Inhoud

[1 Inleiding barcodes 2](#_Toc468709608)

[1.1 Opbouw van barcode 2](#_Toc468709609)

[1.2 Herkennen van cijfers 3](#_Toc468709610)

[2 Zelf barcode maken met eigen algoritme 4](#_Toc468709611)

[2.1 Controlenummer bepalen 4](#_Toc468709612)

[2.2 Verantwoording gekozen algoritme 4](#_Toc468709613)

[2.3 ISBN-nummer 5](#_Toc468709614)

[2.4 Toepassingen ICT 5](#_Toc468709615)

[2.5 Handig rekenen met ISBN-nummers 6](#_Toc468709616)

[3 Inleiding QR-codes 7](#_Toc468709617)

[3.1 Mask van een QR-code 7](#_Toc468709618)

[3.2 Indeling van de QR-code 8](#_Toc468709619)

[3.3 Decoderen van een QR-code 8](#_Toc468709620)

[3.4 Veiligheidssleutel van QR-code 10](#_Toc468709621)

# Inleiding barcodes

Dit is een introductie van hoe barcodes werken. Het is belangrijk dat je de achtergrond informatie als leerkracht weet voordat je je leerlingen er zelf een laten maken. Er wordt een kleine samenvatting gegeven van de EAN-13 barcode die in Europa wordt gebruikt. Dit is een barcode met 13 cijfers waarvan 1 uiterst links staat en de andere 12 bevinden zich onder de barcode. Er is een verband tussen de cijfers die onder de barcode staan en de streepjes die de barcode vormen.

## Opbouw van barcode

De barcode is opgebouwd uit 95 kleine even grote kolommen die je niet met het blote oog kan zien. Een streepje van de barcode is dan in feite één ingekleurde kolom. Een dikkere streep zijn dan meerdere aanliggende kolommen die zwart gekleurd zijn. Een open witte ruimte zijn dan aanliggende kolommen die wit gekleurd zijn.

Als je met een lezer de barcode zou scannen, scant de lezer al die 95 kolommen en controleert welke kolom licht absorbeert (zwart) of reflecteert (wit). Afhankelijk of licht wordt gereflecteerd of geabsorbeerd, registreert de computer het cijfer 0 of 1. Scant de lezer een zwart streepje, registreert de computer dit als een 1. Omgekeerd als de lezer een witte kolom scant, registreert de computer dit als een 0. Iedere barcode begint en eindigt met dezelfde lijntjes, die zijn trouwens ook altijd iets langer dan de rest van de lijntjes. Dit is omdat deze lijntjes het begin en het einde van barcode weergeven.

De barcode begint met een lijn, dan een opening en dan weer een lijn. Dit is ook op het einde het geval. De computer leest het begin en einde dus als 101.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nummer** | 1 | 0 | 1 | … | 1 | 0 | 1 |
| **Kolom** | Kolom 1 | Kolom 2 | Kolom 3 | … | Kolom 93 | Kolom 94 | Kolom 95 |
| **Kleur** |  |  |  | … |  |  |  |

De cijfers die onder de barcode staan, zijn meestal in 2 groepen van 6 gedeeld. De scheiding van de twee groepen gebeurd ook altijd door dezelfde lijntjes. De computer registreert het midden van de barcode altijd als 01010.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nummer** | … | 0 | 1 | 0 | 1 | 0 | … |
| **Kolom** | … | 46 | 47 | 48 | 49 | 50 | … |
| **Kleur** | … |  |  |  |  |  | … |

De 95 kolommen worden dus ook in groepen gedeeld. 11 van alle kolommen zijn standaard hetzelfde en duiden samen het begin, midden en einde van de barcode aan. Dan worden de 84 resterende kolommen verdeeld in 12 even grote groepen. In totaal worden er 95 kolommen dus in 15 groepen verdeeld.

## Herkennen van cijfers

Op voorhand stelt iedere computer een binaire code gemeenschappelijk aan alle cijfers van 0 tot 9.

