



# Olympiades in de Informatica

<http://uclouvain.acm-sc.be/olympiades>

## Voorbeeldvragen voor het secundair onderwijs



Dit document bevat enkele voorbeeldvragen ter illustratie van de wedstrijd voor leerlingen van het secundair onderwijs. Het eerste deel bevat logicavragen en het tweede deel algoritmische vragen. We raden u aan om het document "*Introductie tot de algoritmiek*" te lezen om de oplossingen beter te begrijpen.

## 1 Logica



### 1.1 Zandlopers

Je moet 9 minuten afmeten met behulp van twee zandlopers: een van 7 minuten en een van 4 minuten. Je mag de zandlopers al beginnen manipuleren voordat je de 9 minuten begint te tellen. We vragen je om de oplossing te vinden die in totaal het minste tijd vergt.

Gegeven: 2 zandlopers, waarmee je 4 en 7 minuten kunt afmeten. Laat ons de eerste  $A$  noemen en de tweede  $B$ . We moeten 9 minuten kunnen meten, maar we moeten niet vanaf het begin starten met het tellen van die 9 minuten. We kunnen op voorhand al dingen doen met de zandlopers om ze klaar te zetten. Er is maar 1 ding dat we kunnen doen: *omdraaien*. We kunnen er eentje omdraaien, of de twee tegelijk. De vraag is om de oplossing te vinden die je in totaal het minste tijd kost. Hieronder staat een mogelijke oplossing. Voor elke zandloper noteren we twee getallen  $x/y$ . Het getal  $x$  is de resterende tijd dat de zandloper nog zal lopen,  $y$  is de reeds verstreken tijd.

Handeling	Observatie	 $A$	 $B$	
$A$ en $B$ omdraaien		4/0	7/0	
	$A$ is leeg	0/4	3/4	(4 minuten verstreken)
$A$ omdraaien		4/0	3/4	<b>Begin 9 minuten</b>
	$B$ is leeg	1/3	0/7	(3 minuten verstreken)
$A$ en $B$ omdraaien		3/1	7/0	
	$A$ is leeg	0/4	4/3	(3 minuten verstreken)
$A$ en $B$ omdraaien		4/0	3/4	
	$B$ is leeg	1/3	0/7	(3 minuten verstreken)
				<b>Einde 9 minuten</b>

Deze oplossing vraagt in totaal 13 minuten, maar het is niet de beste. De best mogelijke oplossing vraagt exact 9 minuten. Ze staat hieronder

Handeling	Observatie	 $A$	 $B$	
$A$ en $B$ omdraaien		4/0	7/0	
	$A$ is leeg	0/4	3/4	(4 minuten verstreken)
$A$ omdraaien		4/0	3/4	
	$B$ is leeg	1/3	0/7	(3 minuten verstreken)
$B$ omdraaien		1/3	7/0	
	$A$ is leeg	0/4	6/1	(1 minuut verstreken)
$B$ omdraaien		0/4	1/6	
	$B$ is leeg	0/4	0/7	(1 minuut verstreken)



## 1.2 Wat is de vraag ?

Beeld je in dat er een machine bestaat die correct kan antwoorden op elke ja-of-nee vraag. De machine laat dit zien door twee gekleurde LEDs (groen en rood). Uiteraard is deze machine erg populair. Twee Chinese bedrijven (WEGNAARWAARHEID en PAD-NAARWAARHEID) brengen hun eigen model op de markt.

Elk bedrijf heeft zijn eigen idee over kleurgebruik. Een van de twee bedrijven gebruikt groen = ja en rood = nee; het ander bedrijf doet het tegenovergestelde. Helaas worden de machines geleverd met enkel een Chinese handleiding, die je niet kan begrijpen. De twee machines zien er identiek uit en worden langs hetzelfde kanaal verkocht.

Na enkele weken ontvang je eindelijk een exemplaar van deze machine. Nieuwsgierig als je bent, vraag je je 3 dingen af:

- (a) Welke vraag moet je stellen aan de machine om te weten van welke fabrikant die afkomstig is ?
- (b) Welke vraag moet je stellen om de kleurcode te kennen die de firma WEGNAAR-WAARHEID gebruikt?
- (c) Welke vraag moet je stellen om het groene lichtje te doen branden bij machines van beide fabrikanten ?

## 1.3 Honden en katten

Jan heeft een bepaald aantal katten en honden. Als een van de katten een hond zou worden, dan zou Jan evenveel katten als honden hebben. Als een van de honden een kat zou worden, dan zou Jan twee keer meer katten dan honden hebben. Hoeveel katten en honden heeft Jan?

