Afbeelding met tekst, illustratie

Automatisch gegenereerde beschrijving

**Leren programmeren met Python : oefeningen**

**Thomas More Kempen**

**Alain Bex**

**September 2023**

Inhoudsopgave

[Inleiding 1](#_Toc145837245)

[H03 Bewerkingen en Output 2](#_Toc145837246)

[H03\_Oefening\_01\_jouw gegevens (\*) 2](#_Toc145837247)

[H03\_Oefening\_02\_jouw gegevens\_etiket (\*) 2](#_Toc145837248)

[H03\_Oefening\_03\_jouw gegevens\_1\_print (\*) 2](#_Toc145837249)

[H03\_Oefening\_04\_tafel\_printen (\*) 3](#_Toc145837250)

[H03\_Oefening\_05\_uitvinding\_schaakspel 3](#_Toc145837251)

[H03\_Oefening\_06\_10\_keer\_vouwen 4](#_Toc145837252)

[H04 Variabelen en input 5](#_Toc145837253)

[H04\_Oefening\_01\_etiket\_maken (\*) 5](#_Toc145837254)

[H04\_Oefening\_02\_stemmen\_tellen (\*) 5](#_Toc145837255)

[H04\_Oefening\_03\_rekenen (\*) 6](#_Toc145837256)

[H04\_Oefening\_04\_uitvinding\_schaakspel\_veld 6](#_Toc145837257)

[H04\_Oefening\_05\_euro2dollar (\*) 7](#_Toc145837258)

[H04\_Oefening\_06\_herhaal\_woord (\*) 7](#_Toc145837259)

[H04\_Oefening\_07\_verkort (\*) 8](#_Toc145837260)

[H04\_Oefening\_08\_alarm 8](#_Toc145837261)

[H04\_Oefening\_09\_temperatuur (\*) 8](#_Toc145837262)

[H04\_Oefening\_10\_naam\_voornaam 9](#_Toc145837263)

[H04\_Oefening\_11\_altijd\_acht 9](#_Toc145837264)

[H04\_Oefening\_12\_verwissel\_strings (\*) 9](#_Toc145837265)

[H04\_Oefening\_13\_print\_1000\_maal 10](#_Toc145837266)

[H04\_Oefening\_14\_verwissel\_getallen 10](#_Toc145837267)

[H05 Eenvoudige functies 11](#_Toc145837268)

[H05\_Oefening\_01\_3\_getallen (\*) 11](#_Toc145837269)

[H05\_Oefening\_02\_IP\_adres 11](#_Toc145837270)

[H05\_Oefening\_03\_oppervlakte\_cirkel (\*) 11](#_Toc145837271)

[H05\_Oefening\_04\_dobbelstenen (\*) 12](#_Toc145837272)

[H05\_Oefening\_05\_volgende\_letter(\*) 12](#_Toc145837273)

[H05\_Oefening\_06\_willekeurige\_letters 13](#_Toc145837274)

[H05\_Oefening\_07\_print\_een\_aantal\_keer 13](#_Toc145837275)

[H06 Condities 14](#_Toc145837276)

[H06\_Oefening\_01\_volwassen\_of\_niet (\*) 14](#_Toc145837277)

[H06\_Oefening\_02\_BMI\_berekenen (\*) 14](#_Toc145837278)

[H06\_Oefening\_03\_blad\_steen\_schaar 15](#_Toc145837279)

[H06\_Oefening\_04\_aantal\_klinkers 16](#_Toc145837280)

[H06\_Oefening\_05\_speelkaart 16](#_Toc145837281)

[H06\_Oefening\_06\_hoger\_lager (\*) 17](#_Toc145837282)

[H06\_Oefening\_07\_geslaagd (\*) 17](#_Toc145837283)

[H06\_Oefening\_08\_langste\_woord(\*) 17](#_Toc145837284)

[H06\_Oefening\_09\_IP\_adres 17](#_Toc145837285)

[H06\_Oefening\_10\_1\_karakter 17](#_Toc145837286)

[H06\_Oefening\_11\_max\_min 18](#_Toc145837287)

[H07 Loops 19](#_Toc145837288)

[H07\_Oefening\_01\_blijven\_vragen\_tot\_0 (\*) 19](#_Toc145837289)

[H07\_Oefening\_02\_blijven\_vragen\_tot\_j (\*) 19](#_Toc145837290)

[H07\_Oefening\_03\_vragen\_tot\_te\_groot (\*) 19](#_Toc145837291)

[H07\_Oefening\_04\_4\_keer\_vragen (\*) 19](#_Toc145837292)

[H07\_Oefening\_05\_n\_keer\_vragen (\*) 19](#_Toc145837293)

[H07\_Oefening\_06\_tafel\_van (\*) 20](#_Toc145837294)

[H07\_Oefening\_07\_grootste\_kleinste (\*) 20](#_Toc145837295)

[H07\_Oefening\_08\_eerste\_integers\_onder\_elkaar (\*) 21](#_Toc145837296)

[H07\_Oefening\_09\_eerste\_integers\_naast\_elkaar (\*) 21](#_Toc145837297)

[H07\_Oefening\_10\_eerste\_integers\_tegen\_elkaar 21](#_Toc145837298)

[H07\_Oefening\_11\_alfabet 21](#_Toc145837299)

[H07\_Oefening\_12\_even\_getallen 22](#_Toc145837300)

[H07\_Oefening\_13\_machten\_van\_2 (\*) 22](#_Toc145837301)

[H07\_Oefening\_14\_IP\_adres 22](#_Toc145837302)

[H07\_Oefening\_15\_5\_reeksjes\_printen (\*) 22](#_Toc145837303)

[H07\_Oefening\_16\_sterrenrechthoek (\*) 23](#_Toc145837304)

[H07\_Oefening\_17\_hoger\_lager\_6\_beurten 23](#_Toc145837305)

[H07\_Oefening\_18\_speelkaarten (\*) 23](#_Toc145837306)

[H07\_Oefening\_19\_genereer\_code 24](#_Toc145837307)

[H07\_Oefening\_20\_sterren\_rijen 24](#_Toc145837308)

[H07\_Oefening\_21\_2\_keer\_3\_letters 24](#_Toc145837309)

[H07\_Oefening\_22\_naar\_binair 25](#_Toc145837310)

[H07\_Oefening\_23\_naar\_hexadecimaal 26](#_Toc145837311)

[H08 Functies 27](#_Toc145837312)

[H08\_Oefening\_01\_Celcius\_2\_Fahrenheit (\*) 27](#_Toc145837313)

[H08\_Oefening\_02\_Euro\_2\_Dollar (\*) 27](#_Toc145837314)

[H08\_Oefening\_03\_druk\_vierkant (\*) 27](#_Toc145837315)

[H08\_Oefening\_04\_dobbelsteen (\*) 28](#_Toc145837316)

[H08\_Oefening\_05\_integer\_naar\_string (\*) 28](#_Toc145837317)

[H09 Recursie 30](#_Toc145837318)

[H10 Strings 31](#_Toc145837319)

[H10\_Oefening\_01\_vervang\_door\_# (\*) 31](#_Toc145837320)

[H10\_Oefening\_02\_vindt\_ze\_allemaal (\*) 31](#_Toc145837321)

[H10\_Oefening\_03\_vervang\_paswoord 32](#_Toc145837322)

[H10\_Oefening\_04\_palindroom (\*) 32](#_Toc145837323)

[H10\_Oefening\_05\_achter\_# 33](#_Toc145837324)

[H10\_Oefening\_06\_IP-adres 33](#_Toc145837325)

[H10\_Oefening\_07\_genereer\_IP-adres 34](#_Toc145837326)

[H10\_Oefening\_08\_woord\_raden 34](#_Toc145837327)

[H10\_Oefening\_09\_geldig\_paswoord 35](#_Toc145837328)

[H10\_Oefening\_10\_genereer\_geldig\_paswoord 35](#_Toc145837329)

[H10\_Oefening\_11\_let\_it\_snow 36](#_Toc145837330)

[H10\_Oefening\_12\_vier\_vieren 37](#_Toc145837331)

[H11 Lists 38](#_Toc145837332)

[H11\_Oefening\_01\_list\_wissel (\*) 38](#_Toc145837333)

[H11\_Oefening\_02\_list\_willekeurig (\*) 38](#_Toc145837334)

[H11\_Oefening\_03\_max\_en\_min (\*) 38](#_Toc145837335)

[H11\_Oefening\_04\_kaartspel (\*\*\*) 39](#_Toc145837336)

[H11\_Oefening\_05\_print\_list (\*) 39](#_Toc145837337)

[H11\_Oefening\_06\_quiz(\*) 40](#_Toc145837338)

[H11\_Oefening\_07\_600\_dobbelstenen (\*) 40](#_Toc145837339)

[H11\_Oefening\_08\_tel\_de\_aantallen 41](#_Toc145837340)

[H11\_Oefening\_09\_zoek\_de\_verschillen 41](#_Toc145837341)

[H11\_Oefening\_10\_100\_verschillende\_getallen (\*) 41](#_Toc145837342)

[H12 Tuples 42](#_Toc145837343)

[H12\_Oefening\_01\_cirkel\_info(\*) 42](#_Toc145837344)

[H12\_Oefening\_02\_grootste\_verschil 42](#_Toc145837345)

[H13 Dictionary 43](#_Toc145837346)

[H13\_Oefening\_01\_dierengeluiden (\*) 43](#_Toc145837347)

[H13\_Oefening\_02\_list\_naar\_dict (\*) 43](#_Toc145837348)

[H13\_Oefening\_03\_dict\_naar\_list 43](#_Toc145837349)

[H14 Sets 44](#_Toc145837350)

[H14\_Oefening\_01\_list\_naar\_set 44](#_Toc145837351)

[H14\_Oefening\_02\_sets 44](#_Toc145837352)

[H15 Os Module 45](#_Toc145837353)

[H15\_Oefening\_01\_grootste\_bestand (\*) 45](#_Toc145837354)

[H16 Tekstbestanden 46](#_Toc145837355)

[H16\_Oefening\_01\_voornamen (\*) 46](#_Toc145837356)

[H16\_Oefening\_02\_voornamen\_omgekeerd (\*) 46](#_Toc145837357)

[H16\_Oefening\_03\_boeken (\*) 46](#_Toc145837358)

[H16\_Oefening\_04\_playlist 46](#_Toc145837359)

[H16\_Oefening\_05\_contacten 47](#_Toc145837360)

[H16\_Oefening\_06\_hoogste\_temperatuur (\*) 48](#_Toc145837361)

[H16\_Oefening\_07\_weerstation 48](#_Toc145837362)

[H16\_Oefening\_08\_voorraad\_merge (\*) 48](#_Toc145837363)

[H16\_Oefening\_09\_woorden 49](#_Toc145837364)

[H16\_Oefening\_10\_studenten\_afstand 50](#_Toc145837365)

[H17 Tijd 51](#_Toc145837366)

[H17\_Oefening\_01\_hoe\_lang (\*) 51](#_Toc145837367)

[H17\_Oefening\_02\_woorden\_lezen\_en\_printen (\*) 51](#_Toc145837368)

[H18 Command line verwerking 52](#_Toc145837369)

[H18\_Oefening\_01\_zoekwoord (\*) 52](#_Toc145837370)

[H19 Exceptions 53](#_Toc145837371)

[H19\_Oefening\_01\_getal\_ingeven (\*) 53](#_Toc145837372)

[H20 Object orientatie 54](#_Toc145837373)

[H20\_Oefening\_01\_cursist\_lijst (\*) 54](#_Toc145837374)

[H20\_Oefening\_02\_cursist\_lijst\_overloading (\*) 54](#_Toc145837375)

[H20\_Oefening\_03\_toverspreuk (\*) 55](#_Toc145837376)

[H20\_Oefening\_04\_hardloper (\*) 55](#_Toc145837377)

[H20\_Oefening\_05\_les\_tijdstip 56](#_Toc145837378)

[H20\_Oefening\_06\_leverancier\_woonplaats (\* 56](#_Toc145837379)

[H20\_Oefening\_07\_drank 57](#_Toc145837380)

[H20\_Oefening\_08\_zuivel 58](#_Toc145837381)

[H20\_Oefening\_09\_kotstudent 59](#_Toc145837382)

[H20\_Oefening\_10\_bij\_de\_dokter 60](#_Toc145837383)

[H21 SQLite en DBMS 61](#_Toc145837384)

[H20\_Oefening\_01\_hotel 61](#_Toc145837385)

[H22 GUI applicaties 63](#_Toc145837386)

[H23 XML 64](#_Toc145837387)

[H22\_Oefening\_01\_planten 64](#_Toc145837388)

[H23\_Oefening\_02\_bioscopen 65](#_Toc145837389)

[H23\_Oefening\_03\_jobkrant 65](#_Toc145837390)

[H23\_Oefening\_04\_songs 66](#_Toc145837391)

[H24 Regex 67](#_Toc145837392)

[H24\_Oefening\_01\_GSM\_nummer (\*) 67](#_Toc145837393)

# Inleiding

* Hoe meer je programmeert hoe beter je het kan!
* Maak minsten de oefeningen met een (\*)
* Gebruik enkel instructies die tot en met het hoofdstuk behandeld zijn.  
  Dit is belangrijk, omdat je leert de bouwstenen te gebruiken die er beschikbaar zijn: sommige programmeertalen hebben maar een beperkte set aan bouwstenen. De cursus leert je voor een stuk programmeren in eender welke taal.
* Als je meer dan 20 minuten vast zit, stop even. Vraag eventueel hulp.  
  Start een nieuwe oefening. Ga later verder.
* Denk steeds aan de gebruiker. Maak het hem zo duidelijk mogelijk.
* Geef goede Nederlandse namen aan variabelen.
* Schrijf commentaar in je programma.
* Bewaar je oplossing onder dezelfde naam als de naam van de oefening; dan vind je ze sneller terug.
* Zet de programma's op een duidelijke afgesproken plaats zodat je ze later nog terugvindt.
* Gebruik de tekst in de opgaves nuttig. Je kan typewerk besparen door te copy/pasten uit de opgave.
* Van iedere oefening krijg je later een mogelijke oplossing. Je moet iedere lijn van die gekregen oplossing kunnen uitleggen.

