Python les-materialen Yilmaz Mustafa, Instructeur Java/Python

# Getallen en meer in Python!

In deze lezing leren we over getallen in Python en hoe we ze kunnen gebruiken.

We leren over de volgende onderwerpen:

1.) Soorten getallen in Python  
 2.) Basis rekenen  
 3.) Verschillen tussen klassieke indeling en vloerindeling  
 4.) Objecttoewijzing (EN: assignments) in Python

## Soorten nummers

Python heeft verschillende “types” getallen (numerieke letterlijke waarden). We zullen ons vooral concentreren op gehele getallen en getallen met drijvende komma.

Gehele getallen zijn slechts gehele getallen, positief of negatief. Bijvoorbeeld: 2 en -2 zijn voorbeelden van gehele getallen.

Drijvende-kommagetallen in Python zijn opmerkelijk omdat ze een decimaalteken bevatten of een exponentiële (e) gebruiken om het getal te definiëren. 2.0 en -2.1 zijn bijvoorbeeld voorbeelden van getallen met drijvende komma. 4E2 (4 keer 10 tot de macht 2) is ook een voorbeeld van een getal met drijvende komma in Python.

Gedurende deze cursus zullen we werken met gehele getallen of eenvoudige float-nummertypes.

Hier is een tabel met de twee hoofdtypen die we het grootste deel van onze tijd zullen besteden aan het werken met enkele voorbeelden:

Voorbeelden

Nummer-“Type”

1,2,-5,1000

Integers

1.2,-0.5,2e2,3E2

Floating-point nummers

Laten we nu beginnen met wat basisrekenkunde.

### Basis rekenen

# Sum   
100 + 200

# Sum  
2+1

3

# Aftrekken  
2-1

1

# Vermenigvuldiging  
2\*2

4

# Divisie  
3/2

1.5

# Verdiepingsverdeling (floor division)  
7//4

1

**Wauw! Wat is er zojuist gebeurd? De laatste keer dat ik keek, is 7 gedeeld door 4 gelijk aan 1,75 niet 1!**

De reden dat we dit resultaat krijgen, is omdat we de divisie “*verdiepings (floor division)*” gebruiken. De //-operator (twee schuine strepen naar voren) kapt het decimaalteken af zonder afronding en retourneert een geheel getal.

**Dus wat als we gewoon de rest willen na de deling?**

# Modulo  
7%4

3

4 gaat één keer in 7, met een rest van 3. De %-operator retourneert de rest na deling.

# Complexer Calculations  
(5\*4)+(2\*10)+(10/10)

41.0

### Rekenkunde vervolg

# Machten  
2\*\*3

8

# Wortels  
4\*\*0.5

2.0

# Volgorde en prioriteit van instructies in Python  
2 + 10 \* 10 + 3

105

# Kan haakjes gebruiken om het volgorde te specificeren  
(2+10) \* (10+3)

156

## Variabele Toewijzingen (Assignments)

Nu we hebben gezien hoe we getallen in Python als rekenmachine kunnen gebruiken, laten we eens kijken hoe we namen kunnen toewijzen en variabelen kunnen maken.

We gebruiken een enkel gelijkteken om labels aan variabelen toe te wijzen. Laten we een paar voorbeelden bekijken van hoe we dit kunnen doen.

# Laten we een object maken met de naam "a" en het nummer 5 toewijzen  
a = 5

Als ik nu *a* in mijn Python-script aanroep, zal Python het behandelen als het getal 5.

# Het toevoegen van de objecten  
a+a

10

Wat gebeurt er bij herplaatsing? Laat Python het ons overschrijven?

# Hertoewijzing (Reassignment)  
a = 10

# Controleren  
a

10

Ja! Met Python kunt u over toegewezen variabelenamen schrijven. We kunnen de variabelen ook zelf gebruiken bij het opnieuw toewijzen. Hier is een voorbeeld van wat ik bedoel:

# Controleren  
a

10

# Gebruik a om a opnieuw te definiëren  
a = a + a

# Controleren   
a

20

De namen die u gebruikt bij het maken van deze labels, moeten aan een paar regels voldoen:

1. Namen mogen niet beginnen met een cijfer.  
 2. Er mogen geen spaties in de naam staan, gebruik in plaats daarvan \_.  
 3. Kan geen van deze symbolen gebruiken:'",<>/?|\()!@#$%^&\*~-+  
 4. Het wordt als best practice (PEP8) beschouwd dat namen in kleine letters zijn.  
 5. Vermijd het gebruik van de tekens 'l' (kleine letter l), 'O' (hoofdletter o), of 'I' (ı in hoofdletters) als variabelenamen van één teken.  
 6. Vermijd het gebruik van woorden die een speciale betekenis hebben in Python, zoals "list" en "str"

Het gebruik van variabelenamen kan een zeer handige manier zijn om verschillende variabelen in Python bij te houden. Bijvoorbeeld:

# Gebruik objectnamen om beter bij te houden wat er in uw code gebeurt!  
my\_income = 100  
  
tax\_rate = 0.1  
  
my\_taxes = my\_income\*tax\_rate

# Toon mijn belastingen!  
my\_taxes

10.0

Dus wat hebben we geleerd? We leerden enkele basisprincipes van getallen in Python. We hebben ook geleerd hoe we moeten rekenen en Python als basisrekenmachine kunnen gebruiken. Vervolgens hebben we het afgesloten met het leren over variabele toewijzing in Python.