CPROUC/2021-2022

Jesse op den Brouw

CPROUC

Variabelen, constanten en expressies

DE HAAGSE HOGESCHOOL

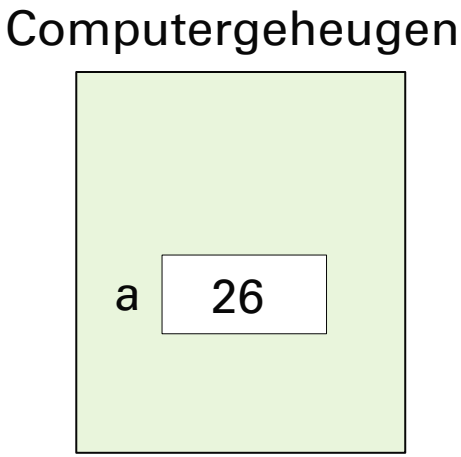
Variabelen

- Een variabele is een klein stukje opslagruimte in het werkgeheugen van de computer. Voorbeeld:

```
int a;
```

- In het geheugen is er nu een stukje ruimte vrijgemaakt en deze heeft de naam a.
- Met behulp van een toekenning kun je waardes stoppen in deze variabele:

```
a = 26;
```



DE HAAGSE HOGESCHOOL

Variabelen

- Integers:
  - signed char 8 bits
  - signed int 32 bits (Arduino: 16 bits)
  - signed short int 16 bits
  - signed long int 32 bits
  - signed long long int 64 bits

- Signed, short en long worden *qualifiers* genoemd.
- Bereik is $-2^{\#bits-1}$ t/m $2^{\#bits-1} -1$

DE HAAGSE HOGESCHOOL

Variabelen

- Integers:
  - unsigned char 8 bits
  - unsigned int 32 bits (Arduino: 16 bits)
  - unsigned short int 16 bits
  - unsigned long int 32 bits
  - unsigned long long int 64 bits

- Unsigned, short en long worden *qualifiers* genoemd.
- Bereik is 0 t/m $2^{\#bits}$ -1

DE HAAGSE HOGESCHOOL

Overflow

- Overflow is een conditie waarbij een berekend resultaat niet “past” in de variabele.
- Overflow wordt gedetecteerd door de processor.
- C “doet” daar niets mee, er wordt geen extra code gegenereerd.
- Voorbeeld met 8 bits:

```
   11111111
   00000001
1) 00000000
```

- Het netto resultaat is 0.

DE HAAGSE HOGESCHOOL

Variabelen

- Floating point:
  - float 32 bits
  - double 64 bits
  - long double 80/128 bits

- long wordt een *qualifier* genoemd.
- Naast gewone getallen kennen floating point getallen ook:
  - +INF plus oneindig
  - -INF min oneindig
  - NaN not a number

DE HAAGSE HOGESCHOOL

Constante

- Een constante wordt eenmalig aangemaakt en kan daarna niet meer veranderd worden.

```
const int aantal = 10;
const char a = 'a';
```

- `const` is een qualifier, `aantal` is een variabele en kan gebruikt worden in expressies.

DE HAAGSE HOGESCHOOL

Karakters

- Een karakter is een integer die een bepaald teken uit de karakterset van de computer symboliseert.

```
char kar = 'a';
```

- Een aantal niet-afdrukbare karakters kan worden opgegeven met een *escape sequence*: `'\n'`, `'\r'`, `'\t'`, `'\a'`, `'\0'`.

```
char newline = '\n';
```

- We gebruiken voor de opdrachten de ASCII-code.