# H03 Bewerkingen en Output

## H03\_Oefening\_01\_jouw gegevens (\*)

Schrijf een programma dat volgende informatie over jou op het scherm zet:

* Voornaam
* Naam
* Straat
* Huisnummer
* Postcode
* Gemeente

Ieder gegeven komt op een aparte lijn.

Voorbeeld:

Jan

Peeters

Ladeuzeplein

18

4000

Leuven

## H03\_Oefening\_02\_jouw gegevens\_etiket (\*)

Schrijf een programma dat volgende informatie over jou op het scherm zet:

* Voornaam
* Naam
* Straat
* Huisnummer
* Postcode
* Gemeente

Gebruik de oplossing van de vorige oefening.

Pas de code aan zodat er een “etiket” zichtbaar is op het scherm.

Je mag enkel code toevoegen! Niets veranderen, verwijderen.

Bijvoorbeeld:

Jan Peeters

Ladeuzeplein 18

4000 Leuven

## H03\_Oefening\_03\_jouw gegevens\_1\_print (\*)

Schrijf een programma van 1 lijn dat volgende informatie over jou op het scherm zet:

* Voornaam
* Naam
* Straat
* Huisnummer
* Postcode
* Gemeente

Gebruik de oplossing van de vorige oefening.

Pas je programma aan zodat er maar 1 print instructie in voorkomt.

## H03\_Oefening\_04\_tafel\_printen (\*)

Schrijf een programma dat de tafels van vermenigvuldiging van 11 op het scherm zet.

Je laat het programma de berekeningen zelf uitvoeren.

Gebruik zoveel mogelijk kopiëren en plakken.

Voorbeeld:

1 maal 11 = 11

2 maal 11 = 22

3 maal 11 = 33

4 maal 11 = 44

5 maal 11 = 55

6 maal 11 = 66

7 maal 11 = 77

8 maal 11 = 88

9 maal 11 = 99

10 maal 11 = 110

## H03\_Oefening\_05\_uitvinding\_schaakspel

Afbeelding met kruiswoordpuzzel, zwart, binnen, wit

Automatisch gegenereerde beschrijving

De uitvinder van het schaakspel vroeg aan zijn opdrachtgever volgende beloning:

“Majesteit, geef me een graankorrel op het eerste vakje, twee op het tweede vakje, vier op het derde, acht op het vierde, en laat mij zo elk van de vierenzestig vakjes van het schaakbord bedekken.”   
Op het eerste vakje ligt slechts een graankorrel. Het aantal graankorrels op elk volgende vakje is steeds het dubbele van het aantal op het vorige vakje.

Schrijf een programma dat het aantal graankorrels op het 64ste vakje op het scherm zet.

Voorbeeld :

Op het 64-ste veld van het schaakbord liggen 9223372036854775808 graankorrels.

## H03\_Oefening\_06\_10\_keer\_vouwen

Een bankbiljet is 0,011 cm dik.

Laten we dat biljet eens 10 keer vouwen.

Hoe dik is het resultaat.

En 20 keer?

# H04 Variabelen en input

## H04\_Oefening\_01\_etiket\_maken (\*)

Schrijf een programma dat gegevens vraagt van de gebruiker en nadien het adresetiket afdrukt.

Voorbeeld van een uitvoering:

Geef je familienaam en duw <enter> : Jan

Geef je voornaam en duw <enter> : Peeters

Geef je straat en duw <enter> : Kerkstraat

Geef je huisnummer en duw <enter> : 7

Geef je postcode en duw <enter> : 2200

Geef je gemeente en duw <enter> : Herentals

-------- dit is het etiket ----------

Peeters Jan

Kerkstraat 7

2200 Herentals

-------- einde programma ---------

## H04\_Oefening\_02\_stemmen\_tellen (\*)

Schrijf een programma waarmee een gebruiker de resultaten van een stemming kan omvormen tot percentages.

Het programma vraagt het aantal Ja-stemmen, het aantal Nee-stemmen en het aantal blanco stemmen.

Het programma toont het percentage van elk soort stemmen.

Voorbeeld van een uitvoering:

Geef het aantal ja-stemmen en duw <enter> : 125

Geef het aantal neen-stemmen en duw <enter> : 563

Geef het aantal blanco-stemmen en duw <enter> : 10

Resultaat:

----------

% ja-stemmen : 0.17908309455587393

% neen-stemmen : 0.8065902578796562

% blanco-stemmen : 0.014326647564469915

-------- einde programma ---------

## H04\_Oefening\_03\_rekenen (\*)

Het programma vraagt een getal

En drukt dan af :

* De helft van dat getal
* Het dubbel van dat getal
* De derde macht van dat getal
* De rest bij deling door 25 van dat getal

Voorbeeld van een uitvoering:

Geef een getal en duw <enter> : 65

Resultaat:

----------

De helft van 65.0 is 32.5

Het dubbel van 65.0 is 130.0

De derde macht van 65.0 is 274625.0

De rest van de deling van 65.0 door 25 is 15.0

-------- einde programma ---------

## H04\_Oefening\_04\_uitvinding\_schaakspel\_veld

Zie H03\_Oefening\_05\_uitvinding\_schaakspel

Schrijf een programma dat vraagt het hoeveelste veld moet bekeken worden.

Het geeft dan het aantal graankorrels dat op dat veld ligt.

Voorbeeld van een uitvoering:

Geef het nummer van het veld en duw <enter> : 25

Op het veld met nummer 25 liggen 16777216 graankorrels

## H04\_Oefening\_05\_euro2dollar (\*)

Schrijf een programma om een bedrag in Euro om te zetten in Dollar.   
Het programma vraagt de wisselkoers.

Daarna het bedrag in Euro. En geeft dan het resultaat.

Voorbeeld van een uitvoering:

Geef de wisselkoers van Euro naar Dollar en duw <enter> : 1.25

Geef het bedrag in euro dat ik moet omzetten en duw <enter> : 250

Resultaat:

----------

250.0 euro = 312.5 dollar.

## H04\_Oefening\_06\_herhaal\_woord (\*)

Schrijf een programma waarmee je een woord een aantal keer herhaald wordt op het scherm.   
Met een spatie er tussen

Het programma vraagt hoeveel keer het woord herhaald moet worden.

Daarna vraagt het programma het woord.

Daarna zet het programma dat aantal keer het woord achter elkaar op het scherm.

Voorbeeld van een uitvoering:

Geef een aantal en duw <enter> : 5

Geef een tekst en duw <enter> : Python

Resultaat:

----------

5 keer Python = Python Python Python Python Python

## H04\_Oefening\_07\_verkort (\*)

Schrijf een programma waarin berekeningen enkel uitgevoerd worden met verkorte notatie.

De eerste lijn van je programma is :

startgetal = 15

De andere getallen moet je laten berekenen door gebruik te maken van operatoren += -= \*= enz.

Je maakt gebruik van 1 variabele.

Op het scherm komt:

Startgetal : 15

150

155

148

149

14900

745

-------- einde programma ---------

## H04\_Oefening\_08\_alarm

Schrijf een programma dat vraagt:

* Hoe laat het is in uren (een getal tussen 0 en 23)
* Binnen hoeveel uren het alarm moet afgaan.

En dan op het scherm zet om hoe laat het alarm afgaat.

(Tip : gebruik de % operator)

Voorbeeld van een uitvoering:

Geef aan hoe laat het is in uren en duw <enter> : 12

Geef aan binnen hoeveel uur het alarm moet afgaan en duw <enter> : 20

Het alarm gaat af om : 8 uur.

## H04\_Oefening\_09\_temperatuur (\*)

Schrijf een programma waarmee je het aantal graden Celsius kan bereken als je zelf ingeeft hoe hoog de temperatuur is in graden Fahrenheit .

Voorbeeld van een uitvoering:

Geef het aantal graden Fahrenheit en duw <enter> 23

---------------------------------------------

23.0 graden Fahrenheit = -5.0 graden Celsius

-------- einde programma ---------

## H04\_Oefening\_10\_naam\_voornaam

Schrijf een programma dat een naam en een voornaam vraagt.

En dan op het scherm zet wat je in het voorbeeld ziet.

Voorbeeld van een uitvoering:

Geef je voornaam en duw <enter> jan

Geef je naam en duw <enter> peeters

jan peeters

peeters jan

peetersjan

janpeeters

janjanjanjanjanjanjanjanjanjan

peeters janpeeters janpeeters janpeeters janpeeters jan

## H04\_Oefening\_11\_altijd\_acht

Gebruik in dit programma enkel de verkorte notatie om te rekenen.

* Het programma vraagt een getal.
* Trekt 1 af van dat getal.
* Vermenigvuldigt die uitkomst met 3
* Telt 12 op bij de uitkomst.
* Deelt die uitkomst door 3
* Telt 5 op bij die uitkomst.
* Trek van die uitkomst het eerst ingegeven getal af.
* En zet die uitkomst op het scherm.

## H04\_Oefening\_12\_verwissel\_strings (\*)

Schrijf code die de waardes van 2 variabelen verwisseld.

Het programma start bijvoorbeeld met :

woord\_1 = "Test1"

woord\_2 = "Test2"

En je moet nu code schrijven zodat aan het einde van het programma de string 'Test1' in variabele woord\_2 zit en string 'Test2' in variabele woord\_1 zit.

En in die code achter de 2 eerste lijnen mag je nergens quotes gebruiken.

## H04\_Oefening\_13\_print\_1000\_maal

Schrijf een programma dat een string vraagt en dan die string 1000 keer achter elkaar gescheiden door een streepje op het scherm zet.

Achteraan mag geen streepje staan

## H04\_Oefening\_14\_verwissel\_getallen

Schrijf code die de waardes van 2 variabelen verwisseld.

Het programma start bijvoorbeeld met :

getal\_1 = 123

getal\_2 = 567

En je moet nu code schrijven zodat aan het einde van het programma de waarde 123 1' in variabele getal\_2 zit en de waarde 567' in variabele getal\_1 zit.

Je mag geen nieuwe variabele gebruiken.

# H05 Eenvoudige functies

## H05\_Oefening\_01\_3\_getallen (\*)

Schrijf een programma dat 3 getallen vraagt.

En dan op het scherm het volgende zet :

* de som
* het gemiddelde
* het grootste
* het kleinste

Schrijf wat commentaar in het programma.

Voorbeeld van een uitvoering:

Geef het eerste getal en duw <enter> 26.5

Geef het tweede getal en duw <enter> 48.7

Geef het derde getal en duw <enter> 112.3

---------------------------------------------

De som van de 3 getallen = 187.5

Het gemiddelde van de 3 getallen = 62.5

De grootste van de 3 getallen = 112.3

De kleinste van de 3 getallen = 26.5

-------- einde programma ---------

## H05\_Oefening\_02\_IP\_adres

Schrijf een programma dat de gebruiker een IP-adres vraagt en dan de lengte van dat IP-adres op het scherm zet.

Voorbeeld van een uitvoering:

Geef een IP-adres en duw <enter> : 145.12.0.45

Resultaat:

----------

De lengte van 145.12.0.45 = 11

-------- einde programma ---------

## H05\_Oefening\_03\_oppervlakte\_cirkel (\*)

De math module heeft een ingebouwde waarde voor PI.

