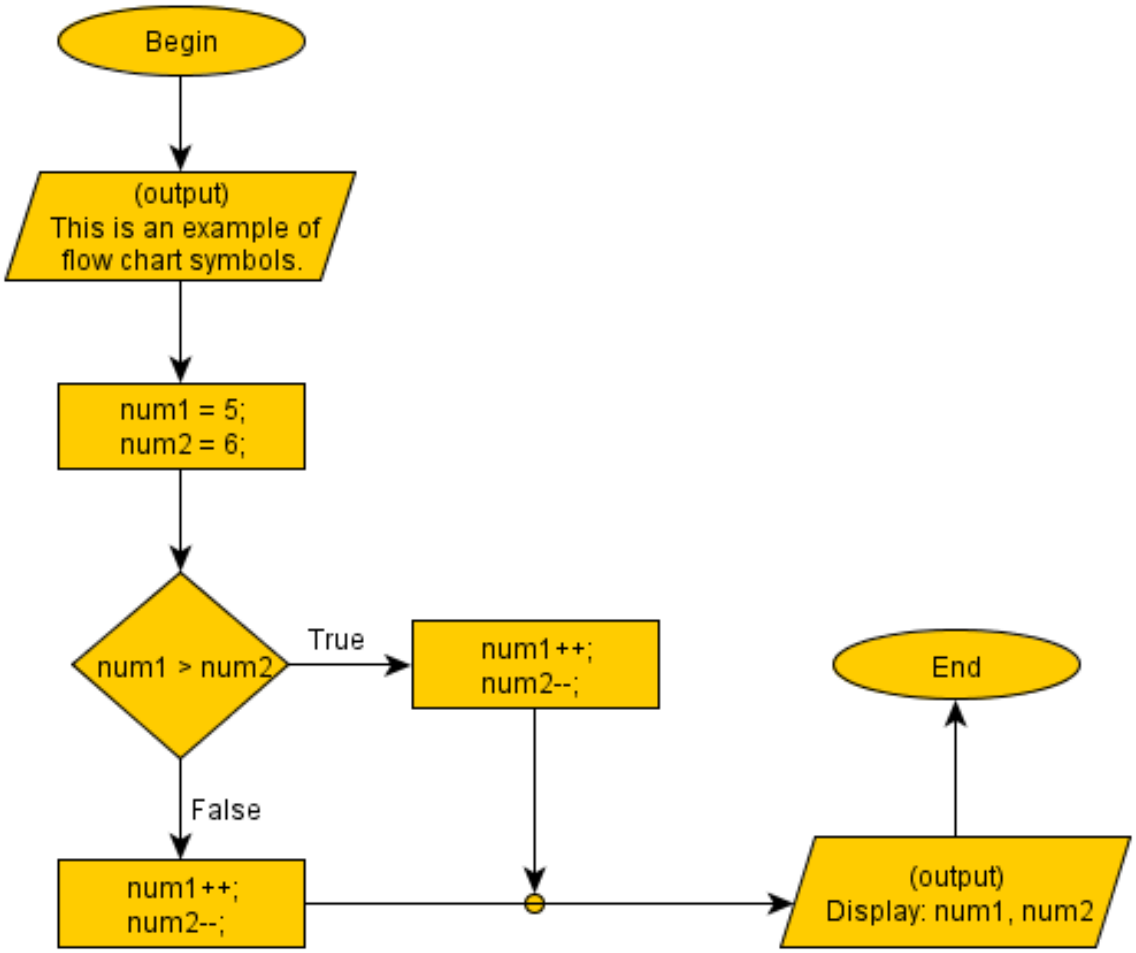
  - Input/output

  - Beslissing

DE HAAGSE HOGESCHOOL

Voorbeeld flow chart

- Voorbeeld flow chart:
- Tekstuele inhoud van alle symbolen is niet aan regels verbonden!
- Hiernaast gebruikte tekst is dus slechts een mogelijkheid!



DE HAAGSE HOGESCHOOL

Tafels stap voor stap

- Schrijf een programma dat een geheel getal $0 < n < 7$ inleest en vervolgens de tafels van 1 t/m n naast elkaar afdrukt.

