VANDAAG

PROGRAMMASTRUCTUUR

DECLARATIES

INSTRUCTIES

SAMENGESTELDE INSTRUCTIES (COMPOUND STATEMENTS)

OPERATOREN

KEUZE-INSTRUCTIES (IF, SWITCH)

HERHALINGSINSTRUCTIES (WHILE, DO WHILE, FOR)

RIJEN en ARRAYS

PROGRAMMASTRUCTUUR

```
COMMENTAAR                  /* ... */
                            // ...

GLOBALE DECLARATIES
    INCLUDE FILES           #include
    CONSTANTEN              #define
    VARIABELEN              int, float
    FUNCTIES

HOOFDFUNCTIE
  KOP
    TYPE
    NAAM                    int main (...)
    PARAMETERLIJST
  COMPOUND STATEMENT
    LOKALE DECLARATIES
      CONSTANTEN            #define
      VARIABELEN            int, float
    INSTRUCTIES
      INVOER                scanf ("...", );
      BEREKENINGEN          var1 = expr;
      UITVOER               printf ("...", );
      AFSLUITING            return ( );
```

COMMENTAAR

IS HEEL BELANGRIJK !!!

VOORBEELD

```
/* Programma om in dialoog met de
 * gebruiker twee maal de som van
 * twee getallen te berekenen.
 */

#include <stdio.h>

int main ()
{
  int getal1, getal2, som;

  printf ("Voer twee getallen in: ");
  scanf ("%d %d", &getal1, &getal2);
  som = getal1 + getal2;
  printf ("De som is: %d\n", som);

  printf ("Voer nogmaals twee ");
  printf ("getallen in: ");
  scanf ("%d %d", &getal1, &getal2);
  som = getal1 + getal2;
  printf ("De som is: %d\n", som);

  return (0);
}
```

```
Voer twee getallen in: 2 3
De som is: 5
Voer nogmaals twee getallen in: 2000 1
De som is: 2001
```

LIBRARIES

BIBLIOTHEKEN ("LIBRARIES") BEVATTEN HANDIGE, VOORGEDEFINIEERDE FUNCTIES DIE IN EEN PROGRAMMA GEBRUIKT KUNNEN WORDEN

EEN ZOGENAAMDE "HEADER" FILE ZORGT VOOR DE VERBINDING TUSSEN PROGRAMMA EN BIBLIOTHEEK

```
#include <stdio.h>
#include <math.h>
```

EEN AANTAL BIBLIOTHEKEN WORDEN "STANDAARD" BIJGEVOEGD ANDERE MOETEN IN DE MAKEFILE OPGEGEVEN WORDEN

DECLARATIES

ALLE VARIABELEN IN EEN PROGRAMMA MOETEN GEDECLAREERD WORDEN

FORMAAT VAN DECLARATIE

```
type naam;
type naam1,
     naam2;
type naam1, ..., naamN;
type naam = waarde;
```

DATA TYPEN

```
char
int
float
double
```

VOORBEELDEN

```
int    lengte, breedte;
char   volgletter;
float  kracht;
double kracht = 10.2;
```

IDENTIFIERS

NAMEN ("IDENTIFIERS") BESTAAN UIT COMBINATIES VAN LETTERS, CIJFERS EN HET TEKEN UNDERSCORE

NAMEN MOETEN MET EEN LETTER OF EEN UNDERSCORE BEGINNEN

ER IS ONDERSCHEID TUSSEN HOOFD- EN KLEINE LETTERS

EEN AANTAL NAMEN (DE "KEYWORDS") ZIJN GERESERVEERD VOOR SPECIALE DOELEN, BIJV. DELEN VAN INSTRUCTIES

VOORBEELDEN

```
float   B4U, H2O, F77;

int     aantal_metingen,
        aantal_Metingen,
        aantalMetingen;
```

DECLARATIES EN INSTRUCTIES

DECLARATIES EN INSTRUCTIES WORDEN AFGESLOTEN DOOR EEN PUNT-KOMMA

DECLARATIES EN INSTRUCTIES KUNNEN DAARDOOR OVER MEERDERE REGELS UITGEBREID WORDEN

```
x1 = (-b + sqrt (b*b - 4*a*c))
                  /
     (          2 * a          );
```

TOEKENNING / ASSIGNMENT (1)

