Les C-02: Werken met Programma Structuur Diagrammen

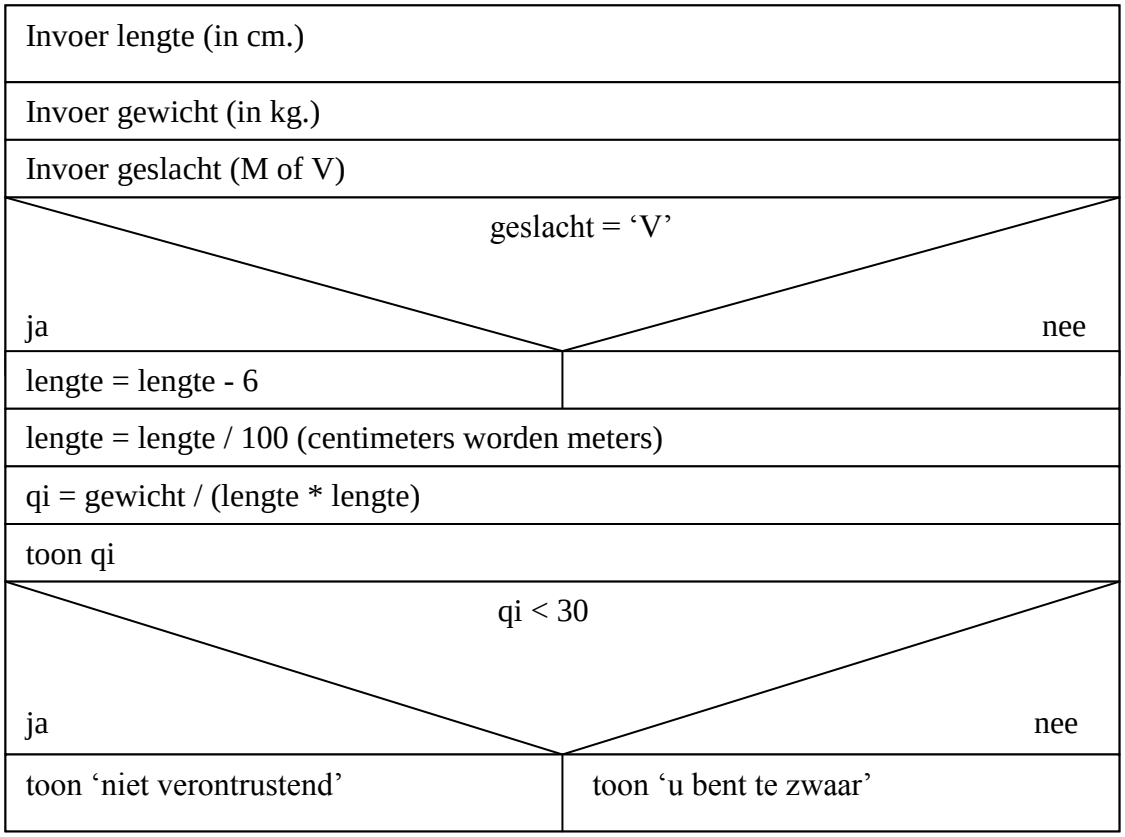
2.0 Inleiding

In deze lesbrief bekijken we een methode om een algoritme zodanig structuur te geven dat er gemakkelijk programmacode bij te schrijven is: bij deze methode worden de instructies uit het algoritme in een diagram (Programma Structuur Diagram, kort: PSD) opgenomen. Eerst bekijken we de bouwstenen van PSD's, daarna zal je zelf PSD's moeten maken.

2.1 Werken met keuzeblokken

Voor je van een algoritme een computerprogramma kan maken is het handig om te bedenken op welke manier de instructies achter elkaar moeten worden geplaatst. De structuur waarin we de instructies plaatsen kunnen we weergeven met **Programma Structuur Diagrammen** (PSD's)

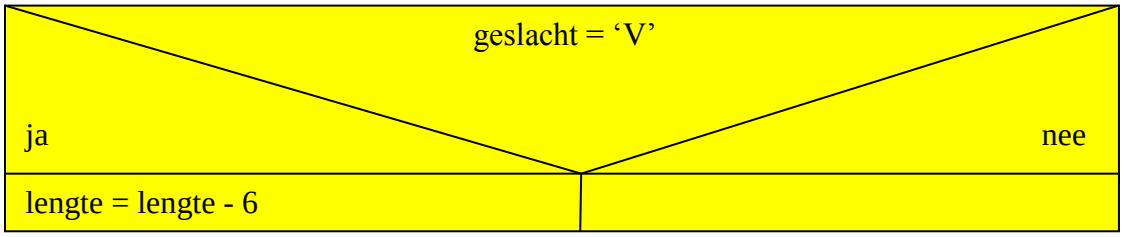
Hieronder zie je een voorbeeld van een Programma Structuur Diagram (PSD). In een PSD wordt weergegeven op welke wijze instructies stap voor stap worden verwerkt.



