

leren. durven. doen.



Leren Programmeren

C# BASIS

C#-taal Basis - Overzicht

C# - Programmastructuur

C# - Syntax

C# - Variabelen

C# - Constants

C# - Data Types

C# - Type Conversie

C# - Operators

C# - Decision Making

- C# - Loops
- C# - Encapsulation
- C# - Methods
- C# - Nullables
- C# - Arrays
- C# - Strings
- C# - Struct
- C# - Enums

leren. durven. doen.



Data types

**LEREN PROGRAMMEREN
BASISBEGRIPPEN**

Overzicht C# Data Types (Herhaling)

Type	Represents	Range	Default Value
<code>bool</code>	Boolean value	True or False	False
<code>byte</code>	8-bit unsigned integer	0 to 255	0
<code>char</code>	16-bit Unicode character	U +0000 to U +ffff	'\0'
<code>decimal</code>	128-bit precise decimal values with 28-29 significant digits	(-7.9 x 10 ²⁸ to 7.9 x 10 ²⁸) / 10 ⁰ to 28	0.0M
<code>double</code>	64-bit double-precision floating point type	(+/-)5.0 x 10 ⁻³²⁴ to (+/-)1.7 x 10 ³⁰⁸	0.0D
<code>float</code>	32-bit single-precision floating point type	-3.4 x 10 ³⁸ to + 3.4 x 10 ³⁸	0.0F
<code>int</code>	32-bit signed integer type	-2,147,483,648 to 2,147,483,647	0
<code>long</code>	64-bit signed integer type	-923,372,036,854,775,808 to 9,223,372,036,854,775,807	0L
<code>sbyte</code>	8-bit signed integer type	-128 to 127	0
<code>short</code>	16-bit signed integer type	-32,768 to 32,767	0
<code>uint</code>	32-bit unsigned integer type	0 to 4,294,967,295	0
<code>ulong</code>	64-bit unsigned integer type	0 to 18,446,744,073,709,551,615	0
<code>ushort</code>	16-bit unsigned integer type	0 to 65,535	0

Data Type - Karakteristieken

Data type heeft:

- Naam (C# keyword or .NET type)
- Grootte (hoeveel geheugen wordt voozien)
- Default waarde

Bijvoorbeeld:

Geheel getal in C#

Naam datatype: **int**

Grootte: 32 bits (4 bytes)

Default waarde: 0

De built-in
primitieve C#
types zijn
synoniemen
voor de
voorgedefinieerde
types in
.Net **System**
namespace.

C# type	.NET type
<code>bool</code>	System.Boolean
<code>byte</code>	System.Byte
<code>sbyte</code>	System.SByte
<code>char</code>	System.Char
<code>decimal</code>	System.Decimal
<code>double</code>	System.Double
<code>float</code>	System.Single
<code>int</code>	System.Int32
<code>uint</code>	System.UInt32
<code>long</code>	System.Int64
<code>ulong</code>	System.UInt64
<code>object</code>	System.Object
<code>short</code>	System.Int16
<code>ushort</code>	System.UInt16
<code>string</code>	System.String

Data Type string

Werken met String Data

Afdrukken van string variabelen:

```
string firstName, lastName;  
firstName = "Rowan";  
lastName = "Atkinson";
```

```
//Voorbeeld van string “concatenation” (aan elkaar “plakken” van  
//string variabelen
```

```
string fullName = firstName + " " + lastName;
```

```
//Andere manieren om variabelen naar de console te schrijven:
```

```
//via placeholders {0} {1},...
```

```
Console.WriteLine("Full Name: {0} {1}", firstName, lastName);
```

```
/*via string interpolation:
```

(opm: werkt niet met compiler .NET 4.7.2 in
<https://dotnetfiddle.net/> Kies bv .NET Core 3.1 als compiler)*/

```
Console.WriteLine($"Full Name: {firstName} {lastName}");
```

Escape Characters

Hoe data moet geprint worden naar output stream
Elke escape character begint met een backslash, gevolgd
met een speciek teken.

Voorbeelden:

```
//Example of backslash in a string literal
Console.WriteLine("C:\\\\MyApp\\\\bin\\\\Debug");
//Example of double quote in a string literal
string message = "Last night I dreamt of \"San Pedro\"";
Console.WriteLine(message);
string cityName = message.Substring(23);
Console.WriteLine(cityName);
```

Overzicht Escape sequences

Escape Sequence	Name	Description
\b	Backspace	Takes the cursor back
\t	Horizontal Tab	Takes the cursor to the next tab stop
\n	New line	Takes the cursor to the beginning of the next line
\v	Vertical Tab	Performs a vertical tab
\f	Form feed	
\r	Carriage return	Causes a carriage return
\"	Double Quote	Displays a quotation mark ("")
\'	Apostrophe	Displays an apostrophe ('')
\?	Question mark	Displays a question mark
\\\	Backslash	Displays a backslash (\)
\0	Null	Displays a null character

String en Equality (gelijkheid)