Mocht je barcode met 1 beginnen dan zou je vierde streepje van je barcode (eerste drie horen bij start barcode) een zwarte lijn zijn. Het volgende streepje is ook zwart en vormt dus samen met het eerste zwarte streepje een dikkere zwarte lijn. Dan komen twee witte streepjes gevolgd door weer twee zwart streepjes. De laatste lijn is weer wit. De in totaal vier zwarte lijnen met hun drie witruimtes symboliseren dus het eerste cijfer van de barcode, namelijk 1.

|  |
| --- |
| Eerste cijfer onder barcode is een 1 |
| **Nummer** | … | **0** | **0** | **1** | **1** | **0** | **0** | **1** | … |
| **Kolom** | … | 4 | 5 | 6 | 7 | 8 | 9 | 10 | … |
| **Kleur** | … |  |  |  |  |  |  |  | … |

Ieder cijfer heeft dus op voorhand een binaire code ingebouwd zodat de lezer deze kan herkennen.

De cijfers onder de barcode werden in twee groepen gedeeld. De cijfers die aan de rechterkant staan zijn anders geprogrammeerd dan de cijfers aan de linkerkant. Staat het cijfer 1 in de linker groep dan stelt de computer dit gelijk aan 0011001, maar staat er ook een 1 in de rechter groep dan stelt de computer dit gelijk aan 1100110.

|  |  |  |
| --- | --- | --- |
| Cijfer | Binaire code links | Binaire code rechts |
| **0** | **0001101** | **1110010** |
| **1** | **0011001** | **1100110** |
| **2** | **0010011** | **1101100** |
| **3** | **0111101** | **1000010** |
| **4** | **0100011** | **1011100** |
| **5** | **0110001** | **1001110** |
| **6** | **0101111** | **1010000** |
| **7** | **0111011** | **1000100** |
| **8** | **0110111** | **1001000** |
| **9** | **0001011** | **1110100** |

De nullen veranderen in enen en enen veranderen in nullen. Dit is belangrijk want het kan het geval zijn dat de barcode omgekeerd gescand wordt en het moet dan nog steeds hetzelfde product lezen. De computer zet de codes die beginnen met een 1 automatisch aan de rechterkant en codes die beginnen met een 0 automatisch aan de linker kant. Zo leest de computer steeds het juiste product ongeacht de houding van het product.

Er zijn ook altijd een oneven aantal enen en automatisch een even aantal nullen aanwezig in de code van de cijfers aan de linkerkant. Dit is een extra controle dat de computer de code juist leest. Automatisch draait het aantal enen en nullen om voor de cijfers aan de rechter kant.

# Zelf barcode maken met eigen algoritme

De barcode is gebaseerd op de cijfers die onder de barcode zelf staan. De EAN-13 code wordt in Europa gebruikt en beschikt over 12 cijfers die de barcode opstellen en een laatste cijfer dat geen nut heeft. Het nummer geeft weer welk land het product origineel komt.

Moest het voorvallen dat de barcode gescheurd is en je kan alleen de cijfers onder de barcode lezen, dan speelt het controlenummer wel een belangrijke rol. In de winkel moeten de kassiersters alle 13 cijfers ingeven en dus niet alleen de 12 cijfers die de barcode vormen. De computer controleert de ingegeven cijfers met een bepaald algoritme. Klopt het algoritme niet, dan heeft de kassierster de code fout ingegeven en zal er ook een foutmelding op het scherm komen te staan.

Om het algoritme te begrijpen moeten we eerst kijken hoe het controlenummer bepaald wordt.

## Controlenummer bepalen

Tel je alle cijfers op die op een even plaats staan (2, 4, 6 … ) en vermenigvuldig de som met 3.

Tel dan de overige cijfers met elkaar op. Tel de twee uitkomsten met elkaar op. Je verkrijgt nu een getal met 2 of 3 cijfers. Het controlenummer van de barcode is het getal dat je bij de uitkomst moet optellen om een veelvoud van 10 te bekomen.