OPDRACHT

Opdracht 2.1
Beschrijf in eigen woorden wat het bovenstaande PSD doet.

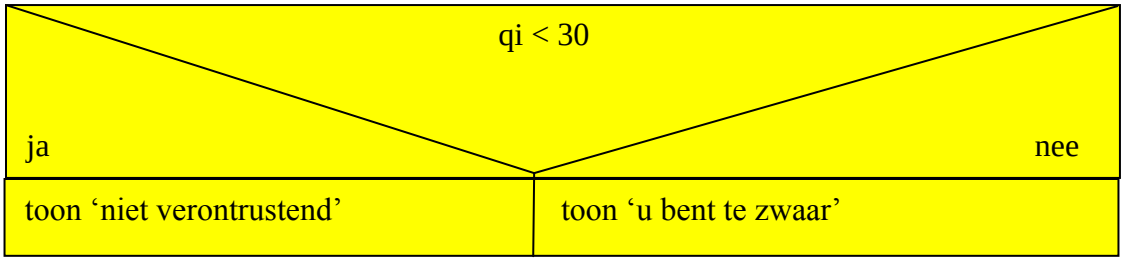
Het PSD van de vorige bladzijde bevat een **keuzeblok** (**selectie**) waarin het programma op basis van de ingevoerde informatie (lengte / gewicht / geslacht) een keuze gemaakt moet worden:



Als het geslacht van de betreffende persoon "V" is dan dient er 6 van de lengte afgehaald te worden en anders niet. Van dit keuzeblok is eenvoudig een (Visual Basic 6) If … Then programma instructie te maken:

```
If geslacht = "V" Then
        lengte = lengte - 6
EndIf
```

In het tweede keuzeblok wordt op basis van een berekende waarde (qi) een keuze gemaakt:



Als de quetelet index van de betreffende persoon kleiner is dan 30 dan dient de tekst "niet verontrustend" te verschijnen en anders de tekst "u bent te zwaar". Ook van dit keuzeblok is eenvoudig een (Visual Basic 6) If … Then … Else programma instructie te maken:

```
If qi < 30 Then
        Textbox.Text = "niet verontrustend"
Else
        Textbox.Text = "u bent te zwaar"
EndIf
```

OPDRACHT

Opdracht 2.2
Wat toont de computer op het scherm bij de volgende invoer door de gebruiker?

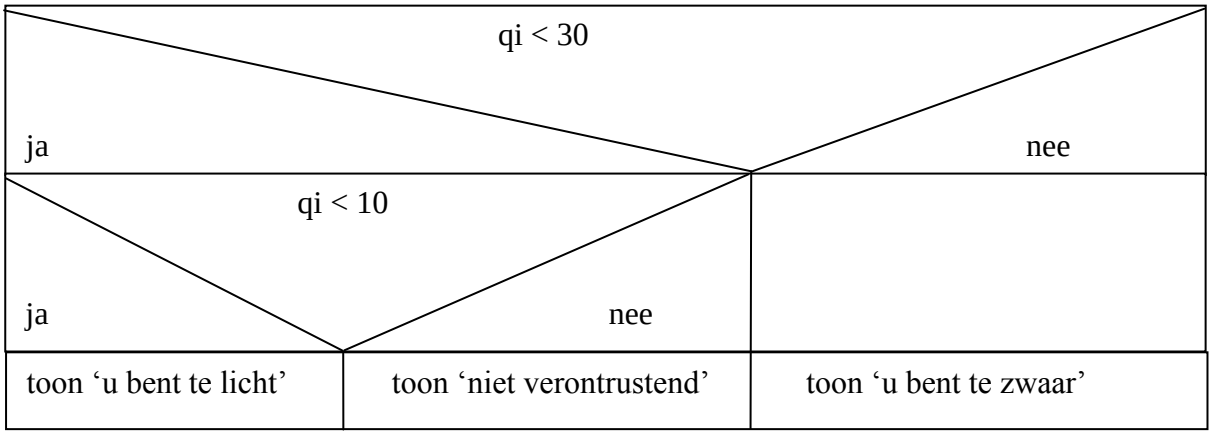
Lengte:	170		Lengte:	180
Gewicht:	80	en bij	Gewicht:	95
Geslacht:	V		Geslacht:	M

