

Inhoud

1. Basisstructuur
2. Inleiding tot C
3. Modelcomputer DRAMA
4. Programma's voor DRAMA

13

1.2 Inleiding tot C

- Kenmerken van C
- Doelstelling
- Overzicht
- Eenvoudige programma's

C
Programmeertaal C

14

C Kenmerken van C

- Hogere programmeertaal
 - ♦ Grote verzameling types, (strenge) type controle
 - ♦ Zelf nieuwe types definiëren
 - ♦ Klassieke controle-structuren
 - ♦ Functies
- Assembleertaal
 - ♦ Operatoren met equivalent op machine-niveau
 - ♦ Bewerkingen op adressen mogelijk
- Basis voor ...
 - ♦ C++, Java, C#, ...

15

C Doelstelling

- Passieve kennis van C
 - ♦ Begrijpen maar niet zelf kunnen schrijven
- Voorbeelden:
 - ♦ Stijgende complexiteit

16

C Overzicht

- Eenvoudig C
- Arrays
- Functies
- Records
- Dynamische gegevenstructuren

17

C Overzicht

- **Eenvoudig C**
 - ♦ Declaraties, main, operatoren, opdrachten, in- en uitvoer
 - Voorbeeld 1-1
 - ♦ if, while, samengestelde opdracht
 - Voorbeeld 1-5
 - ♦ Switch opdracht
- Arrays
- Functies
- Records
- Dynamische gegevenstructuren

18

C Main – Globale variabelen

Voorbeeld 1-1

```
int a, b, somk;

main() {
    a = getint();
    b = getint();
    somk = a * a + b * b;
    printint (somk);
}
```

19

C Main – Globale variabelen

Voorbeeld 1-1

```
int a, b, somk;

main() {
    a = getint();
    b = getint();
    somk = a * a + b * b;
    printint (somk);
}
```

- Globale variabelen
- Type int = integer
- Declaratie vereist

20

C Types

- **Standaard types**
 - int : voor **gehele waarde**
 - C voorziet verschillende groottes
 - short, int, long
 - Niet noodzakelijk verschillend
 - **Geen** type voor **logische waarde**
logische waarde verwacht:
 - 0 ➔ false
 - ≠ 0 ➔ true
 - **String** (rij van symbolen)
 - Constante: "som = "

21

C Main – Globale variabelen

Voorbeeld 1-1

```
int a, b, somk;

main() {
    a = getint();
    b = getint();
    somk = a * a + b * b;
    printint (somk);
}
```

- Hoofdprogramma **main**
- Tussen () : Parameters
- Tussen { } : Lichaam
 - Opdrachten
- In voorbeelden: (voorlopig)
 - geen parameters
 - geen lokale variabelen

22

C Operatoren

- **Uitdrukkingen**
 - Rekenkundige operatoren (zoals Java)
+ - * / %
 - Relationale operatoren (zoals Java)
> >= < <= == !=
 - Logische operatoren
 - && tweede operand niet berekend indien eerste == 0 (false)
 - || tweede operand niet berekend indien eerste != 0 (true)

23

C Operatoren

- **Uitdrukkingen**
 - Toekenningsoperator: **i = 5** ← **UITDRUKKING!**
 - Met waarde: 5
 - Met neveneffect: waarde i is 5
 - Klassieke fout
 - i = 5 ← altijd waar!
 - i == 5
 - j = i = k = 5 j = (i = (k = 5))
 - Samengestelde toekenningsoperatoren
 - i * = j - 3 ➔ i = i * (j - 3)
 - += -= *= /= % =

24

C Opdrachten

- Opdracht
 - Uitdrukking gevolgd door ;
- Invoer en uitvoer
 - C definieert geen opdrachten
 - In C worden functies uit bibliotheken gebruikt
 - Wij gebruiken
 - getint() → leest geheel getal in en geeft dit als resultaat terug
 - printint (<uitdrukking>) → drukt geheel getal af
 - printstr ("string") → drukt string af
- If, while, switch, ...