Gebruik die waarde in een programma dat vraagt naar de straal van een cirkel.

En dan de oppervlakte van de cirkel met die straal op het scherm zet.

De formule voor het berekenen van de oppervlakte :

Oppervlakte = straal \* straal \* Pi.

Voorbeeld van een uitvoering:

Geef de straal van de cirkel en duw <enter> : 5.2

Resultaat:

----------

De oppervlakte van de cirkel = 84.94866535306801

-------- einde programma ---------

## H05\_Oefening\_04\_dobbelstenen (\*)

Schrijf een programma dat 2 dobbelstenen gooit.

En dan het resultaat op het scherm zet.

Laat het verschillende keren lopen en je zal zien dat het telkens een ander resultaat oplevert.

Voorbeeld van een uitvoering:

Resultaat:

----------

Ik gooi een 5 en een 6

-------- einde programma ---------

## H05\_Oefening\_05\_volgende\_letter(\*)

Schrijf een programma dat je vraagt om een letter in te typen.

En dan de letter die er vlak voor komt op het scherm zet.

En daarna de letter die er vlak na komt.

Laat het verschillende keren lopen voor verschillende letters, cijfers, speciale karakters.

Voorbeeld van een uitvoering:

Type een letter en duw <enter> : h

De letter vlak voor h is g

De letter vlak na h is i

## H05\_Oefening\_06\_willekeurige\_letters

Schrijf een programma dat een willekeurige kleine letter gevolgd door een willekeurig grote letter op het scherm zet.

Voorbeeld van een uitvoeringen:

Willekeurige letters: gW

Willekeurige letters: uL

## H05\_Oefening\_07\_print\_een\_aantal\_keer

Schrijf een programma dat een string vraagt en dan een aantal vraagt.

En dan op het scherm die string dat aantal keer achter elkaar herhaalt met een streepje er tussen op het scherm zet. (op het einde dus geen streepje)

Daarna zet het programma de lengte van die geprinte string op het scherm.

Voorbeeld van een uitvoering:

Geef een woord en duw <enter>test

Hoeveel keer moet ik het woord achter elkaar zetten? 100

Resultaat:

----------

test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test-test

# H06 Condities

## H06\_Oefening\_01\_volwassen\_of\_niet (\*)

Schrijf een programma dat het geboortejaar vraagt van de gebruiker.

Het gaat dan beoordelen of de gebruiker volwassen (minstens 18jaar) is of niet.

Voorbeeld van uitvoeringen:

Geef je geboortejaar en duw <enter> : 1990

Je leeftijd = 31

Je bent dus volwassen.

-------- einde programma ---------

Geef je geboortejaar en duw <enter> : 2009

Je leeftijd = 12

Je bent nog niet volwassen.

-------- einde programma ---------

## H06\_Oefening\_02\_BMI\_berekenen (\*)

Iemands BMI (body mass index) wordt als volgt berekend:

BMI = gewicht in kg / (lengte in m \* lengte in m)

De waarde van de BMI is bepalend om een oordeel uit te spreken over het gewicht van een (volwassen) persoon:

|  |  |
| --- | --- |
| BMI < 18 | ondergewicht |
| 18 ≤ BMI < 25 | normaal gewicht |
| 25 ≤ BMI < 27 | licht overgewicht |
| 27 ≤ BMI < 30 | matig overgewicht |
| 30 ≤ BMI < 40 | ernstig overgewicht |
| BMI ≥ 40 | ziekelijk overgewicht |
|  |  |

Maak een programma dat vraagt naar je gewicht in kg en je lengte in cm en op basis daarvan het besluit toont.

Voorbeeld van een uitvoering:

Geef je gewicht in kilogram en duw <enter> : 75

Geef je lengte in centimeter en duw <enter> : 185

-----------------------------------

Een persoon van 75.0 kg met een lengte van 185 cm heeft als BMI 21.913805697589478

Dit is een normaal gewicht.

-------- einde programma ---------

## H06\_Oefening\_03\_blad\_steen\_schaar

**Blad, steen, schaar** is een spel voor 2 spelers.

De spelers kiezen tegelijk één van de mogelijkheden steen, schaar of blad.

Steen verslaat de schaar (de schaar wordt bot), blad verslaat steen (pakt de steen in) en schaar verslaat blad (knipt het stuk).

Als beide spelers dezelfde keuze maken, is het gelijkspel.

Schrijf een programma waar eerst de computer een willekeurige keuze maakt.

En dan aan de gebruiker vraagt om te kiezen tussen één van de drie mogelijkheden *steen*, *schaar* of *blad* .

De computer bepaalt dan wie er wint.

Voorbeeld van uitvoeringen:

Klaar voor een spelletje blad, steen, schaar!

Ik heb iets gekozen!

Wat kies jij?

blad, steen of schaar?

Je verliest ook als je iets verkeerd intypt!

choco

-----------------------------------

Helaas, deze keuze hoort niet bij het spel!

Dus ... .

Je verliest.

Mijn keuze was : schaar

-------- einde programma ---------

Klaar voor een spelletje blad, steen, schaar!

Ik heb iets gekozen!

Wat kies jij?

blad, steen of schaar?

Je verliest ook als je iets verkeerd intypt!

schaar

-----------------------------------

Jij koos schaar

Ik koos blad

Proficiat, je hebt gewonnen!!!

-------- einde programma ---------

## H06\_Oefening\_04\_aantal\_klinkers

De computer vraagt de gebruiker om een string.

Druk af hoeveel verschillende klinkers er in de string zitten.

(als er meerdere keer dezelfde klinker in zit, telt dit voor één.

De hoofdletter versie van een klinker wordt beschouwd als gelijk aan de kleine-letter versie.

3 voorbeelden van een uitvoering:

Geef een woord en duw <enter> banaan

banaan heeft 1 klinker.

Geef een woord en duw <enter> gelei

gelei heeft 2 klinkers.

Geef een woord en duw <enter> typemachine

typemachine heeft 4 klinkers.

## H06\_Oefening\_05\_speelkaart

Een gewoon kaartspel bevat 52 kaarten.

Er zijn 4 soorten kaarten : harten, schoppen, ruiten en klaveren

Er zijn 13 kaarten van iedere soort : 1,2,3, … , 9, 10, boer, dame, heer en aas.

Schrijf een programma dat een willekeurige kaart print.

Een aantal voorbeelden van een uitvoering:

Ik trok uit het kaartspel een ruiten 4

Ik trok uit het kaartspel een schoppen aas

Ik trok uit het kaartspel een harten boer

## H06\_Oefening\_06\_hoger\_lager (\*)

De computer genereert een willekeurige geheel getal tussen 1 en 100.  
Hij vraagt aan de gebruiker een getal.

Als dat getal hoger is dan zijn getal komt er 'Hoger!' op het scherm , is het lager dan komt er 'Lager!' op. Als het hetzelfde is komt er “\*\*\* Juist geraden !!! \*\*\*” op.

## H06\_Oefening\_07\_geslaagd (\*)

Het examen gaat op 20 punten. Als je 10 of meer hebt ben je geslaagd.

De computer vraagt aan de gebruiker een score.

En zet op het scherm een tekst de aangeeft of je geslaagd ben of niet.

## H06\_Oefening\_08\_langste\_woord(\*)

De computer vraagt 2 woorden en zet het langste woord 10 keer achter elkaar op het scherm. Als ze even lang zijn komt er even lang op het scherm.

Een aantal voorbeelden van een uitvoering:

Geef een eerste woord en duw <enter> banaan

Geef een tweede woord en duw <enter> choco

banaanbanaanbanaanbanaanbanaanbanaanbanaanbanaanbanaanbanaan

Geef een eerste woord en duw <enter> gelei

Geef een tweede woord en duw <enter> choco

Even lang

## H06\_Oefening\_09\_IP\_adres

De computer vraagt de gebruiker een IP-adres.

De computer doet dan zoveel mogelijk controles om te zien of dit IP-adres een geldig IP adres is.

In eerste instantie mag je enkel if-instructies gebruiken.

## H06\_Oefening\_10\_1\_karakter

Schrijf een programma dat

aan de gebruiker vraagt om 1 karakter in te tikken en op <enter> te duwen.

Bouw stap voor stap de volgende condities en opdrachten in.

1. het mag maar 1 karakter zijn
2. het mag enkel een cijfer of een kleine letter zijn
3. als het een cijfer is, toon dan het dubbel van dat cijfer
4. als het een kleine letter is, toon dan 100 keer die letter
5. als het een kleine letter is, toon dan de letter juist er voor en de letter juist erna

## H06\_Oefening\_11\_max\_min

Schrijf een programma dat 3 getallen aan de gebruiker vraagt.

En dan aangeeft wat het grootste, het kleinste, het middelste en de som van die 3 getallen op het scherm zet.

**Je mag de max of min functies niet gebruiken,** enkel een goed gebruik van verschillende if blokken, misschien zelfs genest.

# H07 Loops

## H07\_Oefening\_01\_blijven\_vragen\_tot\_0 (\*)

Schrijf een programma dat aan de gebruiker vraagt om een geheel getal in te geven.

De computer blijft een getal vragen tot die gebruiker een 0 ingeeft.

De computer geeft dan het aantal even getallen en de som van die getallen.

Tip : gebruik een if en de % operator

## H07\_Oefening\_02\_blijven\_vragen\_tot\_j (\*)

Schrijf een programma dat aan de gebruiker vraagt om een karakter in te geven.

De computer blijft een karakter vragen tot die gebruiker een cijfer ingeeft.

Als de gebruiker een cijfer ingeeft stopt hij met vragen.

## H07\_Oefening\_03\_vragen\_tot\_te\_groot (\*)

Schrijf een programma dat aan de gebruiker vraagt om een getal in te geven.

De computer blijft een getal vragen tot de som van alle tot dan toe ingegeven getallen gelijk is aan 50.

Dan geeft de computer hoeveel getallen de gebruiker heeft ingegeven.

## H07\_Oefening\_04\_4\_keer\_vragen (\*)

Schrijf een programma dat aan de gebruiker 4 keer vraagt om een getal in te geven.

Dan geeft de computer hoeveel getallen de gebruiker heeft ingegeven en wat de som is van al die ingegeven getallen.  
(Omdat je exact weet hoeveel keer, gebruik je een for)

## H07\_Oefening\_05\_n\_keer\_vragen (\*)

Schrijf een programma dat aan de gebruiker vraagt hoeveel getallen hij wil ingeven.

En dan vraagt de computer exact zoveel getallen aan de gebruiker.

Dan geeft de computer hoeveel getallen de gebruiker heeft ingegeven en wat de som is van al die ingegeven getallen.  
(Omdat je exact weet hoeveel keer, gebruik je een for)

## H07\_Oefening\_06\_tafel\_van (\*)

Schrijf een programma dat aan de gebruiker een getal vraagt tussen 1 en 10.

en dan de tafel van vermenigvuldiging van dat getal op het scherm zet.

Voorbeeld van een uitvoering:

Geef een getal van 1 tot 10 en duw <enter> :7

---------------------------------------------------------

De tafel van: 7

----------------------

1 maal 7 = 0

2 maal 7 = 7

3 maal 7 = 14

4 maal 7 = 21

5 maal 7 = 28

6 maal 7 = 35

7 maal 7 = 42

8 maal 7 = 49

9 maal 7 = 56

10 maal 7 = 63

----------- einde programma -------------

## H07\_Oefening\_07\_grootste\_kleinste (\*)

Schrijf een programma dat aan de gebruiker 4 getallen vraagt tussen 1 en 100.

en dan het grootste en het kleinste van die getallen op het scherm zet.

Voorbeeld van een uitvoering:

Type een getal en duw <enter> :10

Type een getal en duw <enter> :20

Type een getal en duw <enter> :5

Type een getal en duw <enter> :15

---------------------------------------------------------

Resultaat:

----------

Aantal getallen: 4

Grootste : 20.0

Kleinste : 5.0

----------- einde programma -------------

Variant : Dezelfde opgave, maar nu vraagt het programma eerst hoeveel getallen je wil ingeven.

## H07\_Oefening\_08\_eerste\_integers\_onder\_elkaar (\*)

Schrijf een programma dat de eerste 100 gehele getallen onder elkaar op het scherm zet.

## H07\_Oefening\_09\_eerste\_integers\_naast\_elkaar (\*)

Schrijf een programma dat de eerste 100 gehele getallen naast elkaar op het scherm zet. De getallen worden gescheiden door een komma.

