

GRN

20-1-2019

Documentatie Python Codes

Andries de Boer, Benjamin de Jong, Niek Aukes en Douwe Osinga

# Inhoudsopgave

Inhoud

[Inhoudsopgave 1](#_Toc535778498)

[Documentatie 2](#_Toc535778499)

[Trapezium 2](#_Toc535778500)

[Letterbeeld 3](#_Toc535778501)

[Namenlijst 5](#_Toc535778502)

[Driehoek van Pascal 6](#_Toc535778503)

[Perfect Getal 7](#_Toc535778504)

[Proefwerk 8](#_Toc535778505)

[Inputs/Outputs van de codes 9](#_Toc535778506)

[Input/output Trapezium 9](#_Toc535778507)

[Input/output Letterbeeld 10](#_Toc535778508)

# Documentatie

## Trapezium

15

14

12

11

10

9

8

7

5

4

3

2

1

Het doel van deze code is een trapezium te maken op basis van invoer van de gebruiker.

* Regel 1 & 14-15

Hier wordt een try/except-regel uitgedrukt. Als de gebruiker geen integer invoert (TypeError) of als de invoer een verkeerde waarde heeft (ValueError), activeert de code regels 14 en 15, waarin een error message uitgeschreven wordt. Het bijzondere is dat hier de exacte exceptie wordt geprint.

* Regel 2-5

Hier wordt de gebruiker om input gevraagd voor de getallen N, V, en M

* Regel 7-9

Als de cijfers in de goede gebieden zijn, print de code voor elke rij het aantal beginnende sterretjes N plus het aantal sterretjes V wat er per rij bij moet komen. De i dient hier als indicator van hoeveel rijen al geprint zijn, zodat er geen rechthoek ontstaat.

## Letterbeeld

1

14

13

12

11

10

9

8

7

6

5

4

3

2

Het doel van Letterbeeld is om de rondjes en uiteinden te tellen van een door de gebruiker opgegeven woord.

* Regel 1

Hier wordt de gebruiker gevraagd om een woord.

* Regel 2 & 3

In regel 2 en 3 worden variabelen geclassificeerd.

* Regel 4

In regel 4 word een variabele ‘check’ genoemd met als value het alfabet. Dit zijn alle karakters waarop het programma het woord gaat checken.

