

# **Informatica vakstudie 2: algoritmiëk**

**Auteur(s): vakgroep informatica**

Vakstudie informatica 2 (semester 2)

Vakstudie informatica 2 (AL) (semester 2)

# INHOUDSTAFEL

<b>Inhoudstafel</b>	<b>- 1 -</b>
<b>Inleiding</b>	<b>- 3 -</b>
<b>Studiewijzer</b>	<b>- 4 -</b>
<b>1      Basisbegrippen</b>	<b>- 6 -</b>
1.1    Algoritmen en deelalgoritmen	- 6 -
1.1.1   Voorbeeld 1	- 6 -
1.1.2   Voorbeeld 2	- 6 -
1.1.3   Een algoritme	- 6 -
1.2    De processor	- 7 -
1.3    Het programma	- 7 -
1.4    Abstracte en concrete algoritmen	- 7 -
1.5    Kenmerken van een algoritme	- 8 -
1.6    De controlestructuren	- 8 -
1.6.1   De opeenvolging of sequentie	- 8 -
1.6.2   De keuze of selectie	- 9 -
1.6.3   De herhaling of iteratie	- 9 -
1.6.4   Geneste controlestructuren	- 10 -
<b>2      Een algoritme schrijven</b>	<b>- 11 -</b>
2.1    Van probleem naar oplossing: werkwijze	- 11 -
2.2    Operatoren	- 11 -
2.3    Van probleem naar oplossing: uitwerking	- 12 -
2.3.1   Invoer	- 12 -
2.3.2   Verwerking	- 13 -
2.3.3   Uitvoer	- 14 -
2.4    Instructies	- 14 -
2.5    Van probleem naar oplossing: schematische voorstelling	- 14 -
2.6    Structorizer	- 15 -
<b>3      De opeenvolging of sequentie</b>	<b>- 17 -</b>
3.1    Theorie en basisbegrippen	- 17 -
3.2    Structorizer	- 17 -
3.3    Oefeningen	- 17 -
3.4    De operatoren voor de gehele deling en rest bij deling	- 18 -
3.5    Oefeningen	- 18 -
<b>4      De tweevoudige keuze of selectie</b>	<b>- 19 -</b>
4.1    Definitie	- 19 -
4.2    De vergelijkingsoperatoren	- 20 -
4.3    Structorizer	- 21 -

4.4	Oefeningen op de tweevoudige keuze	- 21 -
4.5	De enkelvoudige keuze	- 21 -
4.6	Samengestelde voorwaarden	- 23 -
4.7	Oefeningen	- 23 -
<b>5</b>	<b>De meervoudige keuze</b>	<b>- 24 -</b>
5.1	Definitie	- 24 -
5.2	Structorizer	- 26 -
5.3	Oefeningen	- 26 -
<b>6</b>	<b>De begrensde herhaling of iteratie</b>	<b>- 27 -</b>
6.1	Definitie	- 27 -
6.2	Structorizer	- 30 -
6.3	Oefeningen	- 30 -
<b>7</b>	<b>Voorwaardelijke herhaling</b>	<b>- 31 -</b>
7.1	Definitie	- 31 -
7.2	Structorizer	- 32 -
7.3	De voorwaardelijke herhaling met afbreekvoorwaarde	- 32 -
7.4	Structorizer	- 33 -
7.5	Oefeningen	- 33 -
<b>8</b>	<b>Extra oefeningen</b>	<b>- 34 -</b>
8.1	Opeenvolging	- 34 -
8.2	Keuze	- 34 -
8.3	De begrensde herhaling	- 35 -
8.4	De voorwaardelijke herhaling	- 36 -
<b>9</b>	<b>Gemengde oefeningen</b>	<b>- 37 -</b>
<b>10</b>	<b>Omgekeerde oefeningen</b>	<b>- 38 -</b>
<b>11</b>	<b>Oneindige lussen</b>	<b>- 46 -</b>
<b>12</b>	<b>Voorbeeldexamen</b>	<b>- 48 -</b>

## Inleiding

Doorheen de verschillende vakstudies informatica zal je geleidelijk aan leren programmeren. In deze cursus wordt de basis gelegd van het programmeren. Je zal nog niets intikken op computer, hier wordt eerst het probleemoplossend denken aangepakt.