## H07\_Oefening\_10\_eerste\_integers\_tegen\_elkaar

Schrijf een programma dat de eerste 100 gehele getallen tegen elkaar (geen spatie ertussen) op het scherm zet.

## H07\_Oefening\_11\_alfabet

Schrijf een programma dat met een for-loop de volgende zaken één voor één doet.

1. Zet de letters van het alfabet onder elkaar op het scherm
2. Zet de letters vlak achter elkaar :

Het alfabet:

------------

Abcdefghijklmnopqrstuvwxyz

1. Zet de letters vlak achter elkaar met een streepje tussen (geen streepje vanachter!):

Het alfabet:

------------

a-b-c-d-e-f-g-h-i-j-k-l-m-n-o-p-q-r-s-t-u-v-w-x-y-z

1. Mooi in rijen van 6 letters. (gebruik modulo = %)

Het alfabet:

------------

a b c d e f

g h i j k l

m n o p q r

s t u v w x

y z

Neem de kleine letters. Gebruik ascii-codes.

## H07\_Oefening\_12\_even\_getallen

Schrijf een programma dat de eerste 100 even getallen op het scherm zet.

Variant H07\_Oefening\_12\_A\_even\_getallen

dezelfde opgave, maar nu op het scherm zetten in rijen van 10. (gebruik modulo = %)

## H07\_Oefening\_13\_machten\_van\_2 (\*)

Schrijf een programma dat de eerste 64 machten van 2 op het scherm zet.

Onder elkaar.

## H07\_Oefening\_14\_IP\_adres

Schrijf een programma dat een IP-adres vraagt.

Het programma controleert dan iedere karakter van de ingegeven string op geldigheid.

(het moet een cijfer of een punt zijn)

En geeft de nodige informatie aan de gebruiker.

Voorbeeld van een uitvoering:

Geef een IP-adres en duw <enter> 125.k3.23.#5

Resultaat:

----------

k is niet geldig in een IP-adres

# is niet geldig in een IP-adres

-------- einde programma ----------

Tip: je kan met de ASCII codes werken maar dat is even opzoeken en minder duidelijk.

En andere veel gebruikte manier is om met een variabele te werken waarin je alle toegelaten karakters steekt.

## H07\_Oefening\_15\_5\_reeksjes\_printen (\*)

Schrijf een programma dat de volgende 5 lijnen print. Met een geneste for-loop.

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

En daarna

5 4 3 2 1

4 3 2 1

3 2 1

2 1

1

## H07\_Oefening\_16\_sterrenrechthoek (\*)

Schrijf een programma dat aan de gebruiker vraagt naar het aantal rijen.

En daarna naar het aantal kolommen.

En daarna een rechthoek met sterren op het scherm zet met dat aantal rijen en kolommen afdrukt.

Voorbeeld van een uitvoering:

Geef het aantal rijen en duw <enter> :2

Geef het aantal kolommen en duw <enter> :5

Resultaat:

----------

\*\*\*\*\*

\*\*\*\*\*

-------- einde programma ----------

Gebruik een geneste for.

## H07\_Oefening\_17\_hoger\_lager\_6\_beurten

De computer genereert een willekeurige geheel getal tussen 1 en 100.  
Hij vraagt aan de gebruiker een getal.

Als dat getal hoger is dan zijn getal komt er 'Hoger!' op het scherm , is het lager dan komt er 'Lager!' op.

De computer blijft vragen naar het getal tot de gebruiker het juiste getal ingeeft.

De computer geeft dan het aantal beurten dat gebruikt werd om het getal te raden.

De gebruiker mag maximaal 6 keer raden.

De computer geeft dan het aantal beurten dat gebruikt werd om het getal te raden.

Of hij zegt dat het getal na zoveel beuren nog niet geraden werd.

Het spel is dan gedaan.

## H07\_Oefening\_18\_speelkaarten (\*)

Een gewoon kaartspel bevat 52 kaarten.

Er zijn 4 soorten kaarten : harten, schoppen, ruiten en klaveren

Er zijn 13 kaarten van iedere soort : 1,2,3, … , 9, 10, boer, dame, heer en aas.

Schrijf een programma dat 13 willekeurige kaarten op het scherm zet.

## H07\_Oefening\_19\_genereer\_code

De computer vraagt je om <enter> te duwen om een code te genereren of 0 in te geven om te stoppen

Als de gebruiker <enter> geduwd heeft toont de computer een willekeurige string van 8 willekeurige kleine letters op het scherm. Zolang je <enter> duwt verschijnt er een nieuwe code.

(gebruik ASCII)

## H07\_Oefening\_20\_sterren\_rijen

Schrijf een programma dat aan de gebruiker een geheel getal vraagt.

Afhankelijk van zijn input komen er een aantal rijen met sterren op het scherm.

Als hij 4 ingeeft komt er :

\*

\* \*

\* \* \*

\* \* \* \*

Geeft hij 5 in komt er :

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

Gebruik een geneste for.

## H07\_Oefening\_21\_2\_keer\_3\_letters

De computer vraagt je om <enter> te duwen.

Als de gebruiker <enter> geduwd heeft toont de computer een willekeurige tekst van 3 kleine letters op het scherm.

Dan vraagt de computer opnieuw om <enter> te duwen.

En opnieuw als de gebruiker <enter> geduwd heeft toont de computer een willekeurige tekst van kleine 3 letters op het scherm.

Hint : gebruik randint en de chr functie

Voorbeeld van een uitvoering:

Duw <enter> :

Eerste code van 3 kleine letters : udz

Duw <enter> :

Tweede code van 3 kleine letters : sev

-------- einde programma ---------

## H07\_Oefening\_22\_naar\_binair

De computer blijft een geheel getal vragen tot je niets invoert en dadelijk <enter> duwt.

Als je dadelijk <enter> duwt stopt het programma.

Als je een getal ingeeft en dat geheel getal niet tussen 0 en 255 ligt, geeft hij een foutmelding.

Anders zet hij dit getal om naar zijn binaire voorstelling.

Voorbeeld van een uitvoering:

Geef een geheel getal tussen 0 en 255 en duw <enter>.

Als je wil stoppen : duw <enter> zonder iets in te geven.

? 17

17 is binair 00010001

Geef een geheel getal tussen 0 en 255 en duw <enter>.

Als je wil stoppen : duw <enter> zonder iets in te geven.

? 54

54 is binair 00110110

Geef een geheel getal tussen 0 en 255 en duw <enter>.

Als je wil stoppen : duw <enter> zonder iets in te geven.

?

-------- einde programma ----------

## H07\_Oefening\_23\_naar\_hexadecimaal

De computer vraagt een geheel getal.

Als dat geheel getal niet tussen 0 en 255 ligt, geeft hij een foutmelding.

Anders zet hij dit getal om naar zijn hexadecimale voorstelling.

# H08 Functies

## H08\_Oefening\_01\_Celcius\_2\_Fahrenheit (\*)

Schrijf een programma dat aan de gebruiker een aantal graden Celcius vraagt.

En dan het overeenkomstig aantal graden Fahrenheit op het scherm zet.

Definieer hiervoor een functie die deze waarde teruggeeft.

De formule is : graden\_fahrenheit = graden\_celcius \* 9 /5 + 32

Mogelijke uitvoering :

Geef een aantal graden Celcius en duw <enter> :25

25.0 graden Celcius is 77.0 graden fahrenheit

-------- einde programma ---------

## H08\_Oefening\_02\_Euro\_2\_Dollar (\*)

Schrijf een programma dat aan de gebruiker een aantal Euro vraagt.

En dan de wisslkoers van die dag vraagt.

En dan het overeenkomstig aantal dollar op het scherm zet.

Definieer hiervoor een functie met 2 parameters.

Mogelijke uitvoering :

Geef bedrag in EURO en duw <enter> :88

Geef de wisselkoers en duw <enter> :1.05

88.0 Euro is 92.4 dollar

-------- einde programma ---------

## H08\_Oefening\_03\_druk\_vierkant (\*)

Schrijf een programma dat aan de gebruiker blijft vragen :

een string en een aantal.

tot hij niets meer ingeeft voor de string.

Telkens hij een string en een aantal ingeeft zal het programma een aantal lijnen drukken gelijk aan het opgegeven aantal.

Iedere lijn bevat het opgegeven aantal keer het opgegeven teken.

Maak hiervoor een functie met 2 parameters.

Mogelijke uitvoering :

Geef een symbool en duw <enter> (niets = STOP) :&

Hoeveel keer moet ik het printen? 3

&&&

&&&

&&&

Geef een symbool en duw <enter> (niets = STOP) :oe

Hoeveel keer moet ik het printen? 2

oeoe

oeoe

Geef een symbool en duw <enter> (niets = STOP) :

-------- einde programma ---------

## H08\_Oefening\_04\_dobbelsteen (\*)

Schrijf een programma met een functie er in.

De functie heeft 1 parameter : het aantal zijden van de dobbelsteen.

De functie heeft als default parameter de waarde 6.

De functie geeft dus het ogen van de worp terug.

Roep de functie een paar keer op, met en zonder parameter.

Mogelijke uitvoering :

Ik werp met een dobbelsteen met 6 zijden een : 4

Ik werp met een dobbelsteen met 24 zijden een : 19

\*\*\* einde programma \*\*\*

## H08\_Oefening\_05\_integer\_naar\_string (\*)

Schrijf een functie die een integer omvormt naar een string met een op te geven aantal posities.

Als de lengte van de integer kleiner is dan het opgegeven aantal posities, voegt de functie aan de linker nullen toe.

Als de lengte van de integer groter is dan het opgegeven aantal posities, geeft de functie de string “Fout” terug.

Dus 1 en 2 geeft 01, 1 en 4 geeft 0001, 12 en 5 geeft 00012, 12 en 1 geeft Fout

Mogelijke uitvoering :

input: 1 en 4 geeft 0001

input: 123 en 6 geeft 000123

input: 123 en 2 geeft FOUT

# 

# H09 Recursie

Oefeningen op recursie worden interessanter als je andere, complexere datastructuren kent. Vandaar dat er in de verdere hoofdstukken op sommige plaatsen recursieve oplossingen gevraagd worden.

# H10 Strings

## H10\_Oefening\_01\_vervang\_door\_# (\*)

Schrijf een programma dat aan de gebruiker een zin blijft vragen en op <enter> duwt tot hij niets meer intypt en dus dadelijk op enter duwt.

Voor iedere ingetypte zin toont het programma op het scherm een patroon waarbij iedere karakter (behalve de spaties) van die zin vervangen is door een #.

Mogelijk uitvoering:

Geef een zin en duw <ENTER> (niets om te stoppen) : choco is lekker

Omgevormde zin: ##### ## ######

Geef een zin en duw <ENTER> (niets om te stoppen) : banaan ook

Omgevormde zin: ###### ###

Geef een zin en duw <ENTER> (niets om te stoppen) :

Einde programma.

Tip: doorloop de string met een for loop en maak terwijl je er door loopt een nieuwe string aan.

## H10\_Oefening\_02\_vindt\_ze\_allemaal (\*)

Schrijf een programma dat alle voorkomen gaat zoeken van een string in een andere string en de posities ervan in die string op het scherm zet.

Je kan je programma laten starten met :

dna = "catgacctatcctcatactttcctaccataactatgttgtcacccacccagggtggtacc\

cgtcggatcgacggcgcctacatgattggtgacgtgtaactggaagcacgactgatcgcg\

atgtcatgcgctgagcgttgagtgtggtcggtgaatatactcgctgttgcggtaggcgct\

cagggcggtggtcgaagaggcagagcggaactatttgtggcggggatacgtctgccatcg\

atgatggaacgaattgttgtcccaattatgcaaggacggcttccaccgatttgaccaggc\

agttgttaacggacacttacagatgatttgttttctgtggtaatgacgcatttcttttta"

Mogelijk uitvoering:

Geef een aantal karakters en duw <ENTER> (niets om te stoppen) : gct

Sequentie gct gevonden op 129

Sequentie gct gevonden op 162

Sequentie gct gevonden op 177

Sequentie gct gevonden op 278

Geef een aantal karakters en duw <ENTER> (niets om te stoppen) : mmm

Sequentie niet gevonden)

Geef een aantal karakters en duw <ENTER> (niets om te stoppen) : tttt

Sequentie tttt gevonden op 330

Sequentie tttt gevonden op 354

Sequentie tttt gevonden op 355

Geef een aantal karakters en duw <ENTER> (niets om te stoppen) :

Einde programma.

Hint : de find method kan ook gebruikt worden met een tweede parameter die aangeeft vanaf welke index je moet zoeken in een string.

Hint : als de find method de string niet vindt geeft ze altijd -1 terug.