Vb. Barcode (12 cijfers) =

 Berekening:

 Laatste cijfer:

Totale barcode:

Het controlenummer wordt dus bepaald vanuit de andere cijfers. De andere cijfers die onder de barcode staan hebben een betekenis en zijn dus op voorhand bepaald (land, soort product, …).

Het laatste cijfer kan je dus op een simpele manier berekenen door een ALGORITME toe te passen. Dit algoritme is op voorhand bepaald en als standaard ingevoerd voor alle Europese barcodes.

## Verantwoording gekozen algoritme

**Je kan de leerlingen de opdracht geven om zelf een algoritme te zoeken dat hetzelfde werkt.** Met andere woorden moet je de 3 en/of 1 in de berekening veranderen. Niet ieder getal is echter geldig. De reden dat het algoritme 3 en 1 gebruikt valt te verduidelijken met de tafel van 3 en 1.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **1** | **Laatste cijfer** |  | **3** | **Laatste cijfer** |  | **7** | **Laatste cijfer** |  | **9** | **Laatste cijfer** |
| **1** | 1 | 1 |  | 3 | 3 |  | 7 | 7 |  | 9 | 9 |
| **2** | 2 | 2 |  | 6 | 6 |  | 14 | 4 |  | 18 | 8 |
| **3** | 3 | 3 |  | 9 | 9 |  | 21 | 1 |  | 27 | 7 |
| **4** | 4 | 4 |  | 12 | 2 |  | 28 | 8 |  | 36 | 6 |
| **5** | 5 | 5 |  | 15 | 5 |  | 35 | 5 |  | 45 | 5 |
| **6** | 6 | 6 |  | 18 | 8 |  | 42 | 2 |  | 54 | 4 |
| **7** | 7 | 7 |  | 21 | 1 |  | 49 | 9 |  | 63 | 3 |
| **8** | 8 | 8 |  | 24 | 4 |  | 56 | 6 |  | 72 | 2 |
| **9** | 9 | 9 |  | 27 | 7 |  | 63 | 3 |  | 81 | 1 |
| **10** | 10 | 0 |  | 30 | 0 |  | 70 | 0 |  | 90 | 0 |

Kijken we naar het laatste cijfer dan zien we dat alle cijfers van 0 tot 9 terugkomen. Vermenigvuldigen we in het algoritme de sommen dus met 1 en/of 3 dan bekomen we altijd een getal dat de mogelijkheid biedt dat het controlenummer kan variëren van 0 tot 9. Zouden we het algoritme alleen met 2 nemen i.p.v. 1 en 3, dan kunnen we alleen maar even uitkomsten bekomen. Dit heeft als gevolg dat het controlenummer ook alleen maar even kan zijn (0, 2, 4, 6 of 8).

Nemen we getallen groter dan 10 om in het algoritme in te voeren moeten we alleen maar kijken naar het laatste cijfer van dit getal. Eindigt het getal met de cijfers 1, 3, 7 of 9 dan zal in het algoritme het controlenummer nog steeds kunnen variëren van 0 tot 9.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **2** | **Laatste cijfer** |  | **5** | **Laatste cijfer** |  | **17** | **Laatste cijfer** |  | **19** | **Laatste cijfer** |
| **1** | 2 | 2 |  | 5 | 5 |  | 17 | 7 |  | 19 | 9 |
| **2** | 4 | 4 |  | 10 | 0 |  | 34 | 4 |  | 38 | 8 |
| **3** | 6 | 6 |  | 15 | 5 |  | 51 | 1 |  | 57 | 7 |
| **4** | 8 | 8 |  | 20 | 0 |  | 68 | 8 |  | 76 | 6 |
| **5** | 10 | 0 |  | 25 | 5 |  | 85 | 5 |  | 95 | 5 |
| **6** | 12 | 2 |  | 30 | 0 |  | 102 | 2 |  | 114 | 4 |
| **7** | 14 | 4 |  | 35 | 5 |  | 119 | 9 |  | 133 | 3 |
| **8** | 16 | 6 |  | 40 | 0 |  | 136 | 6 |  | 152 | 2 |
| **9** | 18 | 8 |  | 45 | 5 |  | 153 | 3 |  | 171 | 1 |
| **10** | 20 | 0 |  | 50 | 0 |  | 170 | 0 |  | 190 | 0 |