## 1.4 De rekening

Jan, Mark en Sebastiaan delen een maaltijd. Jan brengt 5 pistolets mee, Mark brengt er 3 mee en Sebastiaan brengt niets mee. De drie verdelen alle voedsel gelijk. Aan het eind van de maaltijd stelt Sebastiaan voor om zijn deel van de maaltijd te betalen en hij legt 8 euro op de tafel voordat hij vertrekt. Jan ziet de 8 muntstukken van 1 euro en beslist om er 5 te nemen, voor de 5 pistolets die hij heeft gedeeld. Er blijft 3 euro over voor Mark die 3 pistolets heeft gedeeld. Mark wordt kwaad omdat hij vindt dat hij recht heeft op meer geld! Wat denk jij? Hoe moeten we de 8 euro van Sebastiaan eerlijk verdelen?

## 1.5 Een oude vraag

Jan was kind gedurende een kwart van zijn leven, jongeman gedurende een vijfde en dan vader gedurende een derde. Nadien heeft hij nog 13 jaar geleefd tot vandaag. Welke leeftijd heeft hij vandaag?



## 2 Algoritmiëk

### 2.1 Het aantal delers

Gegeven een geheel getal  $n > 0$ , schrijf een algoritme dat het aantal positieve gehele delers van  $n$  berekent.

Gegeven: een getal  $n$ , strikt positief en geheel. We moeten het aantal positieve gehele delers van  $n$  berekenen. Eerst moet je de definitie van *dele* herinneren. Een positief geheel getal  $d$  is een dele van  $n$  als je  $n$  exact door  $d$  kan delen, zonder rest. Oftewel: er bestaat een geheel getal  $k > 0$  zodat  $n = k \times d$ . De delers van  $n$  liggen dus automatisch tussen 1 en  $n$ . Als je dit weet, kan je een algoritme schrijven dat voor elk getal  $d$  tussen 1 en  $n$ , of het  $n$  deelt. Een oplossing in het Nederlands kan je bijvoorbeeld zo schrijven:

Zet een teller **cnt** op 0.  
Voor elk geheel getal  $d = 1, 2, \dots, n$  :  
    Als  $d$  deelt  $n$ , dan :  
        Tel 1 op bij **cnt**.  
→ Het getal  $n$  bevat **cnt** positieve gehele delers.

Je kan dit ook gedetailleerder schrijven door middel van pseudo-code zoals hieronder. De operatie  $n \bmod d$  berekent de rest na de gehele deling van  $n$  door  $d$ . Als de rest na de deling gelijk is aan 0, dan is de deling exact, en dan is  $d$  dus een dele van  $n$ .

---

**Algorithm 1:** Het aantal delers

---

**Input:**  $n$ , een strikt positief geheel getal  
**Output:** Het aantal positieve gehele delers van  $n$

```
1  cnt ← 0
2  for d ← 1 to n do
3      if n mod d = 0 then
4          cnt ← cnt + 1
5  return cnt
```

---

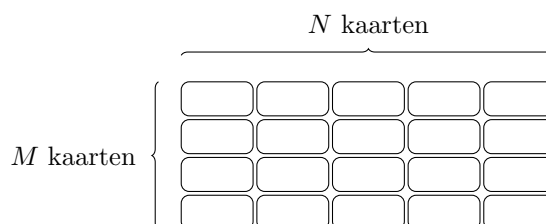
### 2.2 Het kaartentapijt

Alice en Bob beschikken over een heel groot aantal (neem aan: oneindig veel) rechthoekige speelkaarten, allemaal even groot (even lang en even hoog). Ze willen ze naast elkaar leggen in dezelfde richting en zonder overlappings, om een vierkant te maken. Help hen om te vinden hoe groot dat vierkant minimaal zal zijn. Als  $L$  de lengte is van een kaart en  $H$  de hoogte (beide gehele en strikt positieve getallen), moet je een algoritme schrijven dat  $X$  berekent: het aantal benodigde kaarten om het vierkant te vormen.