2.2 Geneste keuzeblokken

Het Programma Structuur Diagram uit paragraaf 2.1 bepaalt de Quetelet Index (qi).
Deze index geeft aan hoe je gewicht zich tot je lengte verhoudt. Als je gewicht te groot is voor je lengte, overschrijdt de index de waarde 30. In dat geval geeft de index aan dat je te zwaar bent.

Als je te licht bent is dat natuurlijk ook niet goed. Stel dat een qi < 10 betekent dat je te licht bent, hoe dient het PSD dan te worden aangepast?

De eerste regels van het PSD, waarin de qi wordt berekend, blijven natuurlijk hetzelfde.
De laatste regels van het PSD worden:



We noemen een keuzeblok binnen een keuzeblok ook wel **geneste** keuze omdat het ene blok zich heeft genesteld in het andere blok. In (Visual Basic 6) programmacode ziet dat er zo uit:

```
If qi < 30 Then
        If qi < 10 Then
                Textbox.Text = “u bent te licht”
        Else
                Textbox.Text = “niet verontrustend”
        EndIf
Else
        Textbox.Text = “u bent te zwaar”
EndIf
```

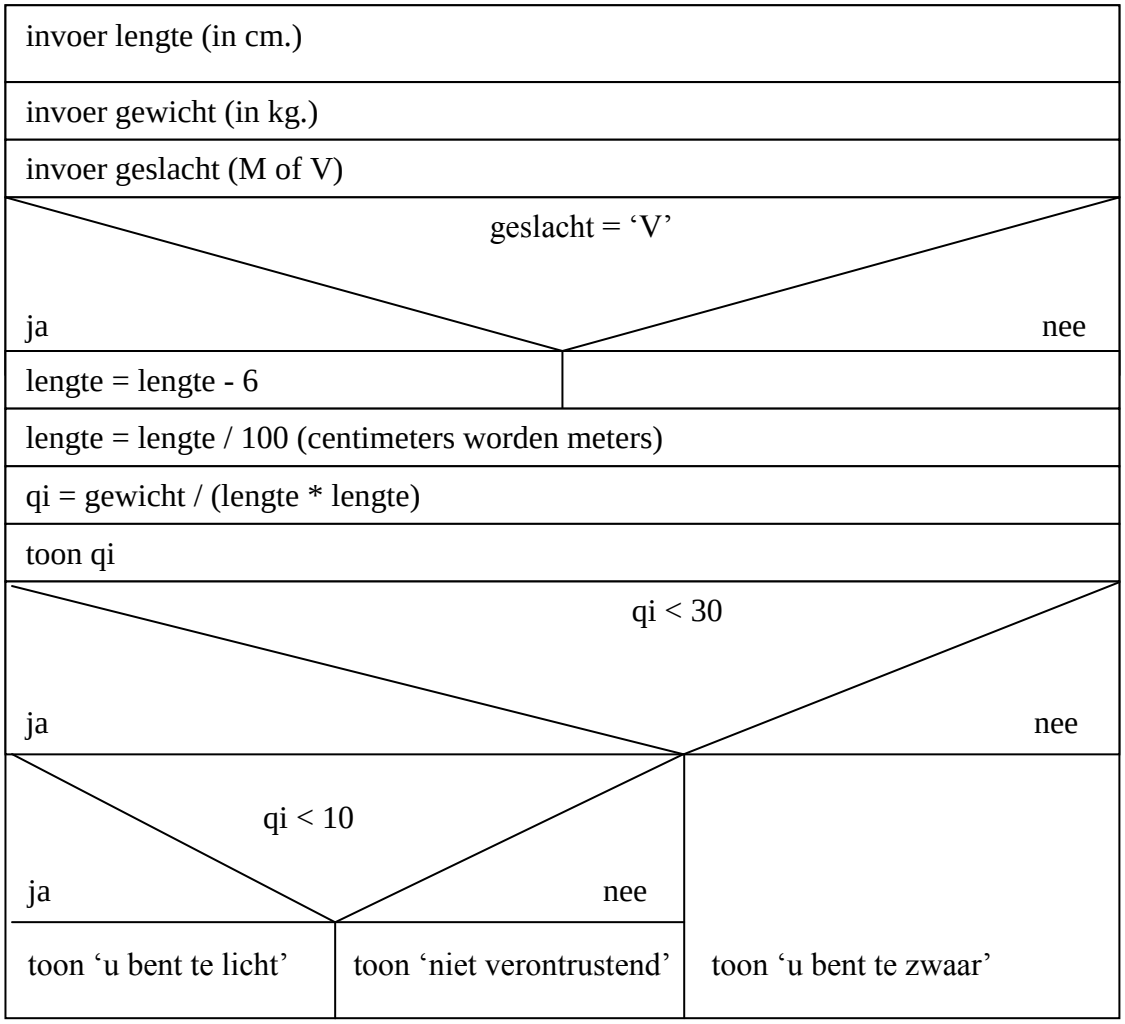
OPDRACHT

Opdracht 2.3
Schrijf een PSD bij het algoritme dat bepaalt of een jaar een schrikkeljaar is (in de Gregoriaanse jaartelling, zie les C-01 paragraaf 1.4).

Gebruik hierbij geneste keuzeblokken.

2.3 Volgorde van instructies

Ook de **volgorde** (**sequentie**) van instructies kan van belang zijn.
We bekijken weer het PSD.



OPDRACHT

In opdracht 2.2 heb je bepaald dat bij onderstaande invoer:

Lengte: 170
Gewicht: 85
Geslacht: V

de uitslag 'u bent te zwaar' getoond wordt.

Opdracht 2.4
Heeft de verwisseling van de instructies:
"Invoer lengte" en "Invoer gewicht" invloed op de uitslag ?

Opdracht 2.5
Heeft de verwisseling van de instructies:
"lengte = lengte / 100" en "qi = gewicht / (lengte * lengte)" invloed op de uitslag ?

2.4 Werken met herhalingsblokken

In sommige algoritmen moeten bepaalde instructies een aantal keren worden herhaald. Dat levert PSD's op zoals hiernaast weergegeven.

teller := 1
uitkomst = teller * 7
schrijf(uitkomst)
teller := teller + 1
uitkomst = teller * 7
schrijf(uitkomst)
teller := teller + 1
uitkomst = teller * 7
schrijf(uitkomst)
teller := teller + 1
uitkomst = teller * 7
schrijf(uitkomst)
teller := teller + 1
uitkomst = teller * 7
schrijf(uitkomst)
teller := teller + 1
uitkomst = teller * 7
schrijf(uitkomst)
teller := teller + 1
uitkomst = teller * 7
schrijf(uitkomst)
teller := teller + 1
uitkomst = teller * 7
schrijf(uitkomst)
teller := teller + 1
uitkomst = teller * 7
schrijf(uitkomst)
teller := teller + 1
uitkomst = teller * 7
schrijf(uitkomst)

OPDRACHT

Opdracht 2.6
Bekijk het PSD hiernaast.
Wat doet dit PSD?

Wanneer herhaald eenzelfde instructie moet worden uitgevoerd is het handig om gebruik te maken van een herhalingsblok. In het PSD hiernaast moet herhaald eenzelfde instructie worden uitgevoerd.

Het PSD (de tafel van 7) kan met behulp van een herhalingsblok veel efficiënter worden weergegeven:

teller = 1	
doe zolang teller < 11	
	uitkomst = teller * 7
	schrijf uitkomst
	teller = teller + 1

In het **herhalingsblok** (**iteratie**) worden de instructies opgenomen die telkens worden herhaald. In de kop van het herhalingsblok wordt aangegeven welk aantal keren de instructies moeten worden herhaald.