String equality is case-sensitive

- "Hello!" <> "HELLO!"

```
static void StringEquality()
{
    Console.WriteLine("=> String equality:");
    string s1 = "Hello!";
    string s2 = "Yo!";
    Console.WriteLine("s1 = {0}", s1);
    Console.WriteLine("s2 = {0}", s2);
    Console.WriteLine();

    // Test these strings for equality.
    Console.WriteLine("s1 == s2: {0}", s1 == s2);
    Console.WriteLine("s1 == Hello!: {0}", s1 == "Hello!");
    Console.WriteLine("s1 == HELLO!: {0}", s1 == "HELLO!");
    Console.WriteLine("s1 == hello!: {0}", s1 == "hello!");
    Console.WriteLine("s1.Equals(s2): {0}", s1.Equals(s2));
    Console.WriteLine("Yo.Equals(s2): {0}", "Yo!".Equals(s2));
    Console.WriteLine();
}
```

Werken met String Data

String methoden

- Length (*geeft de lengte van string*)
- CompareTo() (*vergelijkt twee strings*)
- Contains() (*controleert of string een bepaalde substring bevat*)
- Equals() (*controleert of twee strings identieke data bevatten*)
- ToUpper() (*zet de string in uppercase*)
- ToLower() (*zet de string in lowercase*)
- SubString() (*om een stukje string uit een andere string te extracten*)
- Convert.ToString() en ToString() (*omzetten van een ander data type naar string*)

Converteren van Strings naar getallen

Numerieke types beschikken over een method **Parse(...)** om de numerieke waarde uit een string te halen:

int.Parse(string) – string → int

double.Parse(string) – string → double

float.Parse(string) – string → float

Voorbeelden:

```
string s1 = "123";
int i = int.Parse(s1);
string s2 = "18.45";
double d = double.Parse(s2);
float f = float.Parse(s2);
Console.WriteLine(i);
Console.WriteLine(d);
Console.WriteLine(f);
```

Voorbeelden werken met string

Length (geeft de lengte van string)

Opm: geen methode maar property (eigenschap)

CompareTo() (vergelijkt twee strings)

Geeft -1, 1 of 0 terug:

- -1 (indien alfabetisch eerst)
- 1 (indien alfabetisch laatst)
- 0 (indien gelijk)

```
string tekst = "abcde";
Console.WriteLine(tekst.Length);
```

```
string tekst1 = "ab";
string tekst2 = "cd";
Console.WriteLine(tekst1.CompareTo(tekst2));
```

Contains() (controleert of string een bepaalde substring bevat)

Geeft true of false terug

```
string tekst1 = "abcd";
string tekst2 = "bc";
Console.WriteLine(tekst1.Contains(tekst2));
```

Equals() (controleert of twee strings identieke data bevatten)

Geeft true of false terug

```
string tekst1 = "abcd";
string tekst2 = "bc";
Console.WriteLine(tekst1.Equals(tekst2));
```

Voorbeelden - Vervolg

ToUpper() (*zet de string in uppercase*)

ToLower() (*zet de string in lowercase*)

Substring() (*om een een stukje string uit een andere string te extracten*)

```
string tekst1 = "abcd";
Console.WriteLine(tekst1.ToUpper());
string tekst2 = "ABCD";
Console.WriteLine(tekst2.ToLower());
string tekst3 = "dit is een stukje tekst";
Console.WriteLine(tekst3.Substring(0,5));
```

Convert.ToString() en ToString()

(*om een ander datatype om ze zetten naar een string*)

```
double bedrag=20.5;
string tekstBedrag1 = Convert.ToString(bedrag);
Console.WriteLine(tekstBedrag1);
string tekstBedrag2 = bedrag.ToString();
Console.WriteLine(tekstBedrag2);
```

Oefeningen op Strings

1. Schrijf een programma dat de volgende getallen naar de console schrijft (elk getal op een nieuwe lijn:
5 6 3 11 8 13

2.A Schrijf een programma dat de huidige datum naar de console schrijft

Tip: DateTime.Now geeft de huidige datum en uur terug

2. B Extract het jaartal uit de vorige datum

Data types

Integer Data Type

Wat zijn de integer types?

Integer types:

Stellen gehele getallen voor

Kunnen signed (met teken, ook negative) of unsigned (enkel positieve) zijn

Het bereik (min en max waarden) hangt af van de hoeveelheid gereserveerde geheugenruimte

De default waarde van integer types:

0 – voor integer types, behalve

0L – voor het **long** type



Overzicht Integer Types

De Integer types (gehele getallen):