## H10\_Oefening\_03\_vervang\_paswoord

Volgende string staat aan het begin van je programma:

te\_verwerken = "Aangelogd met paswoord=abcdef op vrijdag 3 april"

Schrijf een programma dat in deze string het paswoord achter de paswoord= gaat vervangen door \*\*\*\*\*\*.

Het resultaat op het scherm moet dus zijn:

Aangelogd met \*\*\*\*\*\* op vrijdag 3 april

Je mag er van uit gaan dat het paswoord steeds achter paswoord= staat.

Soms zit er geen paswoord= in, dan krijgt de gebruiker een passend foutbericht.

Maak een functie met als parameter een string die een string teruggeeft waarbij het paswoord vervangen is door \*'n.

## H10\_Oefening\_04\_palindroom (\*)

Een palindroom is een woord dat je van voor naar achter en van achter naar voor kan lezen, zonder dat de betekenis wijzigt. Voorbeelden: kok, Bob, lepel, radar, Hannah, redder, koortsmeetsysteemstrook

Schrijf een programma dat een woord vraagt aan de gebruiker en afdrukt of het een palindroom is of niet.

Mogelijk uitvoering:

Geef een woord en duw <ENTER> : lepel

lepel is een palindroom.

Einde programma.

En

Geef een woord en duw <ENTER> : vork

vork is geen palindroom.

Einde programma.

## H10\_Oefening\_05\_achter\_#

Schrijf een programma dat op de eerste lijn een zin in een variabele zet :

Voorbeelden :

zin = "Deze zin is een test-zin waarin een # staat."

zin = "Deze zin is een test-zin waarin geen hashtag in staat."

Als er een # in staat geeft het programma hetgeen achter de # staat.

Anders zegt het dat er geen # in staat.

Mogelijke uitvoering

De zin is: Deze zin is een test-zin waarin geen hashtag in staat.

Er staat geen # in de zin

Einde programma.

Of

De zin is: Deze zin is een test-zin waarin geen # staat.

Het stuk achter de # is: staat.

Einde programma.

## H10\_Oefening\_06\_IP-adres

(Je mag de functie methode count niet gebruiken. Zoek wel even op hoe je dat met die methode kan doen; zo kan je je programma controleren)

Schrijf een programma dat op de eerste lijn een ip-adres in een variabele zet :

Voorbeelden :

#ip\_adres = "263.45.3O.88"

#ip\_adres = "263.45.3.02.88"

#ip\_adres = "123.45.4.88"

#ip\_adres = "123.45.4,88"

#ip\_adres = "123.45.4.288"

ip\_adres = "288.45.4.800"

#ip\_adres = "123.45.4.288.6"

#ip\_adres = "12O.45.4.288"

Door de # weg te doen en terug te zetten kan je gemakkelijk je programma testen.

Het programma doet achtereenvolgens een aantal controles op geldigheid.

* er mogen enkel cijfers of punten in staan
* er moeten exact 3 punten in staan
* de stukken waaruit het adres bestaat moeten integers zijn tussen 0 en 255.

Het programma geeft duidelijk aan wat er mis is.

## H10\_Oefening\_07\_genereer\_IP-adres

Schrijf een programma dat een willekeurig IP-adres genereert.

Maak daar dan een functie van.

Geneer een lijst van 5 willekeurige ip-adressen.

Mogelijke uitvoering

75.57.17.12

159.99.81.9

236.71.43.223

142.234.222.195

65.162.73.165

Einde programma.

## H10\_Oefening\_08\_woord\_raden

Schrijf een programma om een vereenvoudigde versie van Rad Van Fortuin te spellen.

Doel van het spel is om een woord te raden, waarbij je bij iedere beurt een letter van het woord probeert te raden. Als het een goede letter is, zie je dat in het patroon dat op het scherm komt.

Als je een \* ingeeft mag je het woord raden.

Je mag het woord hardcoderen in je programma.

De eerste lijn van het programma zou er kunnen uitzien als :

te\_raden\_woord = "chocopasta"

Een mogelijke uitvoering:

Het te raden woord ziet er uit als : ##########

Raad een letter of druk een \* als je denkt de woord te weten: a

Het te raden woord ziet er uit als : ######a##a

Raad een letter of druk een \* als je denkt de uitdrukking te kennen: k

Het te raden woord ziet er uit als : ######a##a

Raad een letter of druk een \* als je denkt de uitdrukking te kennen: \*

OK. Je denkt dat je het weet, zeg het maar: pasta

Oeps, fout antwoord!

Het juiste antwoord : chocopastaJe start met op het scherm Bij iedere beurt kan je een letter ingeven.

en nog een mogelijke uitvoering:

Het te raden woord ziet er uit als : ##########

Raad een letter of druk een \* als je denkt de woord te weten: a

Het te raden woord ziet er uit als : ######a##a

Raad een letter of druk een \* als je denkt de uitdrukking te kennen: c

Het te raden woord ziet er uit als : c##c##a##a

Raad een letter of druk een \* als je denkt de uitdrukking te kennen: \*

OK. Je denkt dat je het weet, zeg het maar: chocopasta

Correct! Je hebt gewonnen!

## H10\_Oefening\_09\_geldig\_paswoord

Schrijf een programma dat een paswoord vraagt.

Maar dat paswoord moet aan een aantal regels voldoen.

* het moet langer zijn dan 8 karakters
* er mogen geen spaties in staan
* er moet minstens een speciale karakter (&, ?, #, !, %) in staan
* er moet minstens 1 hoofdletter in staan
* er moet minstens 1 cijfer in staan

Het programma stopt als je een geldig paswoord intypt of helemaal niets.

Eventueel maak je eerst een programma zonder een functie.

Pas daarna je programma aan met een functie met 1 parameter die in een boolean teruggeeft of het paswoord geldig is. Je roept die functie op als het nodig is. (het programma doet dus identiek hetzelfde, maar in de code zit een functie)

Hint: gebruik volgende variabelen in je programma:

hoofdletters = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

kleine\_letters = hoofdletters.lower()

symbolen = "&?#!%"

cijfers = "0123456789"

## H10\_Oefening\_10\_genereer\_geldig\_paswoord

Schrijf een programma dat een geheel getal vraagt.

En dan evenveel paswoorden op het scherm zet als de gebruiker opgeeft.

Maar dat paswoord moet aan een aantal regels voldoen.

* het moet langer zijn dan 8 karakters
* er mogen geen spaties in staan
* er moet minstens een speciale karakter (&, ?, #, !, %) in staan
* er moet minstens 1 hoofdletter in staan
* er moet minstens 1 cijfer in staan

Je programma kan starten met :

import random

hoofdletters = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

kleine\_letters = hoofdletters.lower()

symbolen = "&?#!%"

cijfers = "0123456789"

Maak een functie die zo 1 paswoord genereert en roep die dan een aantal keer op

Een mogelijke uitvoering:

Geef het aantal paswoorden en duw <ENTER>) : 10

M&2tbgbmc

V%5jurqvy

L%6yhlcbj

I%4snerrc

G!4yuecnk

K!6ttzgov

J!5jxxmlk

K#3cdnomm

D#1hcrezm

O#6vhrzfg

Einde programma.

## H10\_Oefening\_11\_let\_it\_snow

Schrijf een programma dat langzaam sneeuwvlokjes (\*) laat neerdalen op het scherm.

Importeer de module time zodat je de functie sleep kan gebruiken.

Zoek op hoe sleep werkt.

Een mogelijke uitvoering:

\* \* \*

\* \*

\* \* \*

\* \* \*

\* \*

\* \* \* \*

\*

\* \*

\* \* \*

\*

\* \* \* \*

\* \*

\* \* \*

\* \*

\* \* \*

Extra :

Naarmate de tijd vordert begint het harder en harder te sneeuwen. (meer vlokken)

## H10\_Oefening\_12\_vier\_vieren

Schrijf een programma dat alle mogelijke uitkomsten van bewerkingen met 4 vieren uitprint.

Een voorbeeld van een bewerking : 4\*4+4/4 met als uitkomst 17.0

Je mag +, -, \* en / gebruiken. Maar geen haakjes.

Gebruik hiervoor de functie eval. Zoek op hoe ze werkt.

# H11 Lists

## H11\_Oefening\_01\_list\_wissel (\*)

Schrijf een programma om in een gegeven list van strings het eerste en laatste element te wisselen. Druk de List af voor en na de wissel.

Je mag starten met de list:

soorten\_beleg = ["choco", "boter", "gelei", "salami"]

## H11\_Oefening\_02\_list\_willekeurig (\*)

Schrijf een programma dat een willekeurig woord uit een lijst woorden op het scherm laat zien als de gebruiker op enter duwt.

Als hij STOP ingeeft en op <enter> duwt stopt het programma.

Je mag starten met de list (of maak zelf een leuke list) :

soorten\_beleg = ["choco", "boter", "gelei", "salami"]

Mogelijke uitvoer:

Duw <ENTER> voor een willekeurig beleg.

Type STOP en <ENTER> voor een willekeurig beleg.

Duw <ENTER> of type STOP!

gelei

Duw <ENTER> of type STOP!

salami

Duw <ENTER> of type STOP!STOP

Einde programma.

## H11\_Oefening\_03\_max\_en\_min (\*)

Schrijf een programma dat aan de gebruiker positieve gehele getallen blijft vragen tot hij 0 ingeeft .

Daarna toont het programma het grootste en het kleinste getal samen met het hoeveelste getal dat dit grootste en kleinste getal werd ingegeven.

Hint : zet die getallen in een list en gebruik de min en max functie.

Mogelijke uitvoering:

Geef een geheel getal en duw <enter> (0 = STOP): 8

Geef een geheel getal en duw <enter> (0 = STOP): 2

Geef een geheel getal en duw <enter> (0 = STOP): 6

Geef een geheel getal en duw <enter> (0 = STOP): 0

Het grootste : 8 ingegeven bij beurt 1

Het kleinste : 2 ingegeven bij beurt 2

Extra: Als de gebruiker een getal ingeeft dat hij al ingegeven heeft, krijgt hij een melding en zal het programma een nieuw getal vragen. Het programma moet er dus voor zorgen dat er geen dubbels kunnen ingegeven worden.

Mogelijke uitvoering:

Geef een geheel getal en duw <enter> (0 = STOP): 8

Geef een geheel getal en duw <enter> (0 = STOP): 2

Geef een geheel getal en duw <enter> (0 = STOP): 8

Dit getal had je al ingegeven!

Geef een geheel getal en duw <enter> (0 = STOP): 2

Dit getal had je al ingegeven!

Geef een geheel getal en duw <enter> (0 = STOP): 9

Geef een geheel getal en duw <enter> (0 = STOP): 0

Het grootste : 9 ingegeven bij beurt 3

Het kleinste : 2 ingegeven bij beurt 2

## H11\_Oefening\_04\_kaartspel (\*\*\*)

Schrijf een programma dat de basis kan vormen voor een kaartspel.

Het genereert een willekeurig dek van 52 verschillende kaarten.

Mogelijke kleuren : schoppen, ruiten, klaveren, harten.

Mogelijke waarden: 2 tot en met 10, boer, dame, heer, aas

Het programma gaat dan de 52 kaarten verdelen over 4 spelers, iedereen krijgt 13 kaarten.

Dit resultaat wordt mooi op het scherm gezet.

## H11\_Oefening\_05\_print\_list (\*)

Als je veel met lists werkt dan kan het soms handig zijn om zo'n list op een compacte manier op scherm te brengen.

Als je alle elementen onder elkaar zet, dat staat het scherm snel vol.

Een handige functie zou kunnen zijn : je geeft aan de functie de naam van een list en die functie print alle elementen van de list af gescheiden met een -.

Schrijf een programma met zo'n functie.

Extra:

Voorzie een bijkomende (optionele) parameter die aangeeft hoeveel elementen er maximaal op een regel mogen staan.

## H11\_Oefening\_06\_quiz(\*)

De eerste lijnen van je programma zijn:

quiz = [\

    ["Oostenrijk", "Wenen"],\

    ["Nederland", "Amsterdam"],\

    ["België", "Brussel"]]

vraag = "Geef de hoofdstad van"

Schrijf een programma dat je vragen blijft stellen tot je alle antwoorden allemaal correct hebt ingegeven.

Als je de vraag correct beantwoordt, dan wordt ze niet meer gesteld.

## H11\_Oefening\_07\_600\_dobbelstenen (\*)

Schrijf een programma dat 600 keer met de dobbelsteen gooit (Gebruik de functie uit voorbeeld : H08\_Voorbeeld\_05\_functie\_dobbelsteen) en die de 600 worpen in een lijst zet.

