Opdrachten Python

# Berekeningen

Naast de operator + die je net gezien hebt, kent Python ook de andere rekenkundige operatoren die je van wiskunde kent.

|  |  |
| --- | --- |
| + | optellen |
| - | aftrekken |
| \* | vermenigvuldigen |
| / | delen |
| \*\* | machtsverheffen |

**Opdracht:** Schrijf een stukje Python-code die met elk van deze operatoren een berekening maakt. Laat Python bijvoorbeeld uitrekenen en afdrukken hoeveel 5 + 3, 5 - 3, 5 × 3, en 53 is.

Je hebt gezien dat Python een breuk als decimaal getal aangeeft. Soms wil je echter alleen zien wat het gehele deel ervan is, net zoals je op de basisschool leerde delen; daar is het immers zo dat 5 gedeeld door 3 gelijk is aan 2, met rest 1. Python heeft hier ook operatoren voor.

|  |  |
| --- | --- |
| // | delen zonder rest |
| % | rest na deling (modulo) |

**Opdracht:** Schrijf een stukje Python-code die met deze twee operatoren een berekening maakt. Laat Python bijvoorbeeld uitrekenen en afdrukken hoeveel 8 gedeeld door 3 is zonder rest, en welke rest je dan overhoudt.

Soms wil je weten of een getal deelbaar is door een ander getal. 8 is bijvoorbeeld deelbaar door 4, omdat daar precies 2 uit komt zonder rest. 8 is echter niet deelbaar door 3, omdat je dan rest 2 overhoudt. Je kunt hiervoor de modulo-operator gebruiken; als daar 0 uitkomt, dan was het getal dat ervoor stond deelbaar door het getal wat erna staat.

**Opdracht:** Laat Python uitrekenen of 91606 deelbaar is door 281.

Je kan de modulo-operator ook gebruiken om te tellen zoals bij een klok: de uren in een klok lopen van 0 tot en met 23. Als je bij 23 uur nog een uur optelt, kom je in de volgende dag en dan is het weer 0 uur. Dit kan je berekenen door onderstaande code.

**print**((23 + 1) % 24)

Je ziet dat er haakjes gebruikt worden om te zorgen dat de optelling eerst gedaan wordt. Het getal 24 slaat hier op het aantal uren in een dag; na het getal 23 komt er hier dus weer het getal 0, en zo heb je precies 24 mogelijkheden.

**Opdracht:** Laat Python uitrekenen hoe laat het is 37 uur na 15 uur ’s middags.

# If-statements

Je hebt gezien hoe je met een if-statement keuzes kan maken aan de hand van de waardes van variabelen. Je hebt al een paar manieren gezien om twee waarden te vergelijken, maar er zijn er nog meer. Deze kan je vinden in de onderstaande tabel.

|  |  |
| --- | --- |
| == | is gelijk aan |
| != | is ongelijk aan |
| > | is groter dan |
| >= | is groter dan of gelijk aan |
| < | is kleiner dan |
| <= | is kleiner dan of gelijk aan |

**Opdracht:** Schrijf een stukje Python-code dat je leeftijd in de variabele leeftijd zet, en dan vervolgens door middel van een if-statement kijkt of je ouder dan 12 bent. Als dat zo is, moet de code je leeftijd printen.

**Opdracht:** Schrijf een stukje Python-code dat kijkt of je leeftijd deelbaar is door 5. Als dat zo is, dan moet de code je leeftijd printen. Onthoud dat deelbaar zijn betekent dat de rest na deling gelijk is aan 0; je kan de rest na deling berekenen via de modulo-operator.

Je hebt nu gezien hoe je iets kan doen als de vergelijking waar is. Je kan Python ook nog wat code laten uitvoeren als de vergelijking niet waar is. Dat gaat met een else-statement, zoals in onderstaand voorbeeld.

**if** leeftijd >= 12:

**print**('waar')

**else**:

**print**('onwaar')

**Opdracht:** Jan is 15 jaar en Piet 16. Schrijf een stukje Python-code waar je hun leeftijden in de variabelen jan en piet zet, en je een if-statement gebruikt om te kijken of Jan en Piet even oud zijn. Als ze even oud zijn, moet je Jan en Piet zijn even oud printen. Anders moet je Jan en Piet zijn niet even oud printen. Verander om allebei de gevallen te testen de leeftijd van Jan naar 16 en voer je code opnieuw uit.

# Strings en lijsten

Je hebt gezien hoe je een stukje tekst in een variabele kan zetten door een string te gebruiken, en hoe je letters kan opzoeken in een string, en je omgekeerd de letter kan krijgen die op een bepaalde positie in de string staat.

**Opdracht:** Schrijf een stukje Python-code dat je naam in een variabele naam zet, en daar de eerste letter van print. Onthoud dat Python bij nul begint te tellen voor de positie in een string.

**Opdracht:** Schrijf een stukje Python-code om uit te printen hoe lang je naam is. Laat Python dit berekenen.

**Opdracht:** Schrijf een stukje Python-code dat print op welke plek de letter o voorkomt in het woord Hanzehogeschool. Laat dit door Python berekenen, door middel van de method find die je gezien hebt. Zoals je kan zien komt de letter o vaker in dat woord voor, wat valt je op aan het resultaat?

Je kan ook negatieve getallen gebruiken om de string te *indexeren*. Probeer wat er gebeurt als je onderstaande code uitvoert. Wat valt je op?