DE HAAGSE HOGESCHOOL

ASCII-code

Dec	Hex		Dec	Hex		Dec	Hex		Dec	Hex		Dec	Hex		Dec	Hex		Dec	Hex	
0	00	NUL	16	10	DLE	32	20		48	30	0	64	40	@	80	50	P	96	60	`	112	70	p
1	01	SOH	17	11	DC1	33	21	!	49	31	1	65	41	A	81	51	Q	97	61	a	113	71	q
2	02	STX	18	12	DC2	34	22	"	50	32	2	66	42	B	82	52	R	98	62	b	114	72	r
3	03	ETX	19	13	DC3	35	23	#	51	33	3	67	43	C	83	53	S	99	63	c	115	73	s
4	04	EOT	20	14	DC4	36	24	$	52	34	4	68	44	D	84	54	T	100	64	d	116	74	t
5	05	ENQ	21	15	NAK	37	25	%	53	35	5	69	45	E	85	55	U	101	65	e	117	75	u
6	06	ACK	22	16	SYN	38	26	&	54	36	6	70	46	F	86	56	V	102	66	f	118	76	v
7	07	BEL	23	17	ETB	39	27	'	55	37	7	71	47	G	87	57	W	103	67	g	119	77	w
8	08	BS	24	18	CAN	40	28	(	56	38	8	72	48	H	88	58	X	104	68	h	120	78	x
9	09	HT	25	19	EM	41	29	)	57	39	9	73	49	I	89	59	Y	105	69	i	121	79	y
10	0A	LF	26	1A	SUB	42	2A	*	58	3A	:	74	4A	J	90	5A	Z	106	6A	j	122	7A	z
11	0B	VT	27	1B	ESC	43	2B	+	59	3B	;	75	4B	K	91	5B	[	107	6B	k	123	7B	{
12	0C	FF	28	1C	FS	44	2C	,	60	3C	<	76	4C	L	92	5C	\	108	6C	l	124	7C	|
13	0D	CR	29	1D	GS	45	2D	-	61	3D	=	77	4D	M	93	5D	]	109	6D	m	125	7D	}
14	0E	SO	30	1E	RS	46	2E	.	62	3E	>	78	4E	N	94	5E	^	110	6E	n	126	7E	~
15	0F	SI	31	1F	US	47	2F	/	63	3F	?	79	4F	O	95	5F	_	111	6F	o	127	7F	DEL

DE HAAGSE HOGESCHOOL

Karakters

- Een karakter is een integer die een bepaald teken uit de karakterset van de computer symboliseert en kan gebruikt worden in berekeningen:

```
int cijfer = 5;
char kar = cijfer + '0';
printf("%c", kar);
```

- Zo kan je gemakkelijk een getal afdrukken op het scherm.

Expressies

- Een expressie is een stukje C-programma waar een resultaat (integer, floating point, karakter) uit voortkomt.
- We kunnen dit resultaat in een variabele stoppen.
- Voorbeeld: `a = 3 * b;`
- `3 * b` is een expressie.
- Ook `a = …` is een expressie.
- Volgorde van bewerken: * / % gaat voor op + -
- Haakjes kunnen de volgorde veranderen
- Er is geen operator voor machtsverheffen

Voorwaarden

- Een voorwaarde is een expressie die 0 of 1 oplevert.
- We kunnen dit resultaat gebruiken bij testen.
- Er zijn zes *relationele operatoren*:
  - == gelijk aan
  - != ongelijk aan
  - > groter dan
  - < kleiner dan
  - >= groter dan of gelijk aan
  - <= kleiner dan of gelijk aan

DE HAAGSE HOGESCHOOL

Voorwaarden

- De voorwaarde `a == b` levert 1 op als a gelijk is aan b, anders levert de voorwaarde 0 op.
- Deze voorwaarden komen voor bij beslissen en herhalen:

```
if (a == b) { … }        /* beslissen */
while (a > b) { … }      /* herhalen */
```

- Als de voorwaarde waar is, wordt de beslissing of herhaling uitgevoerd.

DE HAAGSE HOGESCHOOL

Logische operatoren

- Voorwaarden kunnen worden verbonden met logische operatoren:
- && en
- || of

```
if (a == b && c == d) { … }
while (a > b || a > c) { … }
```

- Er zijn geen haakjes nodig want == en > hebben een hogere prioriteit dan && en ||.