Herinner de Zeef van Eratosthenes (zie les C-01 paragraaf 1.3).
Het bijbehorende algoritme luidt:

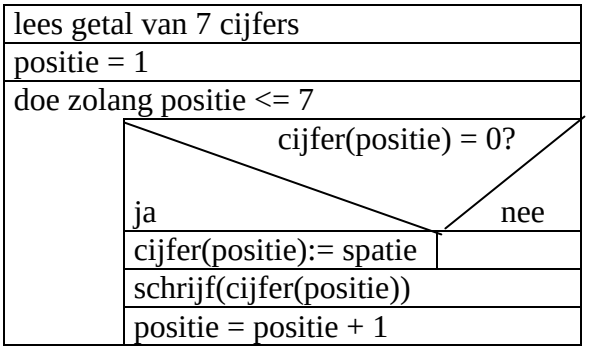
We willen alle priemgetallen kleiner dan n weten.
Schrijf alle getallen 2 tot en met n op.
Start met het kleinste getal (2 dus).
Streep alle veelvouden van dat getal door.
Ga naar het volgende niet doorgestreepte getal (3 dus).
Streep alle veelvouden van dat getal door.
Herhaal dit zolang het getal kleiner is dan $\sqrt{n}$*.*

Het PSD bij dit algoritme luidt:

schrijf rij met getallen 2 tot en met *n*	
getal = 2	
doe zolang getal < $\sqrt{n}$	
	streep veelvouden van getal weg
	getal = volgend getal in de rij

OPDRACHT

Gegeven is het volgende PSD. In dit PSD zit zowel een herhalingsblok en een keuzeblok.



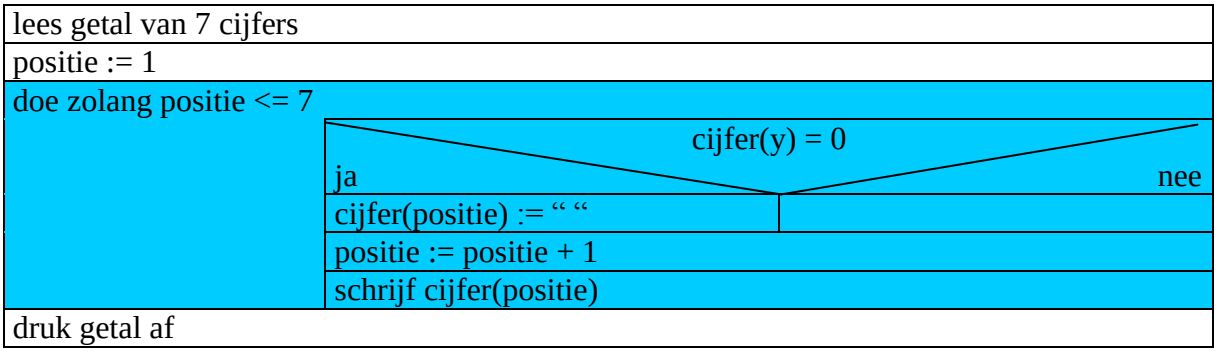
We voeren bijvoorbeeld het getal 5602301 in

Opdracht 2.7
Vul de onderstaande tabel aan:

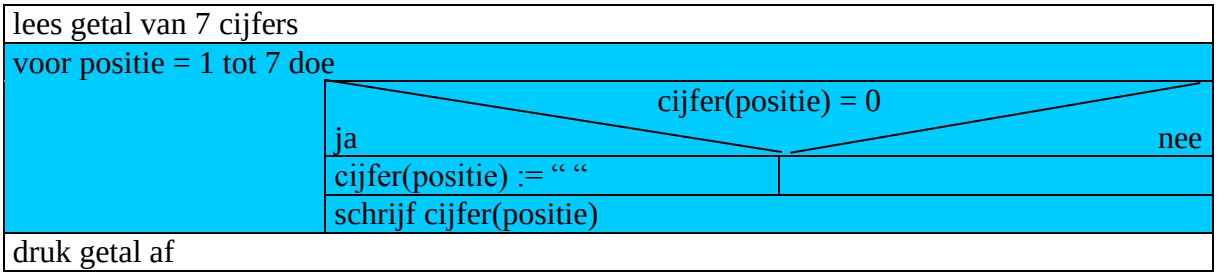
positie	positie <= 7	cijfer(positie)	cijfer(positie)=0?	cijfer(positie)	positie : = positie + 1
1	ja	5	nee	5	2
2	ja	6	nee	6	3
3					

Opdracht 2.8
De uitvoer van dit PSD bestaat uit een aantal schrijfopdrachten.
Wat is de exacte uitvoer van het programma?

Er zijn verschillende Visual Basic codes voor herhalingsblokken denkbaar.
De bovenstaande voorbeelden bevatten een “doe zolang ….. ” instructie (**Do While** …**Loop**).



