Lesbrief 3 Pandas

[**Pandas Series**](#_jkfg19dy2yg) **1**

[Selectie](#_55uxxmfa2vj) 4

[Selectie met meerdere voorwaarden](#_bei81no9lds0) 5

[Functies voor het samenvatten van een Series](#_x7yo116j2gg) 6

[Loopen door een pandas Series](#_t0ihhb4i7aj8) 7

# Pandas Series

We gaan in deze module gebruik maken van de library pandas. Met pandas kun je vrijwel toveren met data! De pandas library bevat namelijk alles wat je nodig hebt om speciale berekeningen te doen over je data en dit op een duidelijke manier in een grafiek weer te geven. Als je met pandas aan de slag gaat moet je leren werken met twee belangrijke objecten :

* pandas Series
* pandas DataFrame

We beginnen met de pandas Series …

In lesbrief 1 heb je leren werken met Python lijsten. Pandas Series is ook een lijst, maar dan een bijzondere! Laten we eerst zo'n Series maken:

**Code:**

|  |
| --- |
| **import** pandas **as** pd  sLijst = pd.Series([23,43,102,76,8,98,90])  print(sLijst) |

**Uitvoer:**

|  |
| --- |
| 0 23  1 43  2 102  3 76  4 8  5 98  6 90 |

Zoals je ziet krijg je bij het printen van een Series ook meteen de bijbehorende indexen te zien.

|  |
| --- |
| **Verdiepingsblokje Index en waardenlijst in een Series**  Een pandas Series bestaat uit twee gekoppelde Python-lists: de *index* en de *waarden-lijst*. Elk element in de Series heeft een *label* (in de index) en een bijbehorende waarde (in de waarden-lijst).  s0 = pd.Series([131, 12, -19, 37, 5])  print(s0)  0 131  1 12  2 -19  3 37  4 5  dtype: int64  Je kunt deze lijsten opvragen als attribuut van een Series s: s.index en s.values.  print(s0.values)  print(s0.index)  array([131, 12, -19, 37, 5])  RangeIndex(start=0, stop=5, step=1)  De koppeling tussen beide lijsten zie je bijvoorbeeld als je de Series sorteert. Je kunt sorteren op de waarden:  s0.sort\_values()  print(s0)  2 -19  4 5  1 12  3 37  0 131  dtype: int64  Je kunt ook sorteren op de index; we krijgen hier de oorspronkelijke volgorde weer terug.  s0.sort\_index()  print(s0)  0 131  1 12  2 -19  3 37  4 5  dtype: int64  Bij het maken van een Series kun je zowel de waarden-lijst als de index opgeven, bijvoorbeeld:  s1 = pd.Series([1, 4, 9, 16, 26], index=['a', 'b', 'c', 'd', 'e'])  Als je de index weglaat, wordt bij een waarden-lijst van lengte N automatisch een index van de juiste lengte gemaakt, als list(range(N)) |

Je kunt net als met lijsten een specifieke index uitprinten:

**Code:**

|  |
| --- |
| **print** (sLijst[3]) |

**Uitvoer:**

|  |
| --- |
| 76 |

Je kunt natuurlijk ook de eerste vier getallen uitprinten:

**Code:**

|  |
| --- |
| **import** pandas **as** pd  sNum = pd.Series([23,43,102,76,8,98,90])  print(sNum[0:4]) |

**Uitvoer:**

|  |
| --- |
| 0 23  1 43  2 102  3 76 |

**Rekenen met Series**

Je kunt heel handig rekenen met Series. Stel dat je een Series van getallen hebt en je wilt alle getallen vermenigvuldigen met 3. Dan kan dat heel makkelijk:

**Code:**

|  |
| --- |
| **import** pandas **as** pd sNum = pd.Series([23, 43, 102, 76, 8, 98, 90]) print(sNum \* 3) |

**Uitvoer:**

|  |
| --- |
| 0 69  1 129  2 306  3 228  4 24  5 294  6 270 |

Lekker handig, dat pandas!

**Optellen, aftrekken, vermenigvuldigen en delen**

Je kunt zelfs twee lijsten bij elkaar optellen of door elkaar delen of met elkaar vermenigvuldigen. Probeer maar eens uit:

**Code:**

|  |
| --- |
| **import** pandas **as** pd  sNum = pd.Series([23,43,102,76,8,98,90]) sNum2 = pd.Series([2,2,3,2,1,4,5])  print(sNum \* sNum2) |

**Uitvoer:**

|  |
| --- |
| 0 46  1 86  2 306  3 152  4 8  5 392  6 450 |

**LET OP**: Als de twee Series niet even lang zijn dan wordt het resultaat aangevuld met NaN-waarden. NaN betekent *Not a Number* en wordt in pandas gebruikt voor een niet ingevulde waarde.