25

C Voorbeeld 1-1

```
int a, b, somk;

main() {
    a = getint();
    b = getint();
    somk = a * a + b * b;
    printstr ("somkwad = ");
    printint (somk);
}
```

26

C If while Voorbeeld 1-5

```
int a, abs, i;

main() {
    i = 1;
    while (i <= 10) {
        a = getint ();
        if ( a > 0 ) abs = a;
        else abs = -a;
        printint (a, abs);
        i = i + 1;
    }
}
```

27

C If while Voorbeeld 1-5

```
int a, abs, i;

main() {
    i = 1;
    while (i <= 10) {
        a = getint ();
        if ( a > 0 ) abs = a;
        else abs = -a;
        printint (a, abs);
        i = i + 1;
    }
}
```

28

C If while Voorbeeld 1-5

- Opdracht
 - Uitdrukking gevolgd door ;
 - Samengestelde opdracht (zonder afsluitende ;)


```
{
            opdracht
            ...
          }
```
 - If, while, ...

29

C If while Voorbeeld 1-5

```
int a, abs, i;

main() {
    i = 1;
    while (i <= 10) {
        a = getint ();
        if ( a > 0 ) abs = a;
        else abs = -a;
        printint (a, abs);
        i = i + 1;
    }
}
```

30

C Switch-opdracht

```

switch (i - 7) {
  case 0:
    a = 0;
    break;
  case 1:
    a = 1;
    break;
  default:
    a = 2;
    break;
}

```

Diagram illustrating the switch statement structure with a callout box showing a generic template:

```

switch ( uitdrukking )
{
  case waarde1 :
    opdrachten1
  case waarde2 :
    opdrachten2
  ...
  default:
    opdrachten
}

```

31

C Switch-opdracht

```

switch (i - 7) {
  case 0:
    a = 0;
    break;
  case 1:
    a = 1;
    break;
  default:
    a = 2;
    break;
}

```

Execution flow diagram for $i = 9$ and $i = 5$:

- $i = 9$: $a = 1$
- $i = 5$: $a = 2$

32

C Switch-opdracht

```

switch (i - 7) {
  case 0:
    a = 0;
    break;
  case 1:
    a = 1;
    break;
  default:
    a = 2;
    break;
}

```

Execution flow diagram for $i = 8$ and $i = 9$:

- $i = 8$: $a = 0$
- $i = 9$: $a = 1$ (then $a = 2$ due to missing break)

33

Inhoud

1. Basisstructuur
2. Inleiding tot C
3. Modelcomputer DRAMA
4. Programma's voor DRAMA

34

1.3 Modelcomputer

- Verantwoording:
 - Basiskennis (– details)
 - binding commercieel beschikbare processor
- DRAMA
 - Decimale RekenAutomaat met Meerdere Accumulatoren
 - Decimaal (↔) echte computer: binair
 - Compacter, Leesbaarder voor menselijke gebruiker

Diagram comparing DRAMA (Model Computer) and Reële Computer (Real Computer).

35

DRAMA-computer

Processor components:

- Bevelenteller (PC): 1234
- Accumulatoren (Registers): R0 (1234567890), R1 (8234872348), ..., R9 (0000001235)
- Conditiecode (CC): 1

Werkgeheugen (Memory):

- 0000: 1234567890
- 0001: 7362516278
- ...
- 9999: 8271683928

Nooit leeg!

36

DRAMA: Getalvoorstelling

- Gehele getallen
 - 10-complement (negatief: $10^{10} - |x|$)
 - Positief: $0 \rightarrow 4.999.999.999$
 $0000000000 \rightarrow 4999999999$
 - Negatief: $-5.000.000.000 \rightarrow -1$
 $5000000000 \rightarrow 9999999999$

	Inwendig:	In voorbeelden:
♦ +1234 ?	0000001234	1234
♦ -1234 ?	9999998766	-1234

DRAMA: Bevelenset

- Bevel = (meestal) 3 delen

FUNCODE	ACC, OPERAND
---------	--------------
- ♦ Functiecode (3 letters): vb. HIA
 - Eventueel interpretatieveld: vb. HIA.w
- ♦ Naam van Accumulator: vb. R0
- ♦ Operand:
 - Geheugenadres: vb. 1209
 - Geheel getal: vb. 15
 - Naam van Accumulator: vb. R7

DRAMA: Bevelenset

- Transportbevelen
 - geheugenregisters \leftrightarrow accumulatoren
- ♦ HIA = Haal In Accumulator
 - HIA R7,123 | $R7 \leftarrow \text{Geheugen}[123]$
 - HIA.w R6,10 | $R6 \leftarrow 10$
 - HIA R8,R7 | $R8 \leftarrow R7$
- ♦ BIG = Berg In Geheugen
 - BIG R7,123 | $\text{Geheugen}[123] \leftarrow R7$