DE HAAGSE HOGESCHOOL

Logische operatoren

- Voorbeeld: berekenen van een schrikkeljaar:
- Jaartal is deelbaar door 4 maar niet deelbaar door 100, maar weer wel deelbaar door 400:

```
int year, leap;
scanf("%d", &year);
leap = (year % 4 == 0 && year % 100 != 0) || year % 400 == 0;
```

- Merk op dat `leap` 0 of 1 is.

DE HAAGSE HOGESCHOOL

Logische operatoren

- Voorbeeld:

```
if (leap == 1) {
   printf("%d is een schikkeljaar\n", year);
}
if (leap == 0) {
   printf("%d is een geen schikkeljaar\n", year);
}
```

Logische operatoren

- Korter (maar onduidelijker):

```
if (leap) {
   printf("%d is een schikkeljaar\n", year);
}
if (!leap) {
   printf("%d is een geen schikkeljaar\n", year);
}
```

Snelle toekenningen

- Met één verhogen: ++
- Met één verlagen: --

- Er zijn twee vormen: postfix en prefix:

```
i++;        i--;
++i;        --i;
```

- Postfix verhoogt/verlaagt ná gebruik, prefix verlaagt/verhoogd vóór gebruik
- Handig bij gebruik van herhalingen en arrays..

DE HAAGSE HOGESCHOOL

Snelle toekenningen

- Variabele aanpassen:
- Met toekenningsoperatoren: `+=` `-=` `*=` `/=` `%=`

- Dus: `x = x * 3` → `x *= 3`

  `y = y + 2` → `y += 2`

DE HAAGSE HOGESCHOOL

Prioriteiten (eenvoudige versie)

Operator	Associativiteit
( )	Links naar rechts
! + - (unair) ++ --	Rechts naar links
* / %	Links naar rechts
+ -	Links naar rechts
< <= > >=	Links naar rechts
== !=	Links naar rechts
&&	Links naar rechts
||	Links naar rechts
= *= /= %= += -=	Rechts naar links

DE HAAGSE HOGESCHOOL

Lokale variabelen

- Een variabele die binnen de functie main is gedefinieerd is *lokaal*.

```
int main(void)
{
   /* a is nog niet zichtbaar */
   int a;
   /* a is zichtbaar en te gebruiken */
}
/* a is niet meer zichtbaar */
```


DE HAAGSE HOGESCHOOL

Globale variabelen

- Een variabele die buiten alle functies is gedefinieerd, is *globaal*.

```
int a;
int main(void)
{
  /* a is zichtbaar en te gebruiken */
}
/* a is zichtbaar en te gebruiken */
```


DE HAAGSE HOGESCHOOL

Verbergen van globale variabelen

- Een variabele die buiten alle functies is gedefinieerd, is *globaal*.

```
int a;
int main(void)
{
  /* globale a is zichtbaar en te gebruiken */
  int a;
  /* lokale a is zichtbaar en te gebruiken */
}
/* globale a is zichtbaar en te gebruiken */
```


DE HAAGSE HOGESCHOOL

Vaste lengte datatypes

- Een signed int kan uit 32 bits (PC) of uit 16 bits bestaan (bv. Arduino Uno)
- Voor een exacte grootte (in bits) zijn er nieuwe datatypes
- Deze moeten geladen worden met `stdint.h`

`int8_t`	een signed integer met precies 8 bits
`int16_t`	een signed integer met precies 16 bits
`int32_t`	een signed integer met precies 32 bits
`int64_t`	een signed integer met precies 64 bits

- Vooral handig op microcontrollers

DE HAAGSE HOGESCHOOL

Vaste lengte datatypes

- Een unsigned int kan uit 32 bits (PC) of uit 16 bits bestaan (bv. Arduino Uno)
- Voor een exacte grootte (in bits) zijn er nieuwe datatypes
- Deze moeten geladen worden met `stdint.h`

`uint8_t`	een unsigned integer met precies 8 bits
`uint16_t`	een unsigned integer met precies 16 bits
`uint32_t`	een unsigned integer met precies 32 bits
`uint64_t`	een unsigned integer met precies 64 bits

- Vooral handig op microcontrollers

DE HAAGSE HOGESCHOOL

let’s change