sbyte (-128 to 127): signed 8-bit

byte (0 to 255): unsigned 8-bit

short (-32,768 to 32,767): signed 16-bit

ushort (0 to 65,535): unsigned 16-bit

int (-2,147,483,648 to 2,147,483,647): signed 32-bit

uint (0 to 4,294,967,295): unsigned 32-bit

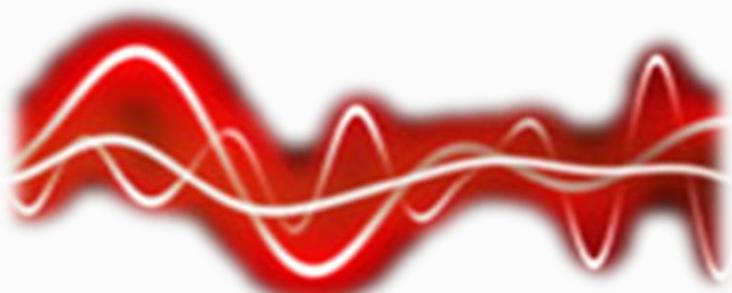
Integer types – Voorbeeld

Afhankelijk van het nodige bereik, wordt het gepaste Integer datatype gekozen:

```
// Declare some variables
byte centuries = 20;
ushort years = 2000;
uint days = 730480;
ulong hours = 17531520;
// Print the result on the console
Console.WriteLine(centuries + " centuries are " + years
+ " years, or " + days + " days, or " + hours + "
hours.");
// Console output:
// 20 centuries are 2000 years, or 730480 days, or
17531520
// hours.
ulong maxValue = UInt64.MaxValue;
Console.WriteLine(maxValue); // 18446744073709551615
```

Integer Types

Demo



01101101101101101
11011101101100101001001110

Data types

Floating-Point en Decimal Types

Wat zijn Floating-Point Types?

Floating-point types:

- Zijn presentaties van reële getallen (komma getallen)
- Kunnen signed (met teken) of unsigned (zonder teken) zijn
- Hebben een bepaald bereik en verschillende precisie (aantal cijfers na de komma), afhankelijk van de hoeveelheid gereserveerde geheugen
- Kunnen abnormale resultaten geven in berekeningen!

Floating-Point Types (komma-getallen)

Floating-point types:

`float` ($\pm 1.5 \times 10^{-45}$ to $\pm 3.4 \times 10^{38}$): 32-bits, precisie van 7 cijfers

`double` ($\pm 5.0 \times 10^{-324}$ to $\pm 1.7 \times 10^{308}$): 64-bits, precisie van 15-16 cijfers

De default value van floating-point types:

0.0F voor `float` type

0.0D voor `double` type

Precisie van PI – Voorbeeld

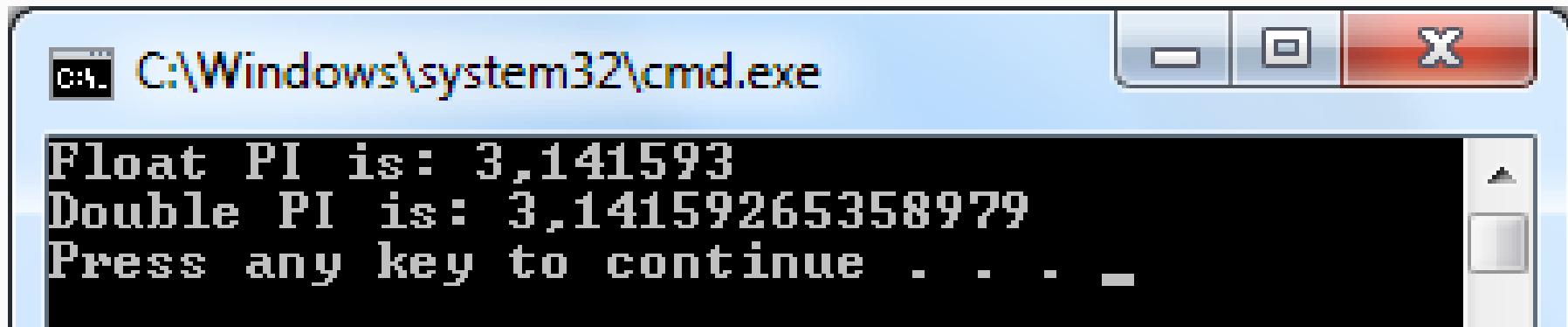
Hieronder zie je het verschil in precisie tussen **float** en **double**:

```
float floatPI = 3.141592653589793238f;  
double doublePI = 3.141592653589793238;  
Console.WriteLine("Float PI is: {0}", floatPI);  
Console.WriteLine("Double PI is: {0}", doublePI);
```

Bemerk de suffix "**f**" in de eerste lijn.

Reële getallen worden standaard als **double** Geïnterpreteerd!

Deze moet **expliciet** geconverteerd worden naar **float**



Abnormale resultaten bij Floating-Point berekeningen

Soms zijn er abnormale resultaten bij het gebruik van floating-point getallen

Het vergelijken van komma getallen kan niet met de `==` operator

Voorbeeld:

De suffix f is voor float types, maar de compiler aanvaardt dat je deze aan het type double toekent, want double heeft grotere precisie dan float. Er zijn echter precisie-problemen wanneer je deze dan optelt, waardoor onderstaande vergelijking false teruggeeft:

```
double a = 1.0f;
double b = 0.33f;
double sum = 1.33f;
bool equal = (a + b == sum); // False!!!
Console.WriteLine("a+b={0}  sum={1}  equal={2}",
    a+b, sum, equal);
```

Decimal Type (komma-getallen)

Er is een decimal type in C#:

decimal ($\pm 1,0 \times 10^{-28}$ to $\pm 7,9 \times 10^{28}$): 128-bits, precisie van 28-29 cijfers