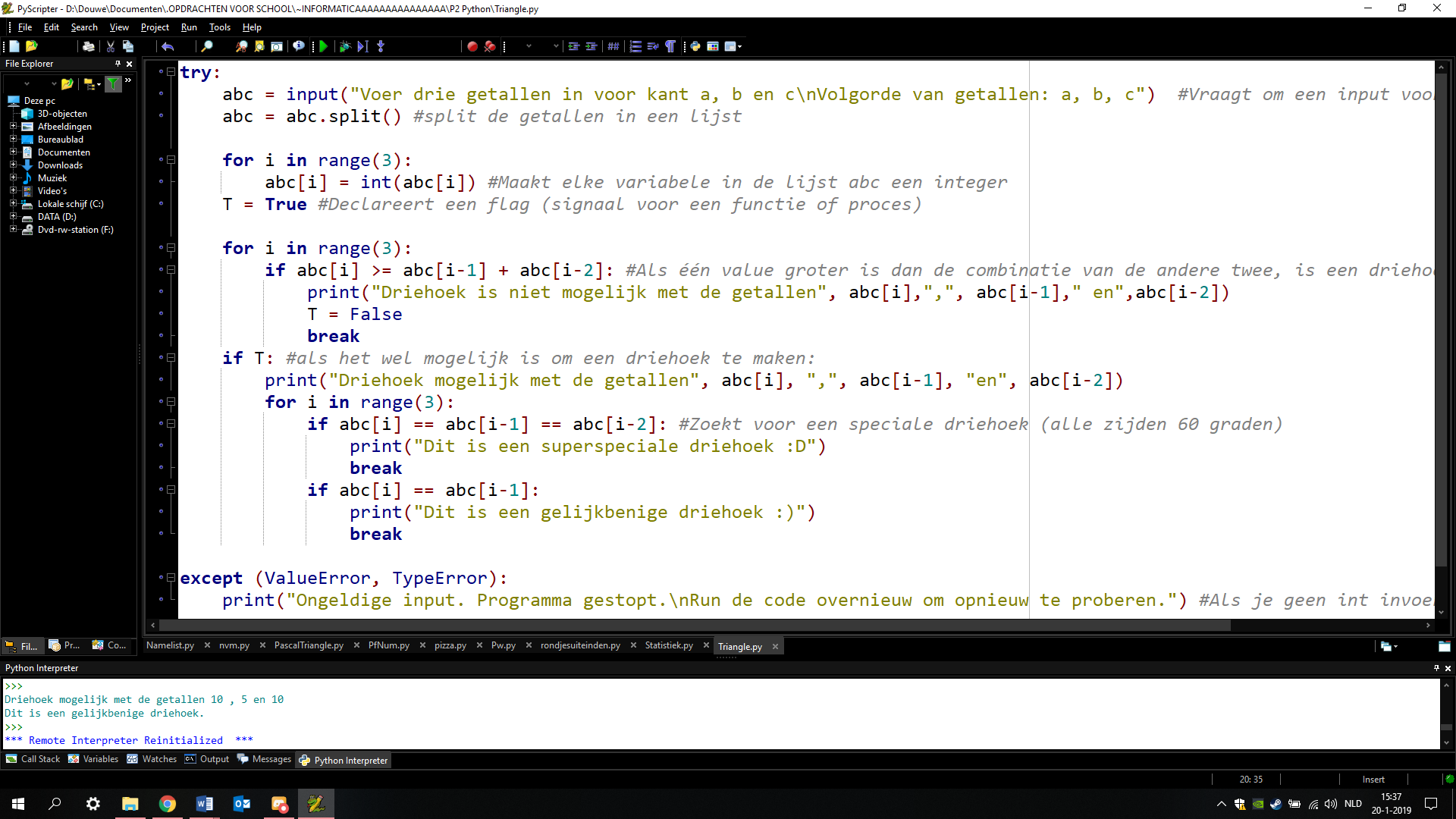
* Regel 5 en 6

Hier wordt gelijk aan het alfabet de rondjes en uiteinden van elke letter genoemd.

* Regel 7-11

Hier wordt voor elke letter in het woord gecheckt of het in de lijst ‘check’ staat. Als niet, dan wordt het woord geskipt. Als het in de lijst, en dus in het alfabet staat, dan telt het de rondjes en uiteinden van die letter bij het totaal op.

* Regel 12-14

Hier wordt het woord opnieuw geprint, samen met de rondjes en uiteinden van het ingevoerde woord.Driehoek

1

7

9

13

14

16

20

24

De code driehoek berekent of een driehoek mogelijk is met drie getallen die de gebruiker invoert.

Dit doet de code zo:

* Regel 1-7:

De gebruiker wordt gevraagd om drie getallen in te voeren, die de code probeert om te zetten naar een list met drie integers.

Er wordt alvast een boolean ‘T’ aangemaakt, die het value ‘True’ krijgt als het mogelijk is een driehoek te maken.

* Regel 9-13:

De code checkt of getal a, b of c niet groter of gelijk is aan de som van de overige getallen. Als dat wel het geval is, wordt er op het scherm “Driehoek is niet mogelijk” geprint met de bijbehorende ingevoerde getallen. ‘T’ wordt een value ‘False’ gegeven, zodat de rest van de code niet runt.

* Regel 14-23:

Als ‘T’ nog steeds True is (van regel 1-7) worden de regels 14 tot en met 23 geactiveerd. Regel 15 geeft aan de gebruiker aan dat een driehoek mogelijk is door de ‘print’-functie die daar staat. Bij regel 16-19 wordt geprobeerd of de driehoek drie gelijke zijden heeft. Als dat zo is, wordt op het scherm de tekst “Dit is een superspeciale driehoek :D” geprint.

Bij regel 20-23 checkt de code of twee van de drie zijden gelijk aan mekaar zijn. In dat geval print het programma “Dit is een gelijkbenige driehoek : )”

## Namenlijst

1

10

9

8

5

12

15

13

11

7

6

4

3

2

De code Namenlijst geeft op basis van de input van de gebruiker een lijst met alle namen, hoeveel die namen voorkomen, en hoeveel unieke namen er zijn.

* Regel 1

Het programma vraagt om input van de gebruiker, en slaat die op in een variabele ‘names’.

* Regel 2

Het programma maakt een dictionary aan (een lijst met twee variabelen, dus in deze code bijvoorbeeld een lijst ‘i’ met variabele lijst ‘seen’ in een).