Het is het makkelijkste om je controlenummer af te laten hangen zodat je totale som een veelvoud van 10 is. Met de bijkomende eigenschap dat 1, 3, 7 en 9 voor uitkomsten zorgen zodat je **altijd** met een cijfer van 0 tot 9 een veelvoud van 10 kan bereiken.

## ISBN-nummer

Een ander algoritme dat wordt gebruikt bij barcodes zijn de barcodes voor boeken of **ISBN-nummers**. Dit is een 10-cijferig waarvan het laatste cijfer het controlenummer is. Alweer geeft iedere groep cijfers iets weer, zoals de schrijver, uitgever, … . Het algoritme hier is het eerste cijfer te vermenigvuldigen met 10, het tweede cijfer met 9, het derde cijfer met 8, enz. De som van de producten zou altijd een veelvoud moeten zijn van 11.

Vb. barcode (9 cijfers) =

 Berekening =

 Missende cijfer = veelvoud 11 => 5

 Totale barcode =

## Toepassingen ICT

Op de website **stemleerkracht.be** kan je een werkblad in Excel vinden waar je de 12-cijferige code van een EAN-13 barcode kan ingeven en het controlenummer wordt automatisch gebruikt. Hetzelfde kan je vinden voor de ISBN codes.

## Handig rekenen met ISBN-nummers

Een trucje om het veelvoud van 11 te vinden is te kijken naar de twee buitenste cijfers van de uitkomst.

Nemen we 165 als voorbeeld dan zijn de twee buitenste cijfers 1 en 5. Nemen we die twee cijfers samen dan hebben we gevonden welk veelvoud 165 van 11 is, namelijk 15. Het middelste cijfer is dan de som van de twee buitenste cijfers.

Zoeken we het veelvoud van 11 dat het dichtste bij 193 ligt dan kijken we naar het eerste en tweede cijfer. Het eerste cijfer is een 1 wat dus wilt zeggen dat we 1 moeten aftrekken van 9 om het rechtse cijfer te bekomen. Dit is een 8, maar bij 193 is dit een 3. We moeten dus nog 5 optellen bij 193 om een veelvoud van 11 te bekomen. 5 is dus ons missende cijfer van de ISBN-nummer.

Opmerking, het kan het geval zijn dat je als uitkomst 188 uitkomt. Om dan een veelvoud van 11 te bekomen zou je 10 moeten bijtellen en 10 is dan je missende ‘cijfer’ in de ISBN-nummer. Maar 10 is een getal dat bestaat uit twee cijfers er is maar plaats voor één cijfer. Dit wordt opgelost door 10 als een X te noteren.

# Inleiding QR-codes

Een QR-code is vanzelfsprekend moeilijker te begrijpen dan een barcode. De QR-code is in feite een soort barcode maar dan in twee dimensies. Zo een code bestaat uit 6 hoofdcomponenten: de hoofdkaders (1), de dimensiecode (2), veiligheidssleutels (3), de ingevoerde inscriptiecode (4) die een begin (5) en eindpunt (6) heeft en tot slot de lengte van iedere byte (7).



(1)

(6)

(2)

(3)

(4)

(7)

(5)

(2)

## Mask van een QR-code

Om te kijken hoe de QR-code is opgebouwd moeten we eerst kijken naar welk formaat de code is gevormd in (2). Het formaat kan je op twee plaatsen aflezen. Het formaat kan je in 5 achtereenvolgende vakjes aflezen.