EEN TOEKENNING HEEFT LINKS EEN VARIABELE EN RECHTS EEN AL DAN NIET INGEWIKKELDE UITDRUKKING

```
variabele = uitdrukking;
```

EEN UITDRUKKING BESTAAT UIT CONSTANTEN, VARIABELEN, FUNCTIES EN OPERATOREN

VOORBEELDEN

```
opp  = lengte * breedte;

pi   = 3.14;
opp  = pi * straal*straal;
omtr = 2.0 * pi * straal;

wortel = sqrt (b*b - 4*a*c);
x1 = (-b + wortel) / (2.0 * a);
x2 = (-b - wortel) / (2.0 * a);
```

TOEKENNING / ASSIGNMENT (2)

LET ALTIJD OP GEHELE GETALLEN IN UITDRUKKINGEN DIE EEN NIET-GEHELE (FLOAT) WAARDE OPLEVEREN

BIJVOORBEELD

```
int   basis = 3, hoogte = 3;
float opp;

opp  = (basis * hoogte) / 2;
printf ("Oppervlak = %f\n", opp);

opp  = (basis * hoogte) / 2.0;
printf ("Oppervlak = %f\n", opp);
```

GEEFT ALS UITVOER

```
Oppervlak = 4.000000

Oppervlak = 4.500000
```

COMPOUND STATEMENT

EEN AANTAL INSTRUCTIES TUSSEN ACCOLADES VORMEN EEN EENHEID, DIE ALS EEN ENKELE OPDRACHT WORDT BESCHOUWD

BINNEN DEZE EENHEID KUNNEN LOKALE DECLARATIES GEDAAN WORDEN

BIJVOORBEELD

```
float basis = 3.0, hoogte = 3.0;
float opp;
{
  float basis = 5.0;
  opp   = (basis * hoogte) / 2.0;
  printf ("Oppervlak = %f\n", opp);
}
opp   = (basis * hoogte) / 2.0;
printf ("Oppervlak = %f\n", opp);
```

GEEFT ALS UITVOER

```
Oppervlak = 7.500000
Oppervlak = 4.500000
```

OPERATORS / OPERATOREN

REKENKUNDIGE OPERATOREN

+ - * / % ++ --

(GEEN MACHTSVERHEFFEN)

LOGISCHE OPERATOREN

== != > < >= <=

! && ||

(FALSE HEEFT WAARDE 0)

CONDITIONS / VOORWAARDEN

VOORWAARDEN ZIJN UITDRUKKINGEN DIE NA EVALUATIE "TRUE" OF "FALSE" GEVEN

```
aantalMetingen > 100
(X < 0.0) || (X > 1.0)
(X < 0.0) && (Y > 3.3)
a == 3.4
```

LET OP, DE VOLGENDE TOEKENNING HEEFT OOK EEN LOGISCHE WAARDE

```
a = 3.4
```

EN DIE HOEFT DUS NIET TE KLOPPEN MET DE WAARDE DIE JE VERWACHT

DE LOGISCHE WAARDE "FALSE" IS 0

Input Output Standard error

```
int main ()
{
  /* declarations */
  float floatOne, floatTwo, floatThree;

  /* message for user */
  printf ("Geef drie floating point getallen, gescheiden door een spatie:\n");

  /* read three floating point values from standard input */
  scanf ("%f %f %f", &floatOne, &floatTwo, &floatThree);

/* print the three values naar standard ERROR output */
  fprintf (stderr,"\nIngevoerde waarden geprint naar stderr: %f, %f, %f\n\n",
     floatOne, floatTwo, floatThree);

  /* print the three values on standard output */
  printf ("Ingevoerde waarden: %f, %f, %f\n", floatOne, floatTwo, floatThree);

  return (0);
}
```

fprintf ()

```
fprintf (stderr,
        "\nIngevoerde waarden geprint naar stderr: %f, %f, %f\n\n",
        floatOne, floatTwo, floatThree);
```

Nauwkeurigheid

Voorbeeld van (gebrek aan) nauwkeurigheid:
Dezelfde berekeningen worden uitgevoerd in
floating point en in double precision,
uitgaande van X = 0.000000000001 = 1.000000e-012.
Resultaten worden exponentieel (resp. float) weergegeven.
De exacte uitkomst is 1.0 (een)!