Gebruik deze lijst in je programma om:

* de getallen mooi in rijen van 20 op het scherm te zetten
* om het aantal keer te tellen dat een bepaalde uitslag gegooid is.

Zo zie je dat random redelijk goed werkt.

Het is soms vervelend dat het telkens andere getallen zijn.

Om er voor te zorgen dat het altijd dezelfde "willekeurige" getallen zijn kan men een "seed" vastleggen met behulp van random.seed(<getal>).

Als je een seed vastlegt in het begin van je programma zullen de random getallen steeds dezelfde zijn.

vb.:

random.seed(2)

## H11\_Oefening\_08\_tel\_de\_aantallen

Start je programma met onderstaande lijst.

getallen = [\

0,3,33,3,66,6,3,6,6,3,0,33,10,3,33,0,6,10,0,10,10,6,33,3,3,0,3,0,3,6,\

33,10,0,6,33,10,33,6,0,3,6,3,0,6,10,0,0,10,66,0,6,6,0,6,0,10,6,3,66,6,\

33,6,6,66,10,10,10,3,10,10,6,6,10,10,33,6,10,0,3,6,6,10,10,6,66,66,0,0,33,10,\

3,66,33,6,10,3,33,6,10,3,0,6,66,6,3,3,6,33,6,6,0,6,3,33,0,6,6,3,3]

Schrijf daarna code die aangeeft hoeveel verschillende getallen er in deze lijst zitten.

Je mag enkel gebruiken wat je tot nu toe gezien hebt.

extra : Je programma moet ook afprinten hoeveel keer ieder getal voorkomt in deze lijst.

## H11\_Oefening\_09\_zoek\_de\_verschillen

Start je programma met onderstaande lijsten.

getallen\_1 = [\

0,3,33,3,66,6,3,6,6,3,0,33,10,3,33,0,6,10,0,10,10,6,33,3,3,0,3,0,3,6,\

33,10,0,6,33,10,33,6,0,3,6,3,0,6,10,0,0,10,66,0,6,6,0,6,0,10,6,3,66,6,\

33,6,6,66,10,10,10,3,10,10,6,6,10,10,33,6,10,0,3,6,6,10,10,6,66,66,0,0,33,10,\

3,66,33,6,10,3,33,6,10,3,0,6,66,6,3,3,6,33,6,6,0,6,3,33,0,6,6,3,3]

getallen\_2 = [\

0,3,33,3,66,6,3,6,6,3,0,33,10,3,33,0,6,10,0,10,10,\

6,33,3,3,0,3,0,3,6,33,10,0,6,33,10,33,6,0,3,6,3,\

0,6,10,0,0,10,66,0,6,6,0,6,0,10,6,3,66,6,33,6,6,\

66,10,10,10,3,10,10,6,6,10,10,33,6,10,0,3,6,6,10,10,6,\

66,66,0,0,33,6,10,3,6,66,33,6,10,3,33,6,10,3,0,6,66,6,3,\

3,6,33,6,6,0,6,3,33,0,6,6,3,3]

De tweede lijst is identiek aan de eerste, behalve dat er iemand 2 getallen heeft bijgevoegd ergens.

Welke getallen en waar?

## H11\_Oefening\_10\_100\_verschillende\_getallen (\*)

Schrijf een programma dat een lijst maakt met daarin 100 verschillende gehele getallen gekozen uit de getallen van 1 tot en met 100.

# H12 Tuples

## H12\_Oefening\_01\_cirkel\_info(\*)

Schrijf een programma met een functie die de oppervlakte en de omtrek van een cirkel met een bepaalde straal berekent. De functie moet dus 2 dingen terug geven.

## H12\_Oefening\_02\_grootste\_verschil

Je programma start met de lijnen :

[8, 12, 11, 47, 22, 95, 86, 40, 33, 78, 28, 79, 5, 75, 88, 21, 56, 82, 51, 93, 66, 48, 70, 57, 65, 35, 6, 4, 49, 60, 41, 50, 55, 68, 23, 72, 24, 31, 30, 7, 25, 42, 26, 18, 67, 69, 52, 71, 87, 73, 27, 58, 54, 96, 74, 98, 53, 76, 46, 59, 61, 29, 97, 62, 92, 99, 63,

84, 77, 32, 64, 36, 80, 81, 83, 85, 89, 90, 91, 45, 94, 100, 1, 2, 3, 9, 10, 34, 43, 13, 37, 14, 38, 15, 16, 44, 39, 17, 19, 20]

Schrijft een programma dat 100 willekeurige getallen genereert en in een lijst zet.

Schrijf een functie die uit een lijst de 2 opeenvolgende getallen haalt met het grootste verschil. Een functie die dus 2 getallen teruggeeft.

# H13 Dictionary

In het hoofdstuk over bestanden ga je nog oefeningen tegenkomen waarbij je dictionaries moet gebruiken.

## H13\_Oefening\_01\_dierengeluiden (\*)

Maak een dictionary met dieren (key) en hun bijhorende geluid (value).

De eerste lijn van je programma bevat deze dictionary.

Het programma vraagt nog 1 dier en bijhorend geluid.

Voeg dat toe aan de dictionary.

Dan doorloopt het programma de volledige dictionary :

het programma vraagt een dier en de gebruiker moet zelf het bijhorende geluid intypen.

Op het einde ziet hij hoeveel hij er juist geraden heeft.

## H13\_Oefening\_02\_list\_naar\_dict (\*)

Vertrek van de lijst (copy paste naar je programma) :

getallen = [\

0,3,33,3,66,6,3,6,6,3,0,33,10,3,33,0,6,10,0,10,10,6,33,3,3,0,3,0,3,6,\

33,10,0,6,33,10,33,6,0,3,6,3,0,6,10,0,0,10,66,0,6,6,0,6,0,10,6,3,66,6,\

33,6,6,66,10,10,10,3,10,10,6,6,10,10,33,6,10,0,3,6,6,10,10,6,66,66,0,0,33,10,\

3,66,33,6,10,3,33,6,10,3,0,6,66,6,3,3,6,33,6,6,0,6,3,33,0,6,6,3,3]

Schrijf een programma dat eerst van deze lijst een dictionary maakt waar je snel kan zien hoeveel keer een bepaald getal in de lijst voorkomt.

Daarna blijft het programma getallen vragen (tot je niets meer ingeeft) en geeft het programma het aantal keer dat het getal voorkomt.

## H13\_Oefening\_03\_dict\_naar\_list

Vertrek met de dictionary:

getallen\_aantallen = { 3 : 22 , 33 : 14, 0 : 20, 66 : 8, 10 : 22, 6 : 33}

Schrijf een programma dat een lijst genereert waarin de key's van deze dictionary precies evenveel voorkomen als de corresponderende values van deze dictionary.

De keys moeten willekeurig (random) verdeelt zijn over deze lijst.

(Je hebt hier meerdere instructies voor nodig, dus werk stap voor stap)

# H14 Sets

In het hoofdstuk over bestanden ga je nog oefeningen tegenkomen waarbij je sets moet gebruiken.

## H14\_Oefening\_01\_list\_naar\_set

Vertrek van de lijst (kopieer in je programma) :

test\_naam = [\

0,3,33,3,66,6,3,6,6,3,0,33,10,3,33,0,6,10,0,10,10,6,33,3,3,0,3,0,3,6,\

33,10,0,6,33,10,33,6,0,3,6,3,0,6,10,0,0,10,66,0,6,6,0,6,0,10,6,3,66,6,\

33,6,6,66,10,10,10,3,10,10,6,6,10,10,33,6,10,0,3,6,6,10,10,6,66,66,0,0,33,10,\

3,66,33,6,10,3,33,6,10,3,0,6,66,6,3,3,6,33,6,6,0,6,3,33,0,6,6,3,3]

Maak een functie met een lijst als parameter.

De functie print de verschillende elementen van die lijst.

( Hint: Zoek op hoe je een lijst converteert naar een set)

## H14\_Oefening\_02\_sets

Start met volgende lijsten met allemaal verschillende getallen (kopieer in je programma):

lijst\_1 = [\

51,92,60,18,45,99,16,33,93,91,11,79,43,83,94,28,14,4,80,61,\

6,64,38,46,59,19,48,35,62,68,71,54,63,77,85,30,89,32,39,17,\

36,84,95,44,57,88,66,47,37,7,5,23,12,72,10,27,49,70,3,9]

lijst\_2 = [\

31,76,70,17,48,78,61,81,75,9,94,2,93,34,71,30,25,87,97,85,\

82,51,20,84,77,67,50,88,90,21,6,39,4,35,79,73,46,89,59,28,\

26,47,52,37,29,80,24,7,3,54,32,14,96,44,60,41,91,27,33,83]

Maak een lijst met al de verschillende getallen uit beide lijsten.

Maak een lijst met de getallen die in beide lijsten zitten.

Maak een lijst met de getallen die enkel in de eerste lijst zitten.

# H15 Os Module

## H15\_Oefening\_01\_grootste\_bestand (\*)

Schrijf een programma dat in de Python directory van de cursus de naam van het grootste bestand en de directory waarin dat bestand zit uitprint. Je mag veronderstellen dat de directory structuur slechts 1 laag diep is.

(extra : indien dit niet zo is, kan je gebruik maken van een recursieve oplossing)

Schrijf een python programma dat binnen een gegeven directory de bestanden uitprint met dezelfde naam en dezelfde grootte.

# H16 Tekstbestanden

## H16\_Oefening\_01\_voornamen (\*)

Los deze oefening een keer op zonder gebruik te maken van het data-type list.

en een keer met gebruik te maken van het data-type list

Je krijgt een bestand H16\_Data\_voornamen.txt

Schrijf een programma dat :

* iedere lijn van het bestand op het scherm zet
* daarna het aantal lijnen op het scherm zet
* daarna alle voornamen met een z afdrukt
* daarna het aantal voornamen met een z afdrukt

## H16\_Oefening\_02\_voornamen\_omgekeerd (\*)

Je krijgt een bestand H16\_Data\_voornamen.txt

Hierin staat op iedere lijn een voornaam.

Lees de volledige file in en plaats alle voornamen in een **List**.

Druk deze list af in omgekeerde volgorde.

## H16\_Oefening\_03\_boeken (\*)

Je krijgt een bestand H16\_Data\_*boeken.txt*.

De oneven lijnen bevatten de titels van boeken, de even lijnen de bijhorende auteurgegevens. Je mag ervanuit gaan dat de file een even aantal lijnen bevat.

Schrijf een programma dat de inhoud van het bestand als volgt afdrukt:

Probeer met list en zonder list

Afbeelding met tekst

Automatisch gegenereerde beschrijving

## H16\_Oefening\_04\_playlist

Je krijgt een bestand H16\_Data\_*playlist.txt.*

Schrijf een programma dat de playlist op deze manier weergeeft. Let op de details!

Afbeelding met tekst

Automatisch gegenereerde beschrijving

## H16\_Oefening\_05\_contacten

Je beschikt over een alfabetisch gesorteerd bestand H16\_Data\_*contactpersonen.csv* met per contactpersoon de volgende informatie: familienaam ; voornaam ; mailadres ; geslacht

Schrijf een programma om de volgende lijst te creëren. Merk op: de lijst is alfabetisch op voornaam

5 meisjes:

Annick Mannaerts

Chris Raes

Greet Peeters

Julie Jacobs

Riet Tyskens

43 jongens:

Bart Meyvis

Ben Lambaerts

Bert Lievens

Brian Onbelet

Dave Michilsen

Diederik Eskens

…

Thomas Eskens

Tim Janssens

Tom Haesen

Yannick De Berti

## H16\_Oefening\_06\_hoogste\_temperatuur (\*)

Voor deze oefening maak je gebruik van de file H16\_Data\_*weerstation\_2018 08.csv.*

Schrijf een programma dat de hoogste temperatuur afdrukt die in die periode gehaald werd.

De hoogste temperatuur in deze periode = 35.92

## H16\_Oefening\_07\_weerstation

In deze oefening maak je een uitbreiding van oefening 6 door ook gebruik te maken van de file H16\_Data\_*weerstation\_2018 10.csv*.

Schrijf een functie waarmee de gebruiker de maand kan doorgeven waarvan hij de weergegevens wil bekijken. De gebruiker antwoordt een getal van 1 tem 12. Zolang hij geen correct getal ingeeft wordt de vraag opnieuw gesteld.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

In het hoofdprogramma wordt het bijhorende bestand geopend, als dit tenminste bestaat.

Het programma drukt dan de volgende gegevens af. Je hebt alleen voor maanden 8 en 10 gegevens ter beschikking.