* Regel 3

In deze regel maakt een variabele ‘uniq’ aan met value ‘0’. Deze variabele staat voor hoeveel unieke namen er zijn, vandaar de naam ‘uniq’.

* Regel 4-11

In regels 4 en 5 split de code de namen die ingevoerd waren en maakt een parallelle lijst aan, zodat het programma in de regels 6 tot en met 11 stuk voor stuk de namen in een lijst kan toevoegen.

In regels 6 tot en met 11 vraagt het programma voor elke naam die ingevoerd is, of die al in de lijst staat. Als de naam niet in de lijst staat

(regel 7), voegt de code het toe aan de lijst (regel 9), en krijgt het een uniekheid van één meer dan het huidige getal (regel 8).

* Regel 12

In regel 12 wordt de parallelle namenlijst (nu met alle namen erin, en hun uniekheid) gesorteerd.

* Regel 13 - 15

Hier wordt voor elke naam in de parallelle namenlijst de naam en de uniekheid geprint, met een streepje ertussenin. Ook wordt de uniekheid geprint.

## Driehoek van Pascal

11

12

8

7

6

5

4

10

9

3

2

1

De code Driehoek van Pascal maakt een driehoek van Pascal van het aantal rijen dat de gebruiker ingevoerd heeft.

* Regel 1

Hier vraagt de code om input van de gebruiker, die opgeslagen wordt in een variabele In.

* Regel 3 - 7

In regel 3 wordt een functie T gedefinieerd met parameters n en k (door Niek zijn gewoonten staat hier \_n en \_k). Deze functie maakt de driehoek van Pascal. Wilt u meer hierover weten, ga naar <https://nl.wikipedia.org/wiki/Driehoek_van_Pascal>.

Hier is die formule in een functie gepast. Deze wordt gebruikt om de positie van het cijfer in de driehoek te bepalen. Als het cijfer zich aan de rand van de drie bevindt, returnt de functie 1. Anders berekent de functie het getal

* Regel 8-12

Voor elke lijn wordt een lijst ‘nk’ aangemaakt. De cijfers worden hier berekend door alle cijfers per rij te berekenen:

Voor elke rij wordt de uitkomst van de functie ‘T’ aan de lijst ‘nk’ toegevoegd. Die wordt uitgeprint, zodat de driehoek van Pascal ontstaat.

## Perfect Getal

2

1

4

3

6

5

7

8

10

9

11

13

12

14

De code Perfect Getal heeft als doel het uitrekenen of het ingevoerde getal wel of niet een perfect getal is.

* Regel 1 & 2

Vraagt om een input en zet die om in een integer (we hadden ook gewoon int(input(“”)) kunnen doen, maar ach).

* Regel 3-7

Declareert een lege lijst ‘d’, waar in regel 6 en 7 alle delers in worden gezet.

In regel 4 tot en met 7 worden de delers uitgerekend.

* Regel 8-12

Print het perfecte nummer en zijn delers

* Regel 13 & 14

Als het getal niet een perfect getal is, wordt dat op het scherm geprint.

## Proefwerk

49

42

34

27

26

25

24

20

19

13

8

1

Proefwerk is ervoor om te zorgen dat de gebruiker even wat entertainment heeft. Het vraagt vragen, waarvan het antwoord opgeslagen wordt, waar een score uit komt.

* Regel 1

Er wordt een dictionary ‘normques’ gemaakt met een paar vragen.

* Regel 8

Er wordt een dictionary ‘diffques’ gemaakt met lastige vragen.

* Regel 13

Er wordt een dictionary ‘easyques’ gemaakt met makkelijke vragen.

# Inputs/Outputs van de codes

## Input/output Trapezium

Input:

N: 10, M: 5, V: 2

Output:

\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Input:

N:25, M:10 , V:4

Output:

The numbers were not in the specified range. Cancelled the action.

Input:

N: Vijfentwinting, M: Tien, V: Vier

Output:

Error occured! Try again. invalid literal for int() with base 10: 'vijfentwintig'

## Input/output Letterbeeld

Input:

COMPUTERNERD

Output:

COMPUTERNERD

Rondjes: 5

Uiteinden: 23

Input:

PRIKKELZINTUIG

Output:

PRIKKELZINTUIG

Rondjes: 2

Uiteinden: 34

Input:

Volgens mij kan dit niet goed komen

Output:

Volgens mij kan dit niet goed komen

Rondjes: 0

Uiteinden: 2 (Want hij telt de kleine letters niet)