Een andere mogelijkheid is een “voor ….. doe” instructie (**For ... Next**).



2.5 Zelf PSD's maken.

In een Programma Structuur Diagram (PSD) worden de instructievoorschriften uit een algoritme gestructureerd. Je hebt nu kennis gemaakt met de belangrijkste ingrediënten van men (PSD):

- de instructies moeten in de juiste volgorde staan
- keuzemogelijkheden worden in een keuzeblok weergegeven
- herhalingen worden in een herhalingsblok weergegeven

Hieronder staat een aantal opdrachten waarbij je zelf een PSD moet maken:

OPDRACHTEN

Opdracht 2.9
Schrijf een PSD voor het omzetten van een decimaal getal kleiner dan 256 in een binair getal van 8 bits.

Opdracht 2.10
Schrijf een PSD voor het omzetten van een binair getal van 8 bits in een decimaal getal.

Opdracht 2.11
Schrijf een PSD dat bepaalt of een ingevoerd getal een priemgetal is.

Opdracht 2.12
Schrijf een PSD dat alle priemgetallen kleiner dan 1000 als uitvoer geeft.

2.6 Samenvatting

Een algoritme geeft de instructievoorschriften die moeten worden doorlopen om een bepaald resultaat te bereiken.

Een Programma Structuur Diagram (PSD) geeft structuur aan de voorschriften. We onderkennen hierbij de volgende basisstructuren:

- instructieblokken, waarbij in het algemeen de volgorde (sequentie) van belang is
- keuzeblokken (selectie)
- herhalingsblokken (iteratie)

Met behulp van PSD's kan je computerprogramma's ontwerpen.

ANTWOORDEN

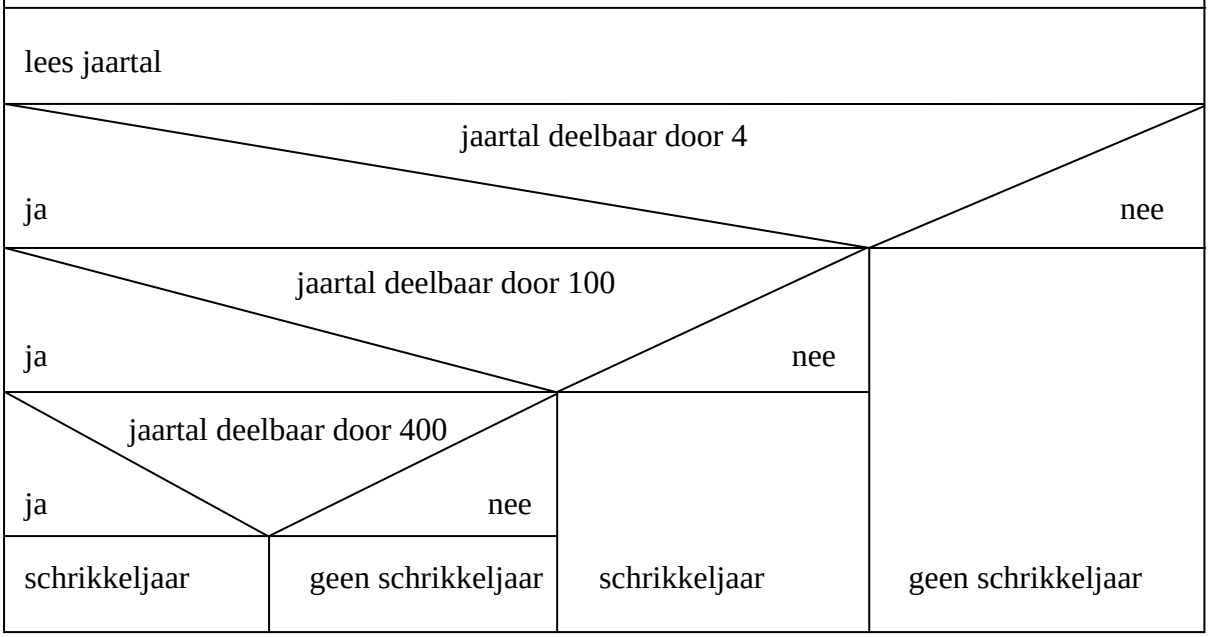
Opdracht 2.1
H et PSD bepaalt, na invoer van lengte, gewicht en geslacht of de betreffende persoon te zwaar is voor zijn/haar lengte of niet.