Eerst en vooral moet je het probleem goed begrijpen. We gaan rechthoekige speelkaarten zij aan zij leggen in dezelfde richting, om een tapijt van kaarten te vormen. Als je de lengte en de hoogte van



een kaart kent, willen we weten hoeveel kaarten we nodig zullen hebben om het kleinst mogelijke vierkant tapijt te vormen. Als we de kaarten horizontaal neerleggen, ziet dat er dus zo uit :



De hoogte van het tapijt is dan  $M \times L$  en de lengte is  $N \times H$  (in de tekening hierboven is  $L$  de korte zijde van een kaart). We willen een vierkant tapijt, zodat  $M \times L = N \times H$ . Wat we nodig hebben is het minimaal benodigde aantal kaarten, dus op het einde willen we  $M \times N$  kennen. Om dit probleem op te lossen, moet je je realiseren dat je hiervoor slechts het kleinste gemeen veelvoud (KGV) nodig hebt van  $L$  en  $H$ . Dan vinden we:

$$M = \frac{P}{H} \quad \text{et} \quad N = \frac{P}{L} \quad \text{met } P = \text{KGV}(L, H)$$

Het probleem komt dus neer op het vinden van het KGV van twee strikt positieve gehele getallen. Een mogelijke manier om dit te doen is om alle veelvouden van  $L$  te doorlopen en te testen of ze ook een veelvoud zijn van  $H$ . Het kleinste houden we over. Voor dit deeltje kunnen we een algoritme schrijven in het Nederlands:

Voor elk geheel getal  $m = 1, 2, \dots, H$  :  
Als  $L \times m$  een veelvoud is van  $H$ , dan :  
→  $L \times m$  is het KGV van  $L$  en  $H$ .

$L \times m$  is veelvoud van  $H$  is hetzelfde als:  $H$  deelt  $L \times m$ . We kunnen dus een algoritme schrijven in pseudo-code met de bewerking mod die we in de vorige vraag hebben gezien. We testen enkel gehele getallen tussen 1 en  $H$ , want we weten dat in het slechtste geval het KGV van  $L$  en  $H$  gelijk is aan  $L \times H$  (dit gebeurt wanneer  $L$  en  $H$  geen gemene delers hebben).

---

**Algorithm 2:** Het kaartentapijt.

---

**Input:**  $L$  en  $H$ , strikt gehele positieve getallen, de lengte en hoogte van een speelkaart**Output:** Het aantal kaarten dat nodig is om een vierkant tapijt te maken van minimale grootte door speelkaarten naast elkaar te leggen in dezelfde richting.

```
1  // Bereken het KGV van L en H
2   $m \leftarrow 1$ 
3   $found \leftarrow \text{false}$ 
4  while not  $found$  and  $m \leq H$  do
5      if  $L \times m \bmod H = 0$  then
6           $found \leftarrow \text{true}$ 
7      else
8           $m \leftarrow m + 1$ 
9  // Bereken aantal benodigde kaarten
10  $kgv \leftarrow L \times m$ 
11  $M \leftarrow kgv/H$ 
12  $N \leftarrow kgv/L$ 
13 return  $M \times N$ 
```

---

## 2.3 Magazijnbeheer

Je bent verantwoordelijk voor een groot magazijn waarin je  $X$  producten kan opslaan. Elke dag komt er een vrachtwagen langs, die  $Y$  nieuwe producten meebrengt en in de plaats daarvan de  $Y$  oudste producten weer weghaalt. De producten die geleverd worden door de vrachtwagen mag je telkens als nieuw beschouwen. De producten in stock en de producten in de vrachtwagen worden voorgesteld door lijsten:  $stock = \langle s_1, s_2, \dots, s_X \rangle$  en  $vrachtwagen = \langle c_1, c_2, \dots, c_Y \rangle$

- Schrijf een algoritme dat de inhoud van de stock en van de vrachtwagen neemt, en dat de staat van de stock aan het eind van de dag teruggeeft.
- pas het vorige algoritme aan zodat het overweg kan met vrachtwagens waarvan de inhoud producten van verschillende leeftijd bevat. Je kan de leeftijd van een product opvragen met de functie  $leeftijd(a)$
- Elke avond doet een werknemer de ronde om te controleren dat de producten niet te oud zijn om nog te kunnen verkopen. Een product dat ouder is dan 126 dagen op het moment van controle, is te oud en moet vernietigd worden. Schrijf een functie die de huidige stocklijst neemt, en een nieuwe stocklijst teruggeeft na controle en eventuele vernietiging van producten.

## 2.4 De lengte van een getal

Schrijf een algoritme dat het aantal cijfers berekent in een strikt positief geheel getal  $n$ .



## 2.5 Het autoklassement

Je bent eindredacteur van een tijdschrift over auto's en je wilt een klassificatie maken van verschillende modellen van auto's. Om dat te doen heb je 5 criteria uitgekozen waarvoor je de waarden hebt voor elk model:

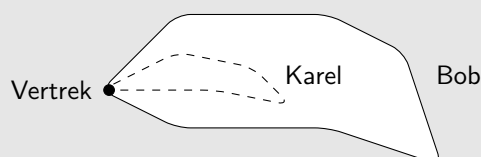
1. lengte ( $L$ ) ;
2. breedte ( $W$ ) ;
3. hoogte ( $H$ ) ;
4. aantal zitplaatsen ( $S$ ) ;
5. aantal airbags ( $A$ ).