## Selectie

Soms wil je maar een deel van je Series laten zien. Met Python lijsten kon je slicen. Dat kan ook met Series. Kijk maar eens:

|  |
| --- |
| **import** pandas **as** pd sNum = pd.Series([23,43,102,76,8,98,90]) print(sNum[0:3]) |

|  |
| --- |
| 0 23  1 43  2 102 |

Of we selecteren de laatste drie elementen uit de Series:

|  |
| --- |
| print(sNum[-3:]) |

Maar we kunnen meer. Stel dat je alle getallen die groter dan 30 zijn wilt selecteren?

Dit kun je met pandas heel erg eenvoudig. Kijk maar eens!

|  |
| --- |
| **import** pandas **as** pd sNum = pd.Series([23,43,102,76,8,98,90])  **print** (sNum[sNum > 30]) |

**Uitvoer:**

|  |
| --- |
| 1 43  2 102  3 76  5 98  6 90 |

De uitdrukking sNum > 30 levert een Series op met boolean waarden. Deze wordt hierboven gebruikt als selectie van sNum. Je kunt sNum > 30 ook gewoon printen.

**Code:**

|  |
| --- |
| **import** pandas **as** pd sNum = pd.Series([23,43,102,76,8,98,90])  **print** (sNum > 30) |

**Uitvoer:**

|  |
| --- |
| 0 False  1 True  2 True  3 True  4 False  5 True  6 True |

### Selectie met meerdere voorwaarden

Je kunt zelfs aangeven dat je alle getallen tussen 30 en 50 wilt hebben. Dat doe je met de **&** (betekent AND). De regel sNum[(sNum > 30) & (sNum < 50)] betekent dus: Als sNum groter is dan 30 *en* ook kleiner dan 50, dan selecteren we hem. Kijk maar eens:

|  |
| --- |
| **import** pandas **as** pd  sNum = pd.Series([23,43,102,76,8,98,90]) **print** (sNum[(sNum > 30) & (sNum < 50)]) |

**Uitvoer:**

|  |
| --- |
| 1 43 |

**Opdracht**

Geef de code waarmee je alle getallen, die kleiner dan 20 of groter dan 100 zijn, in een Series zet. Print de Series uit.

**TIP**: Naast de & (and) hebben we ook de | (**or**). Deze | heb je nodig om de opgave te kunnen maken.

## Functies voor het samenvatten van een Series

Pandas heeft een aantal superhandige functies dat je kunt gebruiken om bijvoorbeeld het gemiddelde van een Series met getallen te berekenen. Dit doe je met de functie mean()

**Code:**

|  |
| --- |
| **import** pandas **as** pd sNum = pd.Series([23,43,102,76,8,98,90]) **print** ("Het gemiddelde is", sNum.mean()) |

**Uitvoer:**

|  |
| --- |
| Het gemiddelde is 62.8571428571 |

Naast de functie mean() heb je ook de volgende functies:

|  |  |
| --- | --- |
| count() | Geeft het aantal waarden in de Series |
| sum() | Telt de waarden in de Series bij elkaar op |
| min() | Pakt de minimumwaarde uit de Series |
| max() | Pakt de maximumwaarde uit de Series |

**Opdracht**

Henk heeft tot nu toe de volgende cijfers behaald: 6, 7, 8, 4, 5.5, 6.5, 7, 2, 5, 4, 3, 8, 9, 9.8, 9.7, 9.4, 2, 5, 7, 8, 6.2, 5.7, 8, 9

Schrijf een programma waarbij je een pandas Series maakt met deze cijfers erin. Vervolgens de volgende gegevens uitprint (Python moet ze allemaal berekenen):

* Hoeveel cijfers heeft hij behaald?
* Wat is het gemiddelde van al zijn cijfers?
* Wat is het laagste cijfer dat hij ooit heeft gehaald?
* Wat is het hoogste cijfer dat hij ooit heeft gehaald?

## Loopen door een pandas Series

Je kunt ook vrij eenvoudig loopen door een pandas Series lijst. Hieronder een voorbeeld:

**Code:**

|  |
| --- |
| sNum = pd.Series([23,43,102,76,8,98,90]) **for** item **in** sNum:  **print** ("Element: ", item) |

**Uitvoer:**

|  |
| --- |
| Element 23  Element 43  Element 102  Element 76  Element 8  Element 98  Element 90 |