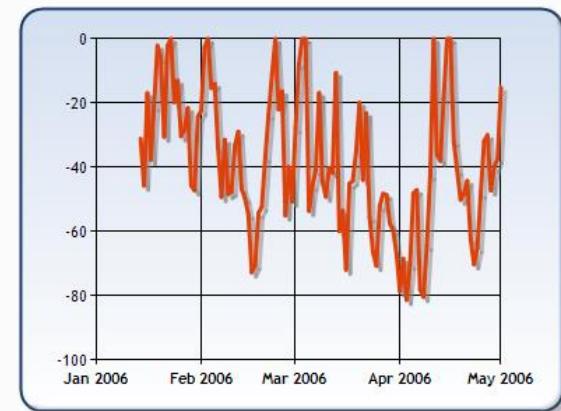
Wordt gebruikt voor wetenschappelijke berekeningen

Geen afrondingsfouten

Bijna geen verlies van precisie

De standaard waarde van een **decimal** type is:

0.0M (M is suffix voor decimale getallen)





Floating-Point Data Types

Demo



Character Type



Data Type Character

character data type:

Voorstelling van symbolische informatie

Declaratie met **char** sleutelwoord

Geeft elk symbool een overeenkomstig geheel getal

Standaard waarde = '\0'

Neemt 16 bits geheugen in

A a B b C c Đ d E
N Θ Y Δ Ξ Ж Й ڦ ڻ
س ع ڏ あ に サ
タ 𩷣 𩷧 𩷩 𩷪 𩷫

Characters en Codes

Elk symbool heeft zijn unieke Unicode code

Voorbeeld:

```
char symbol = 'a';
Console.WriteLine("The code of '{0}' is: {1}",
symbol, (int) symbol);
symbol = 'b';
Console.WriteLine("The code of '{0}' is: {1}",
symbol, (int) symbol);
symbol = 'A';
Console.WriteLine("The code of '{0}' is: {1}",
symbol, (int) symbol);
```



Character Data Type Demo

Boolean Type



Data Type Boolean

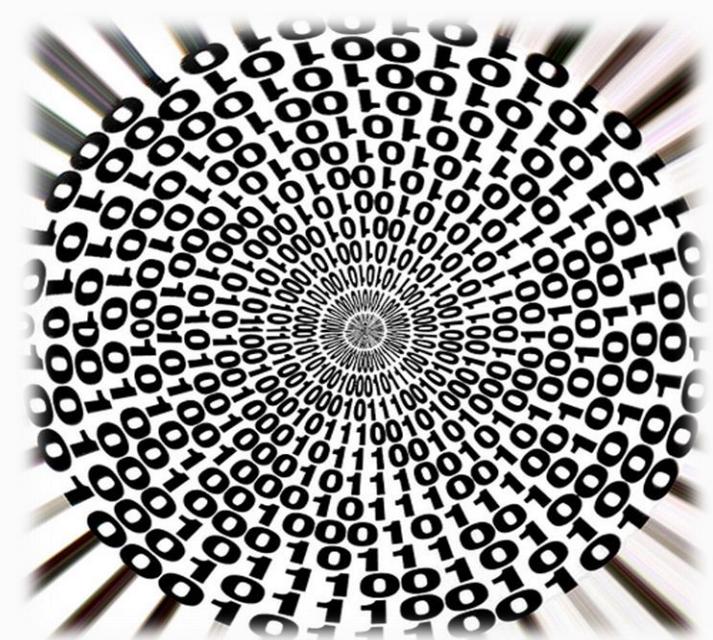
Boolean data type:

Wordt gedeclareerd via `bool` sleutelwoord

Heeft 2 mogelijke waarden: `true` en `false`

Is nuttig in logische uitdrukkingen (expressions)

De standaard waarde is `false`



Boolean Waarden – Voorbeeld

Voorbeeld van boolean variabelen die de waarden **true** of **false** aannemen:

```
int a = 1;  
int b = 2;  
bool greaterAB = (a > b);  
Console.WriteLine(greaterAB); // False  
bool equalA1 = (a == 1);  
Console.WriteLine(equalA1); // True
```

Werken met boolean variabelen



Vergelijken van waarden: geven true of false als resultaat (boolean)

Operator	Beschrijving	Voorbeeld
<code>==</code>	Is gelijk aan (waarde)	<code>age == 18</code>
<code>!=</code>	Is verschillend van	<code>day != "Monday"</code>
<code>></code>	Is groter dan	<code>salary > 9000</code>
<code>>=</code>	Is groter dan of gelijk aan	<code>salary >=9000</code>
<code><</code>	Is kleiner dan	<code>age < 18</code>
<code><=</code>	Is kleiner dan of gelijk aan	<code>age <= 18</code>

Vergelijken – voorbeelden:

voorbeeld: `int x = 5;`

Operatör	Beschrijving	Vergelijkin g	Resultaat
==	Is gelijk aan (waarde)	$x == 8$	false
		$x == 5$	true
!=	Is niet gelijk aan	$x != 8$	true
>	Is groter dan	$x > 8$	false
<	Is kleiner dan	$x < 8$	true
>=	Is groter dan of gelijk aan	$x >= 8$	false
<=	Is kleiner dan of gelijk aan	$x <= 8$	true