DE HAAGSE HOGESCHOOL

Stap 0: Bezint eer gij begint

- Snap je de opdracht?
- Bedenk mogelijke testgevallen.
  - Bij de invoer 4 moet het programma de volgende uitvoer produceren:

```
 1 x 1 =  1  1 x 2 =  2  1 x 3 =  3  1 x 4 =  4
 2 x 1 =  2  2 x 2 =  4  2 x 3 =  6  2 x 4 =  8
 3 x 1 =  3  3 x 2 =  6  3 x 3 =  9  3 x 4 = 12
 4 x 1 =  4  4 x 2 =  8  4 x 3 = 12  4 x 4 = 16
 5 x 1 =  5  5 x 2 = 10  5 x 3 = 15  5 x 4 = 20
 6 x 1 =  6  6 x 2 = 12  6 x 3 = 18  6 x 4 = 24
 7 x 1 =  7  7 x 2 = 14  7 x 3 = 21  7 x 4 = 28
 8 x 1 =  8  8 x 2 = 16  8 x 3 = 24  8 x 4 = 32
 9 x 1 =  9  9 x 2 = 18  9 x 3 = 27  9 x 4 = 36
10 x 1 = 10 10 x 2 = 20 10 x 3 = 30 10 x 4 = 40
```

DE HAAGSE HOGESCHOOL

Stap 1: Alle begin is ~~moeilijk~~ makkelijk.

```c
#include <stdio.h>

/* © 2013 Mark Schrauwen */

int main(void) {

    /* Hier komt de code */

    fflush(stdin);
    getchar();
    return 0;
}
```

DE HAAGSE HOGESCHOOL

Stap 2: Invoer.

```c
#include <stdio.h>

/* © 2013 Mark Schrauwen */
/* Dit programma leest een geheel getal 0 < n < 7 en drukt
   vervolgens de tafels van 1 t/m n naast elkaar af */

int main(void) {
    int n;

    printf("Geef de waarde van n (1..6): ");
    scanf("%d", &n);
    printf("Test n = %d", n);

    fflush(stdin);
    getchar();
    return 0;
}
```

DE HAAGSE HOGESCHOOL

Stap 3: Controle op invoer.

```c
#include <stdio.h>

int main(void) {
    int n;

    do {
        printf("Geef de waarde van n (1..6): ");
        scanf("%d", &n);
    } while (n < 1 || n > 6);

    printf("Test n = %d", n);

    fflush(stdin);
    getchar();
    return 0;
}
```

DE HAAGSE HOGESCHOOL

Stap 4. Eerste regel van de tafels afdrukken.

```c
#include <stdio.h>

int main(void) {
    int n, tafel;

    do {
        printf("Geef de waarde van n (1..6): ");
        scanf("%d", &n);
    } while(n < 1 || n > 6);

    for (tafel = 1; tafel < n + 1; tafel = tafel + 1) {
        printf(" 1 x %d = %2d ", tafel, 1 * tafel);
    }
    printf("\n");

    fflush(stdin);
    getchar(); return 0;
}
```

DE HAAGSE HOGESCHOOL

Stap 5: Alle regels afdrukken

```c
#include <stdio.h>

int main(void) {
    int n, tafel, regel;

/*  ...  */

for (regel = 1; regel < 11; regel = regel + 1) {
     for (tafel = 1; tafel < n + 1; tafel = tafel + 1) {
       printf("%2d x %d = %2d ", regel, tafel, regel*tafel);
     }
     printf("\n");
}
/*  ...  */
```

DE HAAGSE HOGESCHOOL

Stap 6: Een laatste verbetering

```c
#include <stdio.h>

int main(void) {
    int factor, tafel, n;

    do {
        printf("Geef de waarde van n (1..6): ");
        fflush(stdin);
    } while (scanf("%d", &n) != 1 || n < 1 || n > 6);

/*  ...  */
```

DE HAAGSE HOGESCHOOL

Rekening houden met crazy users..

Code:

```c
#include <stdio.h>

int main(void) {
  int n, scanUitvoer;
  printf("geef een symbool op (cijfer, letter of anders): ");
  scanUitvoer = scanf("%d", &n);
  printf("scanf() output: %d",scanUitvoer);
  fflush(stdin);
  getchar();
  return 0;
}
```

Uitvoer:

```
geef een symbool op (cijfer, letter of anders): &&&&
Scanf() output: 0

geef een symbool op (cijfer, letter of anders): 123
Scanf() output: 1
```

Huiswerk

- Bestudeer C boek (Deitel en Deitel. *C How to Program*):
  - Paragrafen 3.1 t/m 3.6 (behalve tekst op blz. 107 onder figuur 3.3)
  - Paragrafen 4.7 en 4.8, 4.10 t/m 4.12

- Maak opdrachten:
  - 2 en 13 van https://www.w3resource.com/c-programming-exercises/conditional-statement/index.php

DE HAAGSE HOGESCHOOL