In elke programmeertaal wordt gebruik gemaakt van de controlestructuren: de opeenvolging, de keuze en de herhaling. We zullen deze drie controlestructuren grondig uitspitten. Voor elke oefening wordt een analyse gemaakt en een oplossingschema opgesteld. Hier werd gekozen voor Nasi-Schneiderman-diagrammen als voorstellingswijze.

Deze module is heel praktisch en bijna volledig gebaseerd op oefeningen. In de les zal de leerstof geleidelijk aangebracht worden. Er zullen heel wat voorbeelden getoond worden en oefeningen gemaakt worden. Het volgen van de lessen is een must. De volgorde van de leerstof tijdens de lessen is gelijklopend met de cursus.

Op het einde van de cursus staan een aantal herhalingsoefeningen en een voorbeeldexamen.

## STUDIEWIJZER

Programmeren is een vaardigheid. Denk er ook aan: leren “probleemoplossend denken” vergt veel tijd! Tijdig starten met studeren en regelmatig oefenen is een must om te kunnen slagen voor deze module. Er wordt dan ook heel sterk aangeraden alle lessen te volgen en na de les de leerstof bij te houden en de oefeningen opnieuw te maken. Daag jezelf ook uit door varianten op de opgaven te zoeken, extra oefeningen te zoeken op het internet, zelf oefeningen op te stellen...

Het “blokwerk” voor deze module is minimaal. Het van buiten leren van oplossingen heeft totaal geen zin. Vanuit deze optiek zullen de docenten ook nooit zomaar oplossingen van oefeningen geven. Als er problemen zijn met een bepaalde oefening, kan dit steeds besproken worden in de les of tijdens het monitoraat. Daarbij zal je op de juiste weg gezet worden.

Monitoraten zijn vraaggestuurd en worden op vraag van de student ingericht. Het volstaat de docent aan te spreken of een mailtje te sturen. Vragen kunnen ook via mail beantwoord worden, maar let erop dat feedback via persoonlijk contact zinvoller is.

Het examen is schriftelijk en gesloten boek. Teneinde je te kunnen voorbereiden op het examen werd achteraan een voorbeeldexamen toegevoegd.

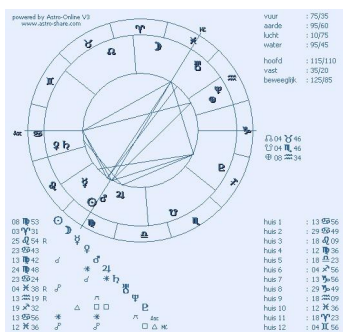
# ALGORITMIEK

# 1 BASISBEGRIPPEN

## 1.1 Algoritmen en deelalgoritmen

### 1.1.1 Voorbeeld 1

Via de site van Skynet kan je een persoonlijke horoscoop laten berekenen. Wil je de volledige uitleg van je sterrenbeeld kennen, dan moet je daar ook voor betalen.



#### Online betalingen - Handleiding

Betalen doe je bij Switchpoint, via je vaste telefoon. Het bedrag wordt automatisch bij je volgende telefoonrekening geteld. Volg nauwkeurig de instructies op die worden getoond op je beeldscherm.

- ✓ Toets of draai dit nummer: 0900-124569
- ✓ Wacht op de pieptoon
- ✓ Geef de code in die op het beeldscherm wordt getoond: 5986
- ✓ Blijf aan de lijn totdat je gegevens zijn doorgezonden en je een biep hoort
- ✓ Je krijgt nu automatisch je volledige horoscoop op je scherm te zien.