Voor onze QR-code kunnen we de 5 vakjes omzetten naar 11010 (wit is 0 en zwart is 1). We moeten nu het cijfer op de oneven plaatsen (1, 3 en 5) veranderen. Waarom, heeft te maken met de programmeertaal en hier zullen we niet verder op ingaan. We krijgen dus de code 01111. De eerste twee cijfers zijn ook niet van belang omdat dit ook iets weergeeft over de error correctie, namelijk hoeveel er in de QR-code aanwezig zijn. We negeren de eerste twee cijfers en we houden dus 111 over. Dit is wel belangrijk en geeft ons de ‘Mask’ ofwel de vorm van de QR-code.



## Indeling van de QR-code

Als we de QR-code even indelen in rijen en kolommen zien we dat onze code 21 rijen en 21 kolommen heeft. De rijen benoemen we met de letter *i* en de kolommen met de letter *j* en . Zo stelt dan *i1 x j1*  het vakje rechts in de onderhoek voor (we beginnen onderaan en rechts te tellen). Om na te gaan of we dat vakje nu moeten interpreteren als wit of zwart hangt van de ‘Mask’ af.

 Het kan goed zijn dat in de QR-code *i21 x j19*  zwart is, maar de software volgens de ‘Mask’ het leest als een wit vakje.

We vullen *i21 x j19*  in de formule in:

 en in de formule wilt zeggen dat je je uitkomst moet delen door 3 en 2 en de uitkomst is je rest. met het aantal keren dat 3, 5 deelt en ∈ ℕ de rest.

Zo is en

 en dit is deelbaar door 3 waardoor we . De volgende stap is te bepalen. 40 is al een veelvoud van 2 dus . We bekomen dus 0 als uitkomst. De software zal in de QR-code *i21 x j19*  als een zwart vakje interpreteren wat met in de ‘Mask’ met een 0 overeen komt.

Nemen we het vakje links ervan *i20 x j19*  dan vullen we dit ook in de formule in: . De software leest dit dus als een zwart vakje.

Alle vakjes die volgens de ‘Mask’ zwart zijn doen de kleur van de originele omwisselen. Enkele voorbeelden:

Is *i5 x j9* wit en valt dit samen met een zwart kadertje van de ‘Mask’, dan moet je het originele witte vakje veranderen naar een zwart vakje.

Is *i5 x j9* zwart en valt dit samen met een zwart kadertje van de ‘Mask’, dan moet je het originele zwarte vakje veranderen naar een wit vakje.

Is *i6 x j4* zwart en valt dit niet samen met een zwart vakje van de ‘Mask’, dan blijft dit vakje zwart.

Is *i6 x j4* wit en valt dit niet samen met een zwart vakje van de ‘Mask’, dan blijft dit vakje wit.

**Opmerking: Om te weten te komen wat de functie div betekend kan je dit opzoeken op stemleerkracht.be. Wil je minder rekenwerk kan je ook op stemleerkracht.be kijken hoe de verschillende ‘Masks’ de QR-code vullen.**

## Decoderen van een QR-code

Je kan achterhalen naar wat de QR-code je zal sturen door naar (5) te kijken in de rechter onderhoek. Je neemt van de hoek een vierkantje van 2x2 hokjes groot en vult de cijfers 1, 2, 4 en 8 in zoals hier is weergegeven.

|  |  |
| --- | --- |
| 1 | 2 |
| 4 | 8 |

Als een vakje in het zwart gekleurd is stelt de computer dit gelijk aan het cijfer 1. Is bijvoorbeeld enkel het cijfer 4 gekleurd dan komt het binaire cijfer 0100 hier mee overeen. Dit geeft ons de module weer hoe de QR-code is opgebouwd. Vanuit 0100 kunnen we afleiden dat in de QR-code is opgebouwd vanuit een 8-byte codering. Alleen deze zal besproken worden.

We kunnen een [kaart](http://www.sciencebuddies.org/science-fair-projects/project_ideas/CompSci_ASCII_Table.shtml) opvragen waarin alle mogelijke tekens staan die we met de 8-byte codering kunnen maken. Hierin staan verschillende tekens, letters (hoofd –en kleine letters) en cijfers. De 8-byte codering zegt ons ook dat om bijvoorbeeld kleine letter a in de QR-code te plaatsen, de computer een binaire code moet herkennen die 8 tekens lang is. Een kleine letter a komt dan met de code 0110 0001 overeen.