Elk automodel wordt dus geïdentificeerd door een lijst van 5 waarden  $\langle L, W, H, S, A \rangle$  die overeenkomen met de 5 criteria. De modellen worden geklasseerd volgens de criteria in de lijst **criteria**. Bijvoorbeeld, als de lijst *criteria* =  $\langle 5, 1, 4 \rangle$ , wilt dat zeggen dat de modellen eerst worden ingedeeld op basis van het aantal airbags (5), in het geval van gelijkheid op basis van lengte (1), en als dat ook niet genoeg is om een onderscheid te maken, op basis van het aantal zitplaatsen (4).

Schrijf een algoritme een lijst met auto's sorteert, gegeven een lijst **criteria**. Maak het zo eenvoudig en zo efficiënt mogelijk.

## 2.6 Blokje lopen?

Bob en Karel beslissen om iets te doen aan hun conditie en gaan hardlopen. Ze hebben elk hun favoriet parcours dat van hetzelfde punt vertrekt en op hetzelfde punt terug aankomt.



Het traject van Bob is  $A$  km lang terwijl dat van Karel  $B$  km lang is. Veronderstel dat  $A$  en  $B$  natuurlijke getallen zijn, niet gelijk aan nul. Schrijf een algoritme dat het aantal toertjes berekent dat Bob en Karel lopen voordat ze elkaar opnieuw kruisen in het vertrekpunt. Je mag aannemen dat ze allebei even snel lopen.

## 2.7 Het spel Nim

We leggen  $n$  lucifers op een tafel. Dit spel wordt met twee gespeeld en om beurt. Elke speler kan 1, 2 of 3 lucifers wegnemen. De verliezer is diegene die het laatst aan de beurt is. Schrijf een algoritme dat beschrijft wat een speler moet doen om altijd te winnen. De speler moet beslissen of hij zelf begint of zijn tegenspeler laat beginnen; daarna moet hij beslissen hoeveel lucifers hij elke beurt wil weghalen.

Je kan de volgende 3 functies gebruiken:

- `setBegin(i)` beslist of je zelf zal beginnen ( $i = 0$ ) of die eer aan je tegenstander laat ( $i = 1$ ).
- `withdraw(i)` zal  $i$  lucifers weghalen (met  $i = 1, 2$  of  $3$ ).
- `getAction()` zegt je hoeveel lucifers er werden weggehaald door je tegenspeler tijdens diens vorige beurt.

## 2.8 Pyramide

We bouwen een pyramide van bollen, en alle bollen die zich op dezelfde hoogte bevinden hebben dezelfde kleur (wit of zwart). Van hoog naar laag hebben de verdiepingen van de piramide dus het volgende aantal bollen:  $1, 4, 9, 16, \dots, i^2, (i+1)^2, \dots$ .

Twee opeenvolgende verdiepingen kunnen niet dezelfde kleur hebben. Witte en zwarte verdiepingen wisselen elkaar dus af. Schrijf een algoritme dat een aantal witte bollen  $n_B$  neemt en een aantal zwarten bollen  $n_N$ , en als resultaat teruggeeft:

- $-1$  als het niet mogelijk is om een pyramide te maken met de nummers  $n_B$  en  $n_N$ ;
- $0$  als de bol op de top van de pyramide wit is;
- $1$  als de bol op de top van de pyramide zwart is.







### 3.2 De faculteit

Vervolledig het onderstaande algoritme. Het neemt een geheel getal  $n > 0$  en berekent er de faculteit van, d.w.z. het product  $1 \times 2 \times \dots \times n$ . Kies een van de volgende instructies:

- (a)  $j \leftarrow j + 1$
- (b)  $j \leftarrow n$
- (c)  $j \leftarrow j + k$

---

**Algorithm 4:** De faculteit

---

**Input:**  $n$ , een strikt positief geheel getal

**Output:** de faculteit van  $n$ , d.w.z. het product  $1 \times 2 \times \dots \times n$

```
1  $i \leftarrow 1$ 
2  $j \leftarrow 0$ 
3  $k \leftarrow n + 1$ 
4 while  $k > 0$  do
5   // hier te vervolledigen
6    $i \leftarrow i * k$ 
7    $k \leftarrow k - 1$ 
8 return  $i/j$ 
```

---