### 1.1.2 Voorbeeld 2

Na aankoop van een gsm met Pay&Go-kaart, word je gevraagd onderstaande handleiding te volgen:

- ✓ Laad de batterij van je gsm op
- ✓ Plaats je Pay&Go-Sim-kaart in je gsm
- ✓ Zet je gsm aan
- ✓ Tik je persoonlijke PIN-code van 4 cijfers in
- ✓ Je oproepnummer vind je op de omslag rond de P&G-gebruiksaanwijzing
- ✓ Iemand bellen: vorm het gewenste nummer (vergeet het zonennummer niet)

### 1.1.3 Een algoritme

Wanneer je de *stappen* die in bovenstaande handleiding beschreven zijn, één voor één nauwkeurig uitvoert, zal dat je tot het gewenste resultaat leiden. Samen vormen deze stappen een *algoritme*.

#### Definitie:

Een *algoritme* is een handleiding of een voorschrift dat stap voor stap aangeeft hoe je een bepaald probleem kan oplossen.

Nog voorbeelden:

- de werkwijze om pannenkoeken te bakken
- richtlijnen bij verlies van je bank- of kredietkaart
- richtlijnen voor het betalen aan een parkeerautomaat
- ...

**Definitie:**

De verschillende stappen die je in een algoritme vindt, zijn *deelalgoritmen*. Deze kunnen op hun beurt uit een aantal stappen bestaan (*samengesteld*) of uit één enkele stap (*elementair*).

## 1.2 De processor

Degene die het algoritme uitvoert is de *processor*. Het kan een persoon zijn (vb. jijzelf, je kotgenoot die de pudding maakt) of een machine (vb. een wasautomaat, een DVD-recorder, ...). Het zijn stuk voor stuk programmeerbare machines. Dé programmeerbare machine bij uitstek is de computer. In deze cursus zal de computer altijd de processor zijn.

**Definitie:**

Het algoritme dat uitgevoerd wordt door de processor noemt het *proces*.

## 1.3 Het programma

Dé programmeerbare machine bij uitstek is de computer, een machine die gebruikt wordt voor het oplossen van heel uiteenlopende problemen.

De computer kan echter enkel problemen oplossen als hij daarvoor in zijn geheugen over het gepaste algoritme beschikt. Zonder algoritmen kan de computer niets.

**Definitie:**

De algoritmen waar de computer mee werkt, noemen we *computeralgoritmen* of *programma's*. Daarin staat beschreven wat deze moet doen om een bepaalde taak uit te voeren. Deze zijn geschreven in een taal die de computer begrijpt: *de programmeertaal*.

Een geheel van programma's waarmee de computer kan werken noemt men *software*.

## 1.4 Abstracte en concrete algoritmen

Afhankelijk van de processor die het algoritme moet uitvoeren, zullen de stappen al dan niet zeer gedetailleerd moeten zijn. Kijk bijvoorbeeld nog eens terug naar stap 6 van de handleiding over Pay&Go. "Iemand bellen" (*abstract*) zou je ook kunnen schrijven als:

- herhaal:
    - lees het eerstvolgende cijfer van het te vormen nummer
    - druk met je vinger op de toets die met dit cijfer overeenstemt
  - totdat alle cijfers van het nummer gevormd zijn
- } (*concreet*)

Hoe concreet je een algoritme moet maken, hangt af van wie of wat de processor is. Mensen kunnen bijvoorbeeld problemen oplossen op basis van een abstract algoritme, maar machines kunnen dat meestal niet.

Je moet er altijd voor zorgen dat het algoritme dat je schrijft concreet genoeg is om door de



processor uitgevoerd te kunnen worden. Je zal dus goed moeten weten wat de computer 'begrijpt' om een algoritme voor deze processor te schrijven.

Soms zijn problemen zo ingewikkeld dat je ze niet onmiddellijk in concrete algoritmen kan uiteenrafelen. Je bepaalt dan best eerst de grote lijnen van de oplossing (*grofstructuur*) en werkt deze dan pas verder concreet uit.

**Definitie:**

De methode waarbij een algoritme opgesplitst wordt in een aantal grotere, maar verstaanbare onderdelen (*abstract*) die nadien verder worden verfijnd (*concreet*), noemt met *stapsgewijze verfijning*.