Dat wordt gedaan aan de hand van de qi (Quetelet index), die de verhouding tussen gewicht en lengte in een getalswaarde uitdrukt. Als deze verhouding te groot is, dan voert het programma uit dat de betreffende persoon te zwaar is.

Opdracht 2.2

Invoer:	Lengte = 170 / Gewicht = 85 / Geslacht = "V"
Verwerking:	Lengte = 164 / Lengte = 1,64 / qi = 31,60
Uitvoer:	'u bent te zwaar'
Invoer:	Lengte = 180 / Gewicht = 95 / Geslacht = "M"
Verwerking:	Lengte = 180 / Lengte = 1,80 / qi = 29,32
Uitvoer:	'niet verontrustend'

Opdracht 2.3



Opdracht 2.4
De verwisseling van de instructies
"Invoer lengte" en "Invoer gewicht" heeft geen invloed op de uitslag !

Opdracht 2.5
De verwisseling van de instructies
"lengte = lengte / 100" en "qi = gewicht / (lengte * lengte)" heeft wel invloed op de uitslag ! Deze wordt nu: 'u bent te licht'

Opdracht 2.6
Het PSD voert de tafel van 7 uit.

Opdracht 2.7

positie	positie <= 7	cijfer(positie)	cijfer(positie)=0?	cijfer(positie)	positie : = positie + 1
1	ja	5	nee	5	2
2	ja	6	nee	6	3
3	ja	0	ja	‘ ‘	4
4	ja	2	nee	2	5
5	ja	3	nee	3	6
6	ja	0	ja	‘ ‘	7
7	ja	1	nee	1	8
8	nee				

Opdracht 2.8
De uitvoer van het PSD is: **56 23 1** (elke ‘0’ wordt vervangen door een spatie)

Opdracht 2.9
Om een decimaal getal in een binair getal (eigenlijk een woord van acht 1-en en 0-en) om te zetten moet worden nagegaan uit welke machten van 2 het decimale getal is opgebouwd. Voorbeeld:
187 = **1** x 128 + **0** x 64 + **1** x 32 + **1** x 16 + **1** x 8 + **0** x 4 + **1** x 2 + **1** x 1 = 10111011
Om te bepalen of je een bepaalde macht van 2 nodig hebt voor het opbouwen van het decimale getal gebruik je het volgende algoritme:

- begin met de grootste macht van 2, 2^7 (N.B. $2^7 = 128$).
- deel het decimale getal door deze macht, 2^7.
- is de uitkomst groter of gelijk aan 1, dan is deze macht van 2 nodig.
- als de macht nodig is, haal dan deze macht van het decimale getal af.
- herhaal de bovenstaande stappen met 2^6 , 2^5 , 2^4 …. , 2^1 .

lees decimaalgetal
teller = 7
binairgetal = ‘’
doe zolang teller >= 0

(decimaalgetal) / (2^{teller}) >= 1	
ja	nee
binairgetal := binairgetal + ‘1’	binairgetal := binairgetal + ‘0’
decimaalgetal = decimaalgetal – 2^{teller}	
teller = teller – 1	

schrijf binairgetal

Opdracht 2.10
Om een binair getal in een binair getal om te zetten moet worden nagegaan uit welke machten van 2 het binaire getal bevat. Voorbeeld:
10111011 = **1** x 128 + **0** x 64 + **1** x 32 + **1** x 16 + **1** x 8 + **0** x 4 + **1** x 2 + **1** x 1 = 187

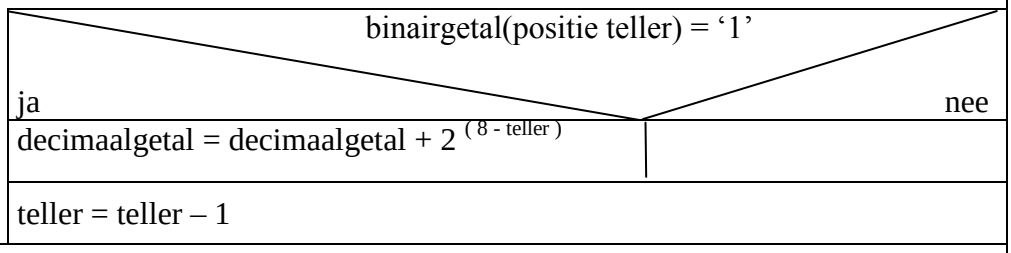
Een algoritme dat een binair getal in een decimaal getal omzet is uitgewerkt in het onderstaande PSD:

lees binairgetal

teller = 1

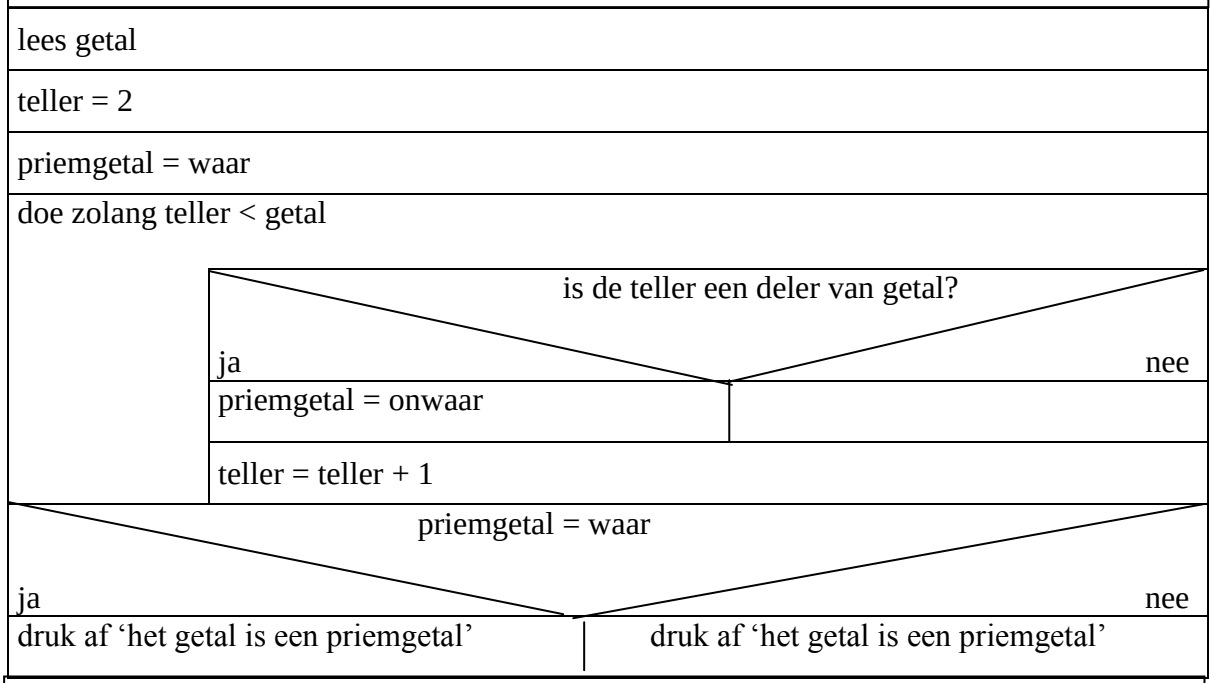
decimaalgetal = 0

doe zolang teller < 9



schrijf decimaalgetal

Opdracht 2.11
Een priemgetal is een getal dat slechts deelbaar is door 1 en zichzelf. Om te bepalen of een getal een priemgetal is moet dus worden nagegaan of er geen andere delers zijn.
Een algoritme dat een binair getal in een decimaal getal omzet is uitgewerkt in het onderstaande PSD. Dit algoritme gaat er van uit dat het ingevoerde getal een priemgetal is totdat er een deler gevonden is.



In dit programma wordt gebruik gemaakt van een *logische variabele*, de variabele priemgetal. Een logische variabele kan twee waarden aannemen: *waar* of *onwaar*.

In programmeertalen worden logische variabelen ook wel *booleans* genoemd, die de waarden *true* en *false* aan kunnen nemen.

Als je later in het jaar een computerprogramma gaat schrijven dat bepaalt of een getal een priemgetal is kom je op een ander probleem dat in dit PSD nog niet is opgelost, namelijk hoe je moet bepalen of de teller een deler is van het getal.

Opdracht 2.12
We kunnen hierbij de uitwerking van verwerkingsvraag 2.12 gebruiken. We voegen hier echter aan toe dat niet voor een ingevoerd getal maar voor alle getallen kleiner dan 1000 nagegaan moet worden of de getallen priemgetallen zijn.