Berekende waarde	Resultaat in floating point	Resultaat in double precision
X^1	1.000000e-012	1.000000e-012
X^2	1.000000e-024	1.000000e-024
X^3	1.000000e-036	1.000000e-036
X^4	0.000000e+000	1.000000e-048
X^4 / X^1	0.000000e+000 (0.0)	1.000000e-036 (0.0)
X^4 / X^2	0.000000e+000 (0.0)	1.000000e-024 (0.0)
X^4 / X^3	0.000000e+000 (0.0)	1.000000e-012 (0.0)
X^4 / X^4	0.000000e+000 (0.0)	1.000000e+000 (1.0)

Voorbeelden

1. Simple math
2. Logische toekenning

KEUZE: IF

IF

```
if (voorwaarde) instructie;
```

```
if (voorwaarde)
  ene_instructie
else
  andere_instructie;
```

```
if (voorwaarde)
{
  een_of_meer_instructies;
}
else
{
  een_of_meer_andere_instructies;
}
```

VOORBEELDEN

```
if ( (i<0) || (i>10) )
{
  printf ("waarde i = %d ", i);
  printf ("buiten interval");
}
```

```
if ( (i>=0) && (i<=10) ) ...
```

KEUZE: NESTED IF

NESTED IF'S

```
if (voorwaarde)
if (andere_voorwaarde)
ene_instructie
else
andere_instructie;
```

WAT WORDT BEDOELD?

```
if (voorwaarde)
{
  if (andere_voorwaarde)
  {
    ene_instructie;
  }
  else
  {
    andere_instructie;
  }
}
```

OF

```
if (voorwaarde)
{
  if (andere_voorwaarde)
    ene_instructie;
}
else
{
  andere_instructie;
}
```

KEUZE: SWITCH

SWITCH

```
char teken;

scanf ("%c", &teken);

switch (teken)
{
  case '0':    case '1':
  case '2':    case '3':
  case '4':    case '4':
  case '6':    case '5':
  case '8':    case '9':
    printf ("Teken is cijfer\n");
    break;

  case ' ':
    printf ("Teken is spatie\n");
    break;

  default:
    printf
     ("Geen cijfer of spatie\n");
    break;
}
```

VOOR KEUZE UIT MEERDERE OPTIES

HERHALING: WHILE

WHILE

```
while (voorwaarde) instructie;
```

VOORBEELD

```
int som = 0, teller = 0;
while (teller <= 10)
{
  teller = teller + 1;
  som    = som + teller;
}
printf ("Totale som = %d\n", som);
```

ALS DE VOORWAARDE DE EERSTE KEER NIET WAAR IS, WORDT ER NIETS GEDAAN

HERHALING: DO WHILE

DO WHILE

```
do
  instructie1; ...;
  instructieN;
while (voorwaarde);
```

VOORBEELD

```
int som = 0, teller = 0;
do
  teller = teller + 1;
  som    = som + teller;
while (teller < 10)
```

ALS DE VOORWAARDE DE EERSTE KEER NIET WAAR IS, WORDT ER TOCH EEN KEER IETS GEDAAN

HERHALING: FOR

FOR

```
for (initialisatie;
     voorwaarde;
     instructie_na_herhaling)
       instructie_van_herhaling;
```

IS HETZELFDE ALS

```
initialisatie;
while (voorwaarde)
{
  instructie_van_herhaling;
  instructie_na_herhaling;
}
```

VOORBEELD

```
int som = 0;
for (teller=1; teller<=10; teller++)
  som += teller:
```

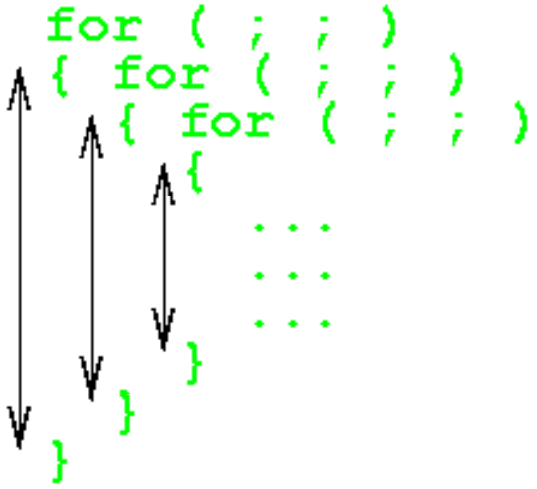
HERHALING: NESTED FOR (1)