## 1.5 Kenmerken van een algoritme

Algoritmen leiden tot een oplossing van een bepaald probleem. Om dat te kunnen, moeten ze aan een aantal voorwaarden voldoen:

- ✓ de processor moet de stappen kunnen uitvoeren (*uitvoerbaar*)
- ✓ de processor moet de stappen verstaan (*verstaanbaar*)
- ✓ het algoritme en de samengestelde algoritmen hebben een naam (*naam*)
- ✓ het algoritme kan eindigen (*eindigend proces*)

## 1.6 De controlestructuren

Met controlestructuren kan je het al dan niet uitvoeren van instructies van een algoritme of de volgorde van het uitvoeren van de instructies van een algoritme beïnvloeden.

De drie belangrijkste controlestructuren zijn:

- ✓ De opeenvolging of sequentie
- ✓ De keuze of selectie
- ✓ De herhaling of iteratie

In dit hoofdstuk worden deze controlestructuren besproken aan de hand van een voorbeeld. Hierbij is de processor telkens de mens. Later, in de verdere hoofdstukken, zullen deze drie controlestructuren verder uitgediept worden. Daar zal telkens de computer de processor zijn.

### 1.6.1 De opeenvolging of sequentie

Instructies die in opeenvolging staan worden in volgorde van voorkomen (van boven naar onder) juist één keer uitgevoerd. In het onderstaande voorbeeld moeten alle instructies éénmaal uitgevoerd worden, te beginnen bij de eerste instructie en te eindigen bij de laatste instructie, om kokend water te verkrijgen:

#### Een waterkoker gebruiken

- ✓ Vul de waterkoker door de schenktuit
- ✓ Plaats de waterkoker op het droge voetstuk
- ✓ Steek de stekker in het stopcontact
- ✓ Druk de aan/uit schakelaar in.
- ✓ Wacht tot het water kookt
- ✓ Schenk het kokende water uit

### 1.6.2 De keuze of selectie

Zoals je in onderstaand voorbeeld kan zien zal ofwel de instructie “Herstel het ventiel” uitgevoerd worden ofwel zullen de instructies die bij “anders” staan uitgevoerd worden. De instructies bij “Dan” en bij “Anders” staan in een *keuze*: de instructies worden al dan niet uitgevoerd afhankelijk van een voorwaarde (“Is er geen gaatje te vinden”). De eerste twee instructies staan in opeenvolging.

#### Fietsband herstellen

Houd de band in een emmer water

Detecteer het gaatje a.d.h.v. de luchtbelletjes

Als er geen gaatje te vinden is

Dan

Herstel het ventiel

Anders

Schuur met schuurpapier rondom het gaatje

Breng lijm aan op het gaatje

Plak een plakker op het gaatje en druk goed aan

Monteer de band terug

### 1.6.3 De herhaling of iteratie

Het kan ook zijn dat één of meerdere instructies verschillende malen zullen moeten uitgevoerd worden. Deze instructies staan dan in een *herhaling* zoals je kan zien in onderstaand voorbeeld:

#### Olie wagen verversen

Plaats de wagen in horizontale positie

Trek de meetstok uit de opening

Veeg de olie van de meetstok met een schone en droge doek

Vul olie bij in de opening.

Stop de meetstok in de opening

Trek de meetstok uit de opening

Herhaal tot het oliepeil boven de minimuminkeping ligt

In de laatste instructie wordt aan de hand van een voorwaarde nagegaan of de instructies nog eens uitgevoerd moeten worden.

#### 1.6.4 Geneste controlestructuren

Het komt vaak voor dat verschillende controlestructuren door elkaar gebruikt worden, zoals hier in onderstaand voorbeeld:

##### **Online boodschappen doen**

kies een winkel uit de lijst

kies een tijdstip waarop u de boodschappen wil doen

herhaal totdat u alle artikelen heeft

    kies een categorie

    selecteer het gewenste artikel

    als verkeerd artikel wordt geselecteerd

        dan

            verwijder artikel uit lijst door op x te drukken

klik op “aankopen”

geef uw gegevens in

haal uw boodschappen af in het door u gekozen filiaal op het door u gekozen tijdstip.

In dit algoritme komt binnen de *herhaling* een *keuze* voor. De *keuze* is “genest” in de *herhaling*.