Voorbeelden vergelijken van variabelen

Oefening: Welke resultaten geven de volgende vergelijkingen?

```
int getal1 = 10;  
int getal2 = 5;
```

```
Console.WriteLine(getal1 == getal2);  
Console.WriteLine(getal1 != getal2);  
Console.WriteLine(getal1 > getal2);  
Console.WriteLine(getal1 >= getal2);  
Console.WriteLine(getal1 < getal2);  
Console.WriteLine(getal1 <= getal2);
```

Condities:

Operator	Description	Example
==	equal to	if (day == "Monday")
>	greater than	if (salary > 9000)
<	less than	if (age < 18)

**Worden gebruikt om een conditie te testen
Het resultaat van een conditie is true of false**

Voorbeeld van gebruik: bij testen van een conditie

```
int age = 16;  
if (age < 18)  
{  
    Console.WriteLine("Te jong!");  
}
```

Condities in C#

Syntax conditie:

```
if (conditie) {  
    codeblok wordt uitgevoerd als conditie true is  
}
```

Voorbeeld van gebruik:

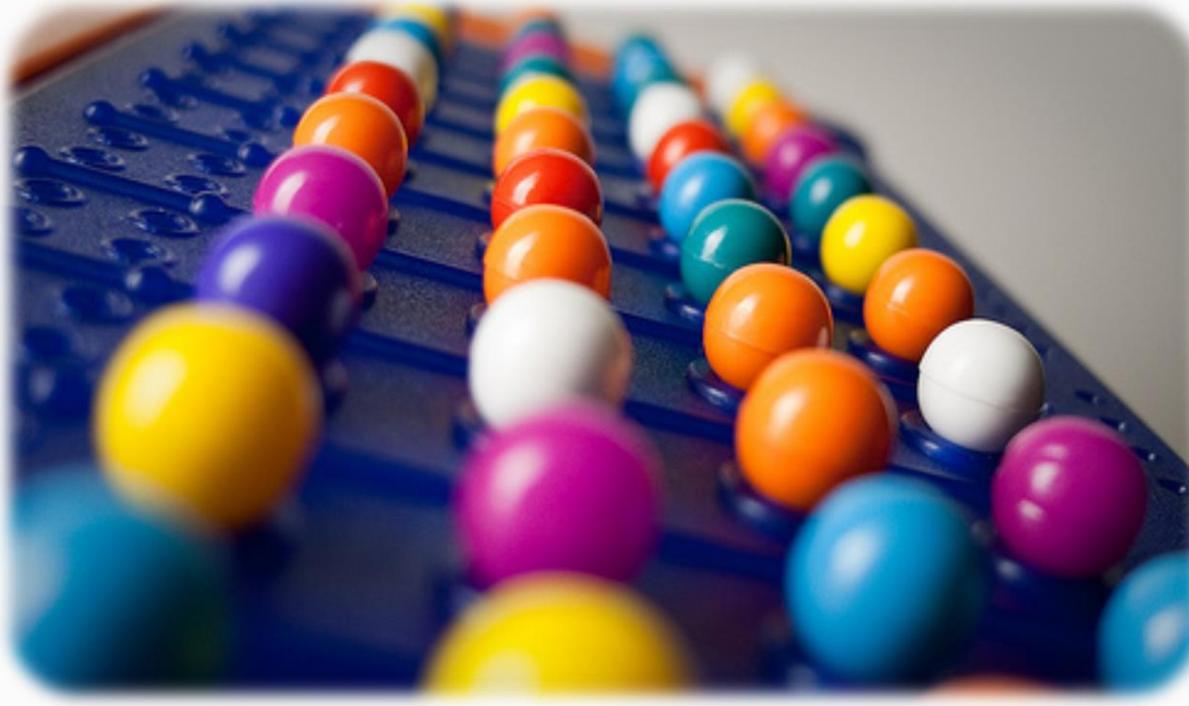
```
int hour = 16;  
if (hour < 18)  
{  
    Console.WriteLine("Good day");  
}
```

Oefening:

vraag hoe laat het is (geheel getal tussen 0 en 23) en lees het opgegeven uur in.

Tip: int uur = int.Parse(Console.ReadLine());

Schrijf "Goede morgen" naar de console indien het getal kleiner is dan 12



Boolean en condities

Demo

Boolean en condities

Oefeningen

Oefeningen C#:

https://www.w3schools.com/cs/exercise.asp?filename=exercise_booleans1

https://www.w3schools.com/cs/exercise.asp?filename=exercise_booleans2

https://www.w3schools.com/cs/exercise.asp?filename=exercise_conditions1

Oefening condities - 1

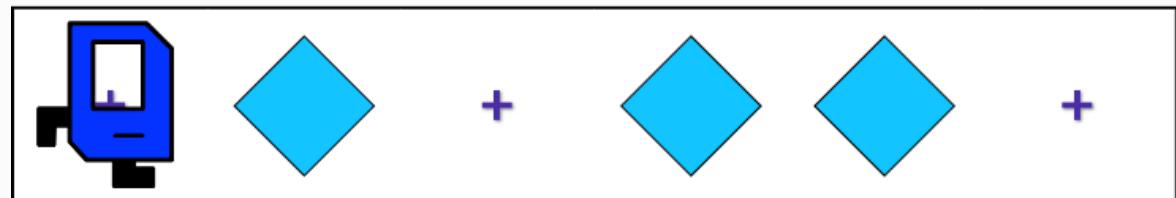
Oefeningen condities:

Oefeningen Karel/Karen op <https://www.kareltherobot.ch/karel.html>

Oefening 1: intro-12: Schrijf code die alle beeper opneemt

Gebruik de volgende conditie om de beepers op te nemen:

```
if (beepersPresent ())  
{  
    pickBeepers ();  
}
```



Oefening condities – 2

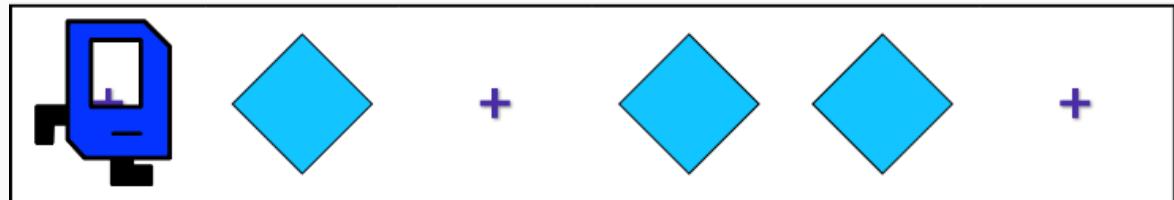
Oefeningen condities:

Oefeningen Karel/Karen op <https://www.kareltherobot.ch/karel.html>

Oefening 2: intro-12:

Wat gebeurt er indien je de condities uit Oefening 1 vervangt door de volgende code:

```
if(beepersPresent())
{
    pickBeeper();
}
else
{
    putBeeper();
}
```



Oefening condities – 3

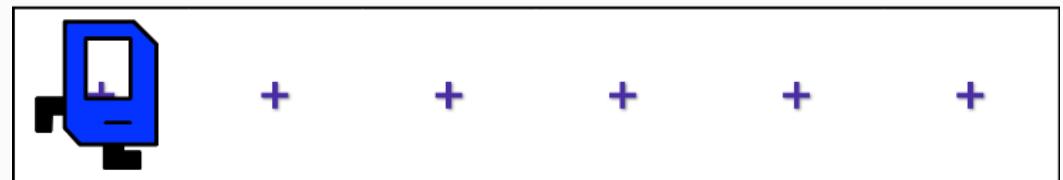
Oefeningen condities:

Oefeningen Karel/Karen op <https://www.kareltherobot.ch/karel.html>

Oefening 3: intro-01:

Schrijf de onderstaande code. Wat krijg je als resultaat?

```
function main() {  
  
    while (frontIsClear())  
    {  
        move();  
    }  
}
```



Oefening condities – 4

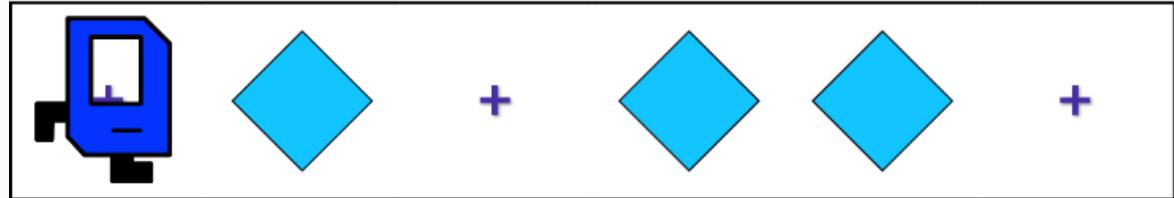
Oefeningen condities:

Oefeningen Karel/Karen op <https://www.kareltherobot.ch/karel.html>

Oefening 4: intro-12:

Vervang de code door de volgende code. Wat krijg je als resultaat?

```
function main() {  
  
    while(frontIsClear())  
    {  
        if(beepersPresent())  
        {  
            pickBeeper();  
        }  
        else  
        {  
            putBeeper();  
        }  
        move();  
    }  
}
```



Oefening condities – 5

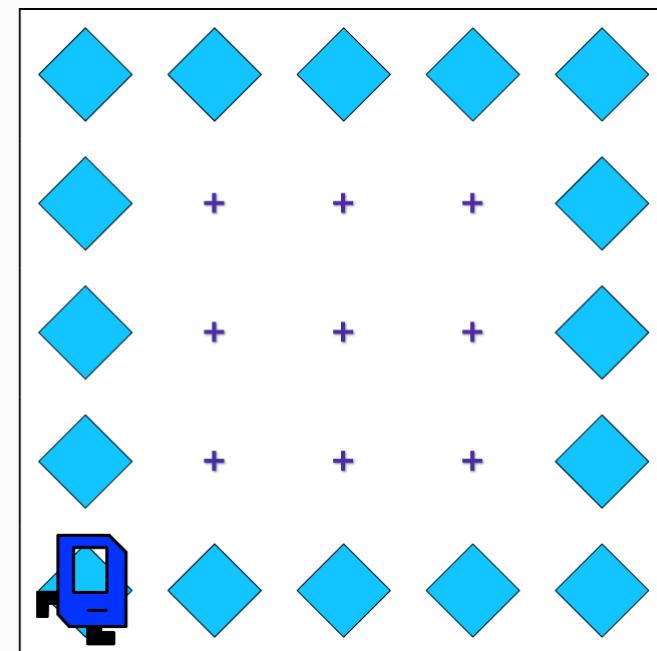
Oefeningen condities:

Oefeningen Karel/Karen op <https://www.kareltherobot.ch/karel.html>

Oefening 5: intro-06:

Schrijf de volgende code. Wat krijg je als resultaat? Pas de code aan zodat alle beepers worden opgenomen

```
function main() {  
    repeat(4)  
    {  
        while(frontIsClear())  
        {  
            move();  
        }  
        turnLeft();  
    }  
}
```



Oefening condities – 6

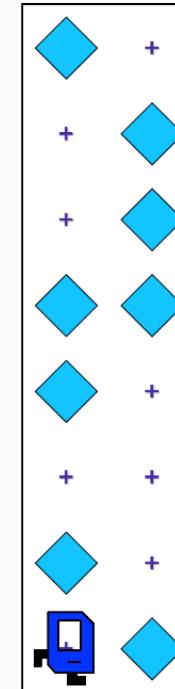
Oefeningen condities:

Oefeningen Karel/Karen op <https://www.kareltherobot.ch/karel.html>

Oefening 6: intro-13:

Schrijf de volgende code. Wat krijg je als resultaat? Pas de code aan zodat alle beepers worden opgenomen

```
function main() {
    turnLeft();
    while(frontIsClear())
    {
        move();
    }
    turnRight();
    move();
    turnRight();
    while(frontIsClear())
    {
        move();
    }
}
```



Oefening condities – 7

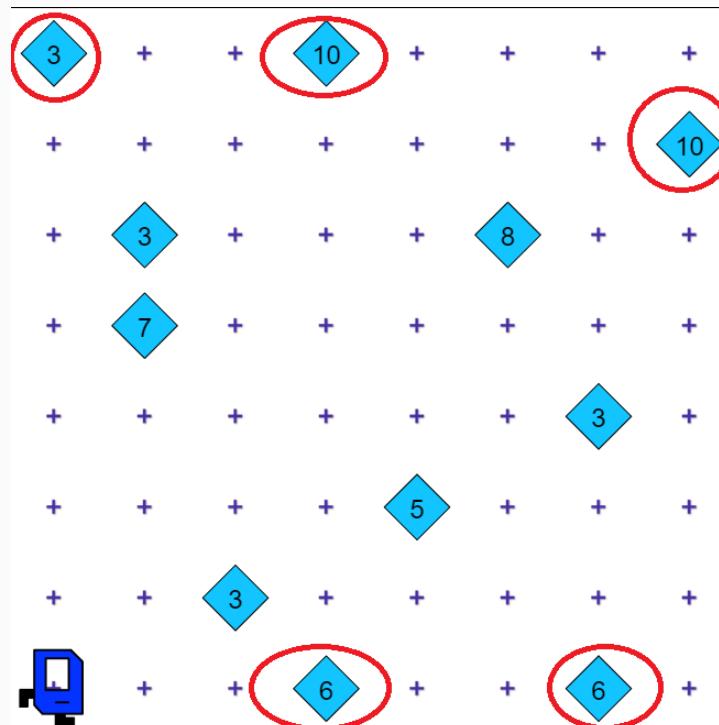
Oefeningen condities:

Oefeningen Karel/Karen op <https://www.kareltherobot.ch/karel.html>

Oefening 7: beepers-8x8:

Schrijf de volgende code. Wat krijg je als resultaat? Pas de code aan zodat alle beepers langs de randen worden opgenomen

```
function main() {  
    repeat (4)  
    {  
        while (frontIsClear ())  
        {  
            move ();  
            if (beepersPresent ())  
            {  
                pickBeeper ();  
            }  
            turnLeft ();  
        }  
    }  
}
```



Oefening condities – 8

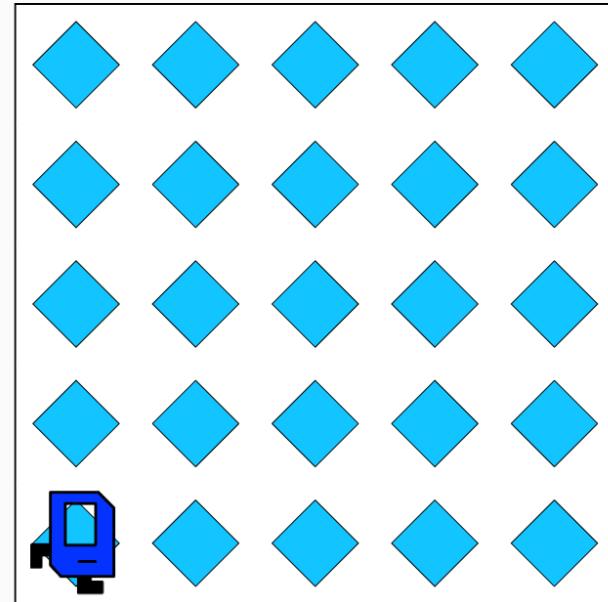
Oefeningen condities:

Oefeningen Karel/Karen op <https://www.kareltherobot.ch/karel.html>

Oefening 8: intro-08:

Schrijf de volgende code. Wat krijg je als resultaat? Pas de code aan zodat alle beepers worden opgenomen

```
function main() {
    repeat(2)
    {
        repeat(4)
        {
            move();
        }
        turnLeft();
        move();
        turnLeft();
        repeat(4)
        {
            move();
        }
        turnRight();
        move();
        turnRight();
    }
    repeat(4)
    {
        move();
    }
}
```



Oefening condities – 9

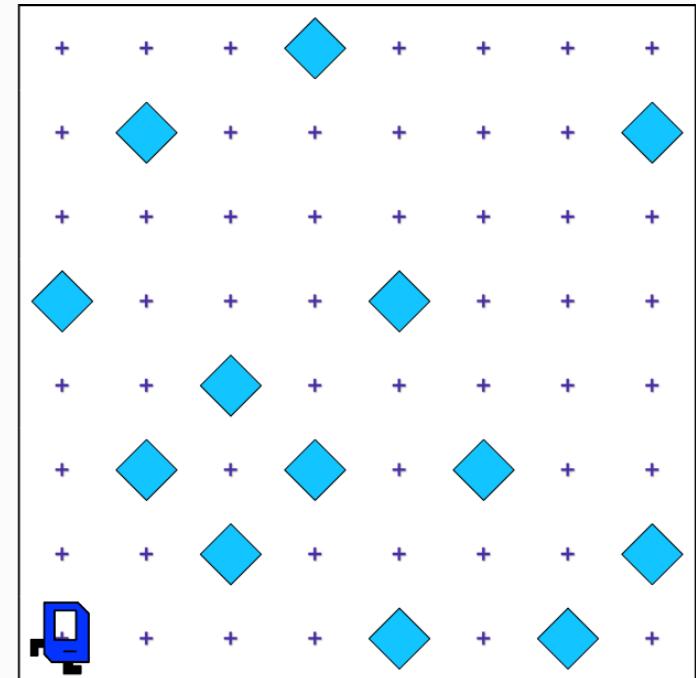
Oefeningen condities:

Oefeningen Karel/Karen op <https://www.kareltherobot.ch/karel.html>

Oefening 9: intro-14:

Schrijf de volgende code. Wat krijg je als resultaat? Pas de code aan zodat alle beepers worden opgenomen

```
function main() {  
    repeat(4)  
    {  
        turnLeft();  
        repeat(7)  
        {  
            move();  
        }  
        turnRight();  
        move();  
        turnRight();  
        repeat(7)  
        {  
            move();  
        }  
        turnLeft();  
        if (frontIsClear())  
        {  
            move();  
        }  
    }  
}
```



Oefening condities – 10

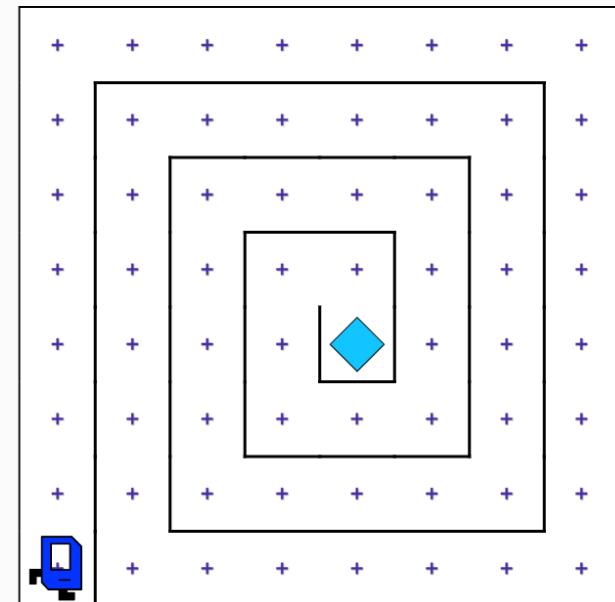
Oefeningen condities:

Oefeningen Karel/Karen op <https://www.kareltherobot.ch/karel.html>

Oefening 10: intro-15:

Schrijf de volgende code. Wat krijg je als resultaat? Pas de code aan zodat de beeper in het midden wordt opgenomen

```
function main() {  
    turnLeft();  
    while (noBeepersPresent())  
    {  
        //Hier code toevoegen  
        turnRight();  
    }  
    pickBeeper();  
}
```



leren. durven. doen.



Vragen?

DATA TYPES

leren. durven. doen.



REFERENTIES

[HTTPS://WWW.LEARNCS.ORG/](https://www.learnCS.org/)

FUNDAMENTALS OF COMPUTER PROGRAMMING WITH C#

© SVETLIN NAKOV & CO., 2013

[WWW.INTROPROGRAMMING.INFO](http://www.introprogramming.info)