NESTED FOR LOOPS

```
for (in_1; vw_1; na_in_1)
{
  for (in_2; vw_2; na_in_2)
  {
    instructie;
  }
}
```

DE BINNENSTE LOOP LOOPT HET SNELST

HERHALING: NESTED FOR (2)



DUIDELIJK AANGEVEN VOORKOMT FOUTEN

VEEL VOORKOMENDE FOUTEN IN LOOPS

- BEREIK KLOPT NIET
- VOORWAARDE KLOPT NIET
- GEDRAG BIJ GRENSWAARDEN

ONEINDIGE LOOPS

HERHALING ONDERBREKEN

break

"break" VERLAAT DE HERHALING TOTAAL

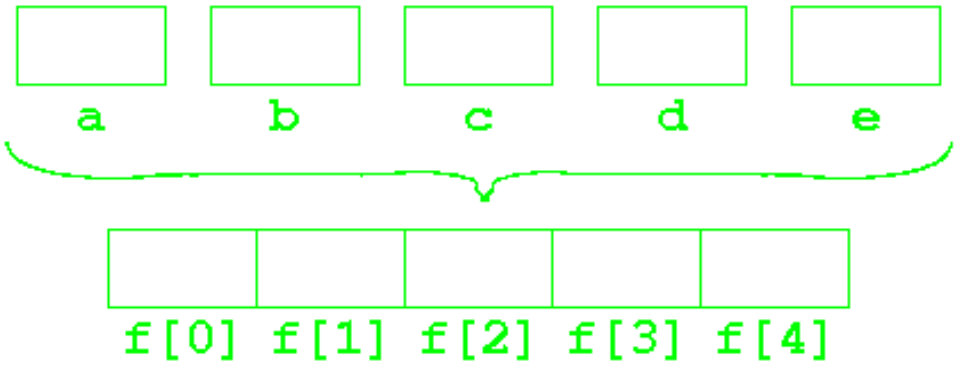
continue

"continue" BREEKT DE HUIDIGE RONDE VAN DE HERHALING AF EN GAAT VERDER MET DE VOLGENDE

```
for (in_1; vw_1; na_in_1)
{
  for (in_2; vw_2; na_in_2)
  {
    ...
    if (...) break;
    ...
  }
}
```

RIJEN / ARRAYS (1)

SAMENGESTELDE OBJECTEN



DECLARATIE

```
type naam [aantal];
float f[5];
```

BELANGRIJKSTE TOEPASSING

IN HERHALINGEN

VOORBEELD VOLGT

RIJEN / ARRAYS (2)

DECLARATIE VAN ARRAY

```
float mol[5];
```

GEEFT VARIABELEN

```
mol[0] mol[1] mol[2] mol[3] mol[4]
```

GEBRUIK VAN VARIABELEN

```
mol[2] = 7.0;

printf ("Waarde derde element %f",
          mol[2];

som_even = mol[1] + mol[3];

index = 3;  mol[index] = 9.9;
```

RIJEN / ARRAYS (3)

IN PLAATS VAN

```
float som, mol[5];

mol[0] = 3.3;   ...
som = mol[0] + mol[1] + mol[2] +
      mol[3] + mol[4];
```

HET VOLGENDE

```
float som, mol[5];
int   i;

som = 0.0;
for (i=0; i<5; i++) som += mol[i];
```

OF, FLEXIBELER

```
#define max_el   10;

float som, mol[max_el];
int   i;

som = 0.0;

for (i=0; i<max_el; i++)
  som += mol[i];
```

MEERDIMENSIONALE ARRAYS

```
/* Initialiseer twee matrices. */

#define max_dim 10

int main ()
{

  double mat_a[max_dim][max_dim],
         mat_b[max_dim][max_dim];

  int    i,
         j;

  for (i=0; i<max_dim; i++)
  {
    for (j=0; j<max_dim; j++)
    {
      mat_a[i][j] = 0.0;
      mat_b[i][j] = 0.0;
    }
  }

}
```

Voorbeelden

1. Arrays
2. Crashing programma met stderr