Afbeelding met tekst

Automatisch gegenereerde beschrijvingAfbeelding met tekst

Automatisch gegenereerde beschrijving

## H16\_Oefening\_08\_voorraad\_merge (\*)

Een firma heeft twee vestigingsplaatsen en op beide plaatsen is er een voorraadmagazijn (H16\_Data\_*voorraad1.txt* en H16\_Data\_*voorraad2.txt*). Men wenst per artikel de totale voorraad te kennen dus wordt er beslist om de twee bestanden samen te voegen.

Schrijf een programma om beide bestanden samen te voegen tot 1 voorraadbestand, waarbij je er mag op rekenen dat beide bestanden gesorteerd zijn op artikelnummer.

|  |  |  |
| --- | --- | --- |
| **voorraad1.txt**  artikelnummer;voorraad 101-66601;12 101-76100;45 101-76200;19 101-76500:33 101-76700;15 101-76900;8 170-33100;54 | **voorraad2.txt**  artikelnummer;voorraad 101-61105;100 101-66601;84 101-76100;5 101-76200;13 101-76400;20 101-76900;32 170-01200;17 170-33100;6 170-37500;2 | **voorraad\_samen.txt**  artikelnummer;voorraad 101-61105;100 101-66601;96 101-76100;50 101-76200;32 101-76400;20 101-76500:33 101-76700;15 101-76900;40 170-01200;17 170-33100;60 170-37500;2 |

## H16\_Oefening\_09\_woorden

Je beschikt over het bestand H16\_Data\_woordenboek.txt.

Schrijf verschillende programmas die dit bestand gaan gebruiken.

Geef je oplossingen namen die overeenkomen met de deelopgave.

Bijvoorbeeld : de oplossing van variant A komt in H16\_Oefening\_09\_A\_woorden.

1. Hoeveel woorden zitten in dit bestand
2. Hoeveel karakters zitten in dit bestand (tel de \n niet mee!)
3. Welke verschillende karakters zitten er; maak een lijst en print die lijst.  
   Doe hetzelfde, maar open het bestand nu met encoding=UTF8 en zie het verschil!  
   (Als je een bestand opent met wordpad, dan kan je de encoding van dat bestand zien)
4. In de oplossing van opgave C zie je dat er in het bestand nogal wat speciale karakters zitten.  
   Speciale karakter = geen letter  
   Laat die woorden met speciale karakter zien op het scherm.   
   Hoeveel woorden zijn er zo?  
   Doe hetzelfde, maar open het bestand nu met encoding=UTF8 en zie het verschil!
5. In oplossing van de vorige opgaves zie je karakters : 'à', 'á', 'ä', 'ç', 'è', 'é', 'ê', 'ë', 'í', 'î', 'ï', 'ñ', 'ó', 'ô', 'ö', 'ú', 'û', 'ü', '₂'. In welke woorden komt '₂' voor? Hoeveel zijn er zo.
6. Hoeveel woorden bevatten een cijfer?
7. Welke palindromen zitten in de lijst? Hoeveel zijn er dat?
8. Afbeelding met badkamer, toilet, kleurrijk, badkuip

   Automatisch gegenereerde beschrijvingIn het spel SCRABBLE mogen woorden gebruikt worden met de karakters uit de lijst 'à', 'á', 'ä', 'ç', 'è', 'é', 'ê', 'ë', 'í', 'î', 'ï', 'ñ', 'ó', 'ô', 'ö', 'ú', 'û', 'ü' als de speciale karakters worden vervangen door respectievelijk : 'a', ‘a', 'a', 'c', 'e', 'e', 'e', 'e', 'i', 'i', 'i', 'n', 'o', 'o', 'o', 'u', 'u', 'u'.  
   Scrabble kent maar één soort letters, dus mag je alle kleine letters door hoofdletters vervangen.  
   Woorden met leestekens mogen niet.  
   Maak een nieuw bestand : H16\_Data\_Out\_Scrabble dat de geldige scrabble woorden bevat op basis van ons woordenboekbestand.  
   Scrabble-woorden bevat geen cijfers.
9. Welke verschillende karakters worden er gebruikt in de woorden en hoeveel van ieder?
10. Zijn er woorden waarin alle letters van jouw voornaam (naam) zitten

## H16\_Oefening\_10\_studenten\_afstand

Schrijf een programma waarmee je van een groep studenten de naam en de afstand tot de hogeschool vraagt.

Deze informatie bewaar je in twee afzonderlijke lijsten.

Na het vragen komt op het scherm:

* een overzicht : student en afstand
* de gemiddelde afstand
* de student die het verste woont met zijn afstand

Als je wil stoppen met informatie ingeven dan typ je "stop" als naam.

Merk op: je mag dan niet meer vragen hoe ver die persoon van de school woont!

Als je meteen stop intikt bij de start van het programma, dan mag je het overzicht onderaan niet afdrukken.

Als je meteen stop intikt bij de start van het programma, dan mag je het overzicht onderaan niet afdrukken.

Alle informatie die je hebt ingevoerd schrijf je weg naar een bestand.

De volgende keer dat je het programma opstart, haal je alle informatie op uit het bestand en zet je het in de 2 lijsten. Daarna kan je terug namen en afstanden invoeren die dan toegevoegd worden aan de lijst.

Opnieuw wordt de lijst opgeslagen op het einde.

Maak deze oefening eerst gebruik makende van de 2 lijsten.

(versie A).

Daarna gebruik makend van dictionaries.

# H17 Tijd

## H17\_Oefening\_01\_hoe\_lang (\*)

Schrijf een programma dat je vraagt om <enter> te duwen waarna het programma een willekeurig aantal seconden tussen 5 en 15 niets doet en je daarna vraagt om te raden hoeveel seconden er verstreken zijn. Het programma geeft dan aan hoeveel percent je antwoord afweek van het echte aantal seconden.

Mogelijke uitvoer:

## H17\_Oefening\_02\_woorden\_lezen\_en\_printen (\*)

Schrijf een programma de eerste 1000 lijnen van het woordenboekbestand H16\_Data\_woordenboek.txt inleest in een lijst.

En dan die woorden één voor één onder elkaar op het scherm zet.

Hoe lang duren beide operaties? (1. Het lezen 2. Het printen)

# H18 Command line verwerking

## H18\_Oefening\_01\_zoekwoord (\*)

Schrijf een programma zodat je aan de command line kan ingeven :

1. Bijvoorbeeld : python zoekwoord choco  
   Dan moet er op het scherm komen of het woord kanarie in het woordenboekbestand van hoofdstuk 16 staat.  
   (gebruik om te testen een testbestand met minder woorden).  
   Zorg voor help en verbose.
2. Bijvoorbeeld : python zoekwoord ?ho??.  
   Dan moet er op het scherm komen : alle woorden uit het woordenboekbestand die voldoen aan het zoekpatroon ?ho??. (Het ? staat voor eender welke karakter)  
   (gebruik om te testen en testbestand met minder woorden).  
   Zorg voor help en verbose.

# H19 Exceptions

## H19\_Oefening\_01\_getal\_ingeven (\*)

Schrijf een programma dat aan de gebruiker iets blijft vragen tot hij uiteindelijk een getal ingeeft. (met een while)

Als hij een getal ingegeven heeft print het programma het dubbel van dat getal.

# H20 Object orientatie

## H20\_Oefening\_01\_cursist\_lijst (\*)

Schrijf een Python-programma dat gebruik maakt van OOP met het 'Cursist'-object om de gegevens van verschillende cursisten te beheren.

Maak een nieuw 'Cursist'-object met de naam 'cursist1'. Het object moet tenminste een 'naam', 'leeftijd' en 'cursus' attribuut hebben.

Druk de informatie van 'cursist1' af met behulp van de methode 'print\_informatie' van de 'Cursist'-klasse.

Maak een ander 'Cursist'-object met een zelfgekozen naam en tenminste een 'leeftijd' en 'cursus' attribuut.

Druk de informatie van het nieuwe 'Cursist'-object af met behulp van de methode 'print\_informatie' van de 'Cursist'-klasse.

Maak een lijst van 'Cursist'-objecten die 'cursist1' en het nieuwe object bevat.

Gebruik een 'for'-loop om door de lijst van 'Cursist'-objecten te itereren en de informatie van elk object af te drukken met behulp van de methode 'print\_informatie' van de 'Cursist'-klasse.

## H20\_Oefening\_02\_cursist\_lijst\_overloading (\*)

Als je oefening 02 gemaakt hebt kan je daar van vertrekken.

Schrijf een Python-programma dat gebruik maakt van OOP met het 'Cursist'-object om de gegevens van verschillende cursisten te beheren.

Maak een nieuw 'Cursist'-object met de naam 'cursist1'. Het object moet tenminste een 'naam', 'leeftijd' en 'cursus' attribuut hebben.

Gebruik nu overloading om de informatie te printen:

Definieer de method \_\_str\_\_ om te definieren wat er moet gebeuren als een Cursist moet ge-"print"

Genereer een lijst van 5 willekeurige cursisten :

* Naam = Cursist + een willekeurig nummer
* Cursus : een willekeurige cursus uit de volgende lijst  
  cursussen = ["Web Development", "Data Science", "AI", "Cloud Computing", "Mobile Development"]
* leeftijd : een willekeurige integer tussen 18 en 42 jaar

Gebruik een 'for'-loop om door de lijst van 'Cursist'-objecten te itereren en de informatie van elk object te "print"en.

Schrijf nu een methode waardoor de < operator overloaded wordt voor cursisten.

Je legt in die methode vast dat een cursist1 < cursist2 als dat geldt voor hun leeftijden.

De method heeft de naam \_\_lt\_\_ (less than)

Gebruik nu de functie sorted om de lijst cursisten te sorteren en print de gesorteerde lijst af.

## H20\_Oefening\_03\_toverspreuk (\*)

Maak een klasse **Toverspreuk.** Een toverspreuk wordt gekenmerkt door een tekst en een aantal.

De methode

* *echo* verdubbelt de tekst en verhoogt aantal met 1  
  (de tekst “gras” wordt dus “grasgras”)
* *\_\_str\_\_* geeft de tekst en het aantal terug als volgt:

De tekst = paspaspas (opgebouwd in 3 keer)

* *vulaan(woord, locatie):* woord wordt bij de bestaande tekst geplakt en aantal wordt met 1 verhoogd. De locatie is ‘a’, ‘v’ of ‘b’ wat betekent dat de tekst achteraan, vooraan of aan beide kanten wordt bijgeplakt.

Maak een Toverspreuk-object met een zelfgekozen tekst.

* + geef aantal de waarde 1
  + echo
  + vul achteraan aan met “hoe”
  + vul vooraan aan met “water”
  + toon het resultaat

## H20\_Oefening\_04\_hardloper (\*)

Maak een klasse **Hardloper** die wordt gekenmerkt door zijn naam, leeftijd en beste\_prestatie.

beste\_prestatie geeft in seconden de best gehaalde tijd op de 100m.

De methode

* *bereken\_snelheid* berekent de snelheid in km/u op de volgende manier:

360 / (best behaalde prestatie in seconden)

* *bepaal\_initialen* geeft de initialen van de persoon terug (eerste letter van de voornaam en eerste letter van de familienaam). Bijvoorbeeld voor een persoon met familienaam “Van Dorpe” en voornaam “Riet” krijg je als returnwaarde RV.   
  Je mag veronderstellen dat de voornaam eerst staat in de naam en geen blanco’s bevat.
* *\_\_str\_\_* geeft de naam, leeftijd en de beste prestatie terug als volgt:

Usain Bolt(UB), 32 jaar en beste prestatie op de 100m = 9.58

Maak een Hardloper-object en experimenteer met de verschillende methodes.

## H20\_Oefening\_05\_les\_tijdstip

Een lessenrooster van een student bestaat uit een aantal objecten van het type Les.

Voorbeelden:

Java maandag 5 P108Y

Engels woensdag 3 G112

Maak een klasse **Tijdstip** die gekenmerkt wordt door een dag en uur en een klasse **Les** gekenmerkt door een vaknaam, lokaal en tijdstip(object van Tijdstip).

Voorzie dat in de klasse **Tijdstip**

* De constructor de dag op ‘maandag’ zet en het uur op ‘1’.
* De methode *toon\_tijdstip()* de volgende string teruggeeft:

bv donderdag 1e lesblok

Voorzie dat in de klasse **Les**

* De methode *toon\_les()*de volgende string teruggeeft (maak gebruik van de methode *toon\_tijdstip()* van Tijdstip):

bv donderdag 3e lesblok HTML5 in lokaal G204

Maak

* een object van de klasse Les met:
  + vaknaam = HTML5
  + lokaal = G204
  + tijdstip = donderdag 3e lesblok
* Toon alle gegevens van de les door gebruikt te maken van de methode toon\_les():



* Toon de volgende gegevens van de les

Afbeelding met tekst

Automatisch gegenereerde beschrijving

## H20\_Oefening\_06\_leverancier\_woonplaats (\*

Maak een klasse **Woonplaats** met een straat+nummer, postcode en gemeente en een klasse **Leverancier** met een naam, contactpersoon en een woonplaats(object van Woonplaats).