DRAMA: Bevelenset

- Rekenkundige bevelen
 - Basisbewerkingen: +, -, ×, /, %
- ♦ OPT, AFT, VER, DEL, MOD
- ♦ Zelfde varianten als HIA
 - OPT R7,123 | $R7 \leftarrow R7 + \text{Geheugen}[123]$
 - AFT.w R6,10 | $R6 \leftarrow R6 - 10$
 - VER R8,R7 | $R8 \leftarrow R8 \times R7$

DRAMA: Bevelenset

- Invoer / Uitvoer
 - Vereenvoudigd: gehele getallen lezen/schrijven
- ♦ LEZ = Lezen
 - LEZ | $R0 \leftarrow \text{InvoerOrgaan}$
- ♦ DRU = Drukken
 - DRU | $\text{UitvoerOrgaan} \leftarrow R0$
- ♦ NWL = Nieuwe Lijn
 - NWL | $\text{UitvoerOrgaan} \leftarrow \downarrow$
- ♦ DRS = Druk String
 - DRS | $\text{UitvoerOrgaan} \leftarrow R0 (\text{adres})$


Voorbeeld

```

int a, b, somk;
main() {
    a = getint();
    b = getint();
    somk = a * a + b * b;
    printint(somk);
}
    
```

geheugenreservaties	
a	$\rightarrow 1000$
b	$\rightarrow 1001$
somk	$\rightarrow 1002$

LEZ	BIG R0,1000	a \leftarrow getint()
LEZ	BIG R0,1001	b \leftarrow getint()
HIA	R1,1000	R1 \leftarrow a*a
VER	R2,1001	R2 \leftarrow b*b
HIA	R2,1001	R1 \leftarrow a*a+b*b
OPT	R1,R2	somk \leftarrow a*a+b*b
BIG	R1,1002	somk \leftarrow a*a+b*b
HIA	R0,1002	printint(somk)
DRU		




Voorbeeld

- 12 DRAMA-bevelen
- Kan efficiënter!
- Maar:
 - Correctheid is belangrijker
 - Optimalisatie kan later
 - Aanpasbaarheid?

	geheugenreservaties
	a → 1000
	b → 1001
	somk → 1002
LEZ	
BIG R0,1000	
LEZ	
BIG R0,1001	
HIA R1,1000	
VER R1,1000	
HIA R2,1001	
VER R2,R2	
OPT R1,R2	
BIG R1,1002	
HIA R0,1002	
DRU	

43




Voorbeeld (bis)

```
int a, b, somk;
main() {
    a = getint();
    b = getint();
    somk = a*a + b*b;
    printint(somk);
}
```

	geheugenreservaties
	a → R1
	b → R2
	somk → R0
LEZ	
HIA R1,R0 a ← getint()	
LEZ	
HIA R2,R0 b ← getint()	
HIA R3,R1	
VER R3,R1 R3 ← a*a	
VER R0,R0 somk ← b*b	
OPT R0,R3 somk ← a*a+b*b	
DRU	printint(somk)

44



Voorbeeld (bis)

```
int a, b, somk, c;
main() {
    a = getint();
    b = getint();
    c = getint();
    somk = a*a + b*b;
    printint(somk);
}
```

	geheugenreservaties
	a → R1
	b → R2
	somk → R0
	c → R4
LEZ	
HIA R1,R0 a ← getint()	
LEZ	
HIA R2,R0 b ← getint()	
← LEZ	
HIA R4,R0 c ← getint()	
HIA R3,R1	
VER R3,R1 R3 ← a*a	
VER R0,R0 somk ← b*b	
OPT R0,R3 somk ← a*a+b*b	
DRU	printint(somk)

45

Samenvatting

- Computer → Logische bouwstenen
 - CPU, RAM, in-/uitvoerorganen, hulpgeheugens
 - Eenvoudige instructies
 - Complexere programma's
- Taak:
 - Gegeven: R0 bevat getal x
 - Gevraagd: Schrijf programma dat alleen bestaat uit **reg-reg instructies** en ervoor zorgt dat R0 ← x bevat

~~VER.w R0, -1~~

46