We weten nu dat de QR-code een 8-byte codering moet lezen d.w.z. dat de computer de vakjes van de QR-code leest in groepjes van 8, want een groepje van 8 komt dus overeen met een letter, cijfer of teken. We beginnen deze tekens te lezen boven het startvierkant in een rechthoek van 2x4 groot.

|  |  |
| --- | --- |
| 1 | 2 |
| 4 | 8 |
| 16 | 32 |
| 64 | 128 |

Om dus de kleine letter a in de QR-code te steken moet de computer dus 0110 0001 lezen. Er moeten dus 3 vakjes zwart gekleurd worden. Dit zijn de vakjes 1, 32 en 64. Dit moeten we optellen en dit geeft ons 97 als som. In de [kaart](http://www.sciencebuddies.org/science-fair-projects/project_ideas/CompSci_ASCII_Table.shtml) komt 97 inderdaad overeen met kleine letter a.

We zijn dus on de rechter onderhoek begonnen met het decoderen van de QR-code. Als de QR-code in 8-byte gecodeerd is moeten we dus QR-code dus steeds lezen in groepjes van 8 in de vorm van een 2x4 rechthoek. Op een bepaald moment gaat we helemaal tot de top geraakt zijn en zal het geval zich voordoen dat we geen groepje van 8 meer kunnen nemen. We lossen dit op door het groepje van 8 links te kantelen en de cijfers anders te schikken.



|  |  |  |  |
| --- | --- | --- | --- |
| 4 | 8 | 16 | 32 |
| 1 | 2 | 64 | 128 |

|  |  |
| --- | --- |
| 1 | 2 |
| 4 | 8 |
| 16 | 32 |
| 64 | 128 |

Nadat we de kader gekanteld hebben doen we dit nog een keer om terug naar onder toe te gaan. De cijfers worden terug normaal geschikt alleen nu van onder naar boven.

|  |  |  |  |
| --- | --- | --- | --- |
| 4 | 8 | 16 | 32 |
| 1 | 2 | 64 | 128 |

|  |  |
| --- | --- |
| 64 | 128 |
| 16 | 32 |
| 4 | 8 |
| 1 | 2 |

Komen we weer **onder aan** dan moeten we de kader opnieuw kantelen en de cijfers herschikken.

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 64 | 128 |
| 4 | 8 | 16 | 32 |

|  |  |
| --- | --- |
| 64 | 128 |
| 16 | 32 |
| 4 | 8 |
| 1 | 2 |

Dit blijf je herhalen totdat je alle letters, cijfers en/of tekens gehad hebt om je naar de juiste link te sturen. Wil je bijvoorbeeld een QR-code naar de website van de Youtube maken dat heb je 11 tekens nodig: y, o, u, t, u, b, e, ., c, o, m.

|  |  |  |
| --- | --- | --- |
| **Binair** | **Cijfer** | **Symbool** |
| 0111 1001 | 121 | y |
| 0110 1111 | 111 | o |
| 0111 0101 | 117 | u |
| 0111 0100 | 116 | t |
| 0111 0101 | 117 | u |
| 0110 0100 | 100 | b |
| 0110 0101 | 101 | e |
| 0010 1110 | 46 | . |
| 0110 0011 | 99 | c |
| 0110 1111 | 111 | o |
| 0110 1101 | 109 | m |

## Veiligheidssleutel van QR-code

Na de laatste symbool dat je nodig hebt om bv je website in te voeren volgt altijd een leeg 2x2 vierkant. Zo weet de lezer waar je ‘tekst’ eindigt. De rest van de QR-code zijn error correcties die volgens een wiskundige [veeltermfunctie](https://en.wikipedia.org/wiki/Reed%E2%80%93Solomon_error_correction) gevormd worden. Dit is veel meer complex en zal niet verder worden uitgelegd.