Voorzie in de klasse **Woonplaats**

* een constructor die de woonplaats instelt op volgend adres:
  + straat\_nr = "geen gegevens"
  + postcode = 0
  + gemeente = "geen gegevens"

Voorzie in de klasse **Leverancier**

* De methode *toon\_leverancier()* van Leverancier die de volgende tekst teruggeeft:
  + - *Leverancier “U&Me”, met als contactpersoon Jef Perks, is gevestigd in Berkendreef 24 bus 7, 2000 Antwerpen*

NB: deze tekst bestaat uit 1 lijn!

Maak nu

* een object van Leverancier met volgende gegevens:
  + naam = U&Me
  + contactpersoon = Jef Perks
  + woonplaats = Berkendreef 24 bus 7

2000 Antwerpen

* Toon alle gegevens van de leverancier door gebruik te maken van de methode toon\_leverancier()



## H20\_Oefening\_07\_drank

Maak volgende klassen aan:

* Drank gekenmerkt door naam, prijs, inhoud(het aantal ml dat de verpakking bevat) en percentage alcohol
* Klant gekenmerkt door klantnr en naam
* Aankoop gekenmerkt door aantal\_stuks, drank en klant(object van Klant)

Voorzie in de klasse **Drank**

* + een methode *is\_sterke\_drank:* deze methode geeft als returnwaarde *True* wanneer het alcoholpercentage in de drank minstens 35 is. Anders is de returnwaarde *False.*
  + een methode *get\_prijs\_liter* die de prijs per liter teruggeeft. Bijvoorbeeld: je hebt een fles alcoholische drank in een fles van 500 ml die €10 kost. De methode zal dan als antwoord geven dat deze drank € 20 kost per liter.

Voorzie in de klasse **Klant**

* + een methode *get\_geheime\_code*: deze methode geeft als returnwaarde een tekst die als volgt wordt samengesteld:  
    neem de eerste 2 letters van de naam + het klantnummer + de laatste 2 letters van de naam en zet dit alles in hoofdletters.   
    Bijvoorbeeld bij klant *Mestdagh* met nummer *175* is de returnwaarde *ME175GH*

Voorzie in de klasse **Aankoop**

* + een methode *toon\_aankoop()* die de volgende tekst geeft

Klant Peter Vandergoten heeft 3 fles(sen) whiskey (11.5%) aangekocht

Maak een Aankoop-object *mijn\_aankoop* met volgende gegevens:

klant: Peter Vergoten met klantnr 123

drank: Famous Grouse whiskey, prijs:14.69, inhoud:1000,

alcoholpercentage:38.5

aantalStuks: 5

Toon alle gegevens van het *mijn\_aankoop-object* door gebruik te maken van zijn methode *toon\_aankoop*()



Toon vervolgens de volgende gegevens. De tekst “een sterke drank” wordt getoond wanneer de methode *isSterkeDrank()* true teruggeeft.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Wanneer de methode *is\_sterke\_drank()* False teruggeeft, wordt de tekst “geen sterke drank” getoond. Zie onderstaand voorbeeld.

Maak nu een 2de object van de klasse Aankoop met volgende gegevens:

Dezelfde klant: Peter Vergoten met klantnr 123

drank: Cara pils, prijs:0.5, inhoud:250,

alcoholpercentage:4.5

aantalStuks: 50

Toon ook van dit object alle gegevens zoals in vorig voorbeeld:

## H20\_Oefening\_08\_zuivel

Maak volgende klassen aan:

* **Artikel** gekenmerkt door een nummer, naam en eenheidsprijs
* **Zuivel**, een subklasse van Artikel, gekenmerkt door aantalDagen (= aantal dagen dat zuivelproduct nog vers is) en een leveranciercode

Voorzie in de klasse **Artikel**

* + een methode \_\_str\_\_ die het artikel in de volgende vorm teruggeeft:

bv voor een object met nummer 770, naam ‘yoghurt’ en prijs 1.5

770 yoghurt: € 1.50

Voorzie in de klasse **Zuivel**

* + een methode *verlaag\_vers*: deze methode verlaagt het aantal dagen dat een product nog vers is met
  + een methode *snel\_verkoop(min\_dagen)*: deze methode halveert de prijs als blijkt dat het aantal dagen dat het product nog vers is <= min\_dagen

Maak een Zuivel-object met

nummer 770, naam ‘yoghurt’, prijs 1.5, aantalDagen = 5, levCode =’A’.

Pas op dit object de verlaag\_vers-methode toe en de snel\_verkoop-methode met als parameter bijvoorbeeld 5.

Test de juiste werking van snel\_verkoop en verlaag\_vers met bijvoorbeeld dit als resultaat:

Afbeelding met tekst

Automatisch gegenereerde beschrijving

## H20\_Oefening\_09\_kotstudent

Maak volgende klassen aan:

* **Student** gekenmerkt door een nummer, naam, opleiding en adres
* **Kotstudent**, een subklasse van Student, gekenmerkt door kotbaas en kotadres

Voorzie in de klasse **Student**

* + een methode *toon\_student* die de student in de volgende vorm teruggeeft:

bv 5056478 Inne Weemaes van de opleiding Toegepaste Informatica

adres: Veldstraat 4 – 2200 Herentals

Voorzie in de klasse **Kotstudent**

* + een methode *toon\_kotstudent* die de kotstudent in de volgende vorm teruggeeft:

bv 5051963 Johan Lievens(Kotstudent) van de opleiding Toegepaste Informatica

adres: Lange Nieuwstraat 9 – 2150 Borsbeek

kotadres: Velveken 33 – 2440 Geel(eigenaar: Luc Geuens)

Maak een Student-object Inne Weemaes (zoals in de notitie hierboven) en een Kotstudent-object Johan Lievens (eveneens zoals in de notitie hierboven).

Vervolgens laat je beide objecten zien mbv de methodes. Bijvoorbeeld:

Afbeelding met tekst

Automatisch gegenereerde beschrijving

## H20\_Oefening\_10\_bij\_de\_dokter

Maak volgende klassen aan:

* **Persoon** gekenmerkt door een naam
* **Dokter**, een subklasse van Persoon, gekenmerkt door een specialisme en ereloon
* **Patient**, een subklasse van Persoon, gekenmerkt door een sisnummer
* **Raadpleging** gekenmerkt door een dokter(object van Dokter) en een patiënt (object van Patient)

Voorzie in de klasse **Persoon**

* + een methode \_\_str\_\_ die de persoon in de volgende vorm teruggeeft:

bv Naam: Miet Adriaens

Voorzie in de klasse **Dokter**

* + een methode \_\_str\_\_ die de dokter in de volgende vorm teruggeeft:

bv Dr. Myriam Dierick, chirurg

Voorzie in de klasse **Patient**

* + een methode \_\_str\_\_ die de patiënt in de volgende vorm teruggeeft:

bv Naam: Peter Boonen Nummer SIS-kaart: 850412 204M14

Voorzie in de klasse **Raadpleging**

* + een methode \_\_str\_\_ die de raadpleging in de volgende vorm teruggeeft:

bv Dr. Jan Vandereyken, endocrinoloog Naam: Peter Boonen Nummer SIS-kaart: 850412 204M14

Maak een object van de klasse Raadpleging met volgende eigenschappen:

Dokter = Dr. Theo Lebon, sportdokter

Patiënt = Tom Boonen met nr. Sis-kaart: 750510 105M16

Toon vervolgens het gecreëerd object van de klasse Raadpleging op het scherm.

# H21 SQLite en DBMS

## H20\_Oefening\_01\_hotel

Maak volgende klassen aan (plaats deze in verschillende .py files)

1. een klasse **Hotel** met volgende attributen:

* naam bv La Torretta
* aantal\_sterren : bv 3
* skigebied: bv Tonale-Ademallo

Zorg ervoor dat een hotel op volgende manier wordt afgedrukt:

Hotel 'La Torretta' 3 sterren, gelegen in Tonale-Adamello

1. een klasse **Aanbod** met volgende attributen:

* periode bv Kerstvakantie
* hotel
* prijs

Maak sqlite\_hotel.py aan waar je een connectie maakt met sqlite en zorg ervoor dat er 2 tabellen worden aangemaakt:

**CREATE TABLE hotel  
 (naam text,  
 aantal\_sterren int,  
 skigebied tekst)**

**CREATE TABLE aanbod  
 (periode text,  
 hotel text,  
 prijs float)**

Maak vervolgens functies aan om:

* een rij toe te voegen in de tabel hotel
* een rij toe te voegen in de tabel aanbod
* alle rijen van de tabel hotel te tonen gesorteerd op naam
* alle rijen van de tabel aanbod te tonen gesorteerd op prijs
* om alle hotels te zoeken met een bepaald aantal sterren
* om alle hotels te zoeken die liggen in een bepaald skigebied
* om alle hotels te zoeken die beginnen met een bepaalde letter
* om alle vakanties (= tabel aanbod) te tonen die binnen een bepaalde prijscategorie liggen
* om alle vakanties te tonen in een bepaalde periode

Maak nu een aantal objecten aan van de klasse Hotel.

Bv

**La Torretta,** 3 sterren, skigebied: **Tonale-Adamello**  
**San Carlo**, 3 sterren, skigebied: **Livigno**  
**Intermonti**, 4 sterren, skigebied: **Livigno**

**…**

En van de klasse Aanbod

Bv.

Periode: **Kerstvakantie**, hotel: **La Torretta** , Prijs: 715  
Periode: **Kerstvakantie**, hotel: **San Carlo**, Prijs: 785  
Periode: **Kerstvakantie**, hotel: **Intermonti**, Prijs: 845

Pas nu de functies toe op de gemaakte objecten en toon het resultaat in de console.

# H22 GUI applicaties

# H23 XML

## H22\_Oefening\_01\_planten

Voor deze oefening maak je gebruik van het XML-bestand H23\_Data\_planten.xml.

Schrijf een programma dat het volgende overzicht maakt van de planten.

Afbeelding met tekst, tafel

Automatisch gegenereerde beschrijving

1. Afbeelding met tekst

   Automatisch gegenereerde beschrijvingPas je programma aan zodat je alleen de planten afdrukt die best in de zon staan.

## Afbeelding met tekst Automatisch gegenereerde beschrijvingH23\_Oefening\_02\_bioscopen

XML-files worden vaak verspreid door open data initiatieven. Voor deze oefening gebruik je het XML-bestand *H23\_Data\_bioscoopoverzicht.xml.* Dit bestand is verspreid als open data door de Stad Antwerpen. Verwerken en combineren van dergelijke open data sources en deze data ombuigen tot informatie die geschikt is voor analyse, is een belangrijk luik voor studenten die interesse hebben in de afstudeerrichting BIT (Business Information Technology).

Bestand *H23\_Data\_bioscoopoverzicht.xml* is niet mooi uitgelijnd. Het is aan jou om het bestand te interpreteren zodat je een lijst kan afdrukken van de Antwerpse cinemazalen. (de eerste 6 vind je in de schermafdruk)

## H23\_Oefening\_03\_jobkrant

Afbeelding met tafel

Automatisch gegenereerde beschrijving

Voor deze oefening maak je gebruik van het XML-bestand *H25\_Data\_jobkrant.xml.*

Schrijf een programma dat het volgende overzicht maakt.

Merk op: enkel de vacatures voor IT-jobs worden vermeld!

## H23\_Oefening\_04\_songs

Afbeelding met tekst

Automatisch gegenereerde beschrijvingAfbeelding met tekst

Automatisch gegenereerde beschrijvingVoor deze oefening maak je gebruik van het bestand *H23\_Data\_songs.txt.* Schrijf een programma dat dit txt-bestand omzet naar een XML- bestand songs.xml.

# H24 Regex

## H24\_Oefening\_01\_GSM\_nummer (\*)

op <https://manual.limesurvey.org/Using_regular_expressions/nl#:~:text=(02)%209123%206535> vind je vele regex patronen!

<https://regex101.com/> is een website waarop je regex patronen kan uittesten.

zo kan je hier bijvoorbeeld het patroon:

((\+|00)32\s?|0)4(60|[789]\d)(\s?\d{2}){3}

invoeren en checken of je GSM nummer 0477/436053 hieraan voldoet.

Schrijf een programma dat aan de gebruiker een GSM-nummer blijft vragen tot het voldoet aan dit Regex patroon.