**print**('hanze'[-1])

**print**('woord'[-2])

**print**('python'[-3])

**Opdracht:** Schrijf een stukje Python-code dat de laatste letter van je naam print, maar natuurlijk zonder dat je zelf bepaalt wat de laatste letter is.

Je hebt net gezien dat strings eigenlijk lijsten van letters zijn. In Python kan je ook lijsten van getallen gebruiken. Die schrijf je op door de getallen met komma’s achter elkaar te zetten, en er dan blokhaken omheen te zetten, zoals in het onderstaande voorbeeld.

lijst = [1, 2, 3]

Je kan lijsten op veel plekken in Python op dezelfde manier gebruiken als strings. Je kan bijvoorbeeld net als bij strings een enkel getal uit de lijst pakken door deze te *indexeren*.

**Opdracht:** Probeer een lijst te maken de leeftijden van jou en twee andere scholieren. Je krijgt dan een lijst met drie elementen. Schrijf dan een stukje Python-code om de middelste van de drie te printen.

Net zoals je de methode find kan gebruiken om een letter in een string te zoeken, kan je ook de methode index gebruiken om een getal in een lijst te zoeken.

**Opdracht:** Schrijf een stukje Python-code om in de lijst van de vorige opdracht de plek van je eigen leeftijd te vinden.

# For-statements

Zoals je gezien hebt kan je een for-statement gebruiken om een stukje code uit te voeren voor elke letter in een string. Je hebt ook gezien hoe je strings kan samenvoegen door de operator + te gebruiken.

**Opdracht:** Schrijf een stukje Python-code om een string om te draaien en de omgedraaide string uit te printen. Kan je een letter misschien ook aan het begin van een bestaande string toevoegen?

Je kan for niet alleen gebruiken om de letters in een string langs te gaan, maar ook om de elementen van een lijst langs te gaan. Dit werkt op dezelfde manier als bij een string.

**Opdracht:** Schrijf een stukje Python-code om door middel van een for-statement de kwadraten van de getallen 1 tot en met 5 uit te printen. Welke lijst zou je hiervoor kunnen maken? En hoe kan je van een getal zijn kwadraat berekenen?

# Caesarcijfer uitbreiden

De code van de functie caesar geven we hieronder:

def caesar(woord, verschuiving):

    alfabet = 'abcdefghijklmnopqrstuvwxyz'

    lengte = len(alfabet)

    antwoord = ''

    for letter in woord:

        nummer = alfabet.find(letter)

        nummer = nummer + verschuiving

        if nummer >= lengte:

            nummer = nummer - lengte

            letter = alfabet[nummer]

        antwoord = antwoord + letter

    return antwoord

Je hebt gezien hoe je in Python een Caesarcijfer kan toepassen. Maar wat zou er gebeuren als je bijvoorbeeld een spatie of een hoofdletter probeert te versleutelen?

**Opdracht:** Kijk wat er gebeurt als je een spatie probeert op te zoeken in de variabele alfabet met de methode find. Welke waarde komt hieruit? Probeer dit ook met een hoofdletter.

Je ziet dat je aan de uitkomst van de methode find kan zien of de letter in de string voorkomt. Als een letter niet voorkomt, kunnen we besluiten deze over te slaan bij het versleutelen. Deze keuze kan je programmeren met een if-statement.

**Opdracht:** Pas de functie caesar aan zodat deze ook werkt als er een letter of teken in de parameter woord zit die niet in de variabele alfabet zit.

Als je een hoofdletter tegenkomt, wil je deze misschien toch versleutelen. Je kan er dan voor kiezen deze om te zetten naar een kleine letter. Je kan dat doen door middel van de methode lower, zoals in het onderstaande voorbeeld.

woord = 'Hanzehogeschool Groningen'

**print**(woord.lower())

**Opdracht:** Kijk wat er gebeurt als je bovenstaande code uitvoert. Probeer aan de hand hiervan de functie caesar zo aan te passen dat deze ook werkt als er een hoofdletter in de parameter woord zit.

Als je nog tijd over hebt, volgen nu nog bonusopdrachten

**Opdracht:** In plaats van het hele woord kun je ook per letter een lower maken met letter.lower().

Met letter.upper() maak je van een letter weer een hoofdletter. Met letter.isupper() kan ik vragen of een letter een hoofdletter is. Om te onthouden of een letter een hoofdletter is kan je het bewaren in een variabele: isHoofdletter = letter.isupper()

Pas het programma nu aan zodat de hoofdletters ook hoofdletters blijven.

# FizzBuzz

Dit is een bonusopdracht. FizzBuzz is een heel bekende opdracht in de ICT, die wel gebruikt wordt bij sollicitatiegesprekken om te kijken of iemand een beetje kan programmeren.

**Opdracht:** Schrijf een stuk Python-code die alle getallen van 1 tot 100 onder elkaar zet. Echter, als het getal deelbaar is door 3, moet je niet het getal printen, maar het woord Fizz. Als het getal deelbaar is door 5, moet je in plaats van het getal het woord Buzz printen. Als het getal deelbaar is door 5 en door 3, dan moet je in plaats van het getal het woord FizzBuzz printen.

Je kan om een lijst van getallen van 1 tot en met 100 te maken, de functie range gebruiken. Deze kan je in een for-lus gebruiken om alle getallen bij langs te gaan, zoals hieronder staat:

**for** nummer **in** range(1, 101):

Hint: google naar modulo