

Inhoudsopgave

Voorwoord	i
Inhoudsopgave	iii
1 Inleiding	1
2 Installatie en eerste programma's	5
2.1 Installatie van Python	5
2.2 Installatie van PyCharm	6
2.3 Eerste programma's	7
3 Variabelen	13
3.1 Inleiding	13
3.2 Datatypes	15
3.2.1 Getallen	15
3.2.2 Strings	18
3.2.3 Lijsten	18
3.2.4 Tupels	19
3.2.5 Verzamelingen	20
3.2.6 Woordenboeken	21
3.3 Conversie van datatypes	22
3.3.1 Conversie naar een geheel getal	22
3.3.2 Conversie naar een reëel getal	23
3.3.3 Conversie naar een complex getal	23

3.3.4	Conversie naar een lijst	24
3.3.5	Conversie naar een tupel	24
3.3.6	Conversie naar een verzameling	25
3.3.7	Conversie naar een string	26
3.3.8	Overige conversies	26
4	Bewerkingen	31
4.1	Bewerkingen op getallen	31
4.1.1	Rekenkundige bewerkingen	31
4.1.2	Onderling vergelijken	33
4.1.3	Gecombineerd toewijzen	34
4.1.4	Logische bewerkingen	35
4.1.5	Binaire bewerkingen	36
4.1.6	Volgorde van bewerkingen	37
4.2	Bewerkingen op strings	37
4.3	Bewerkingen op lijsten	39
4.4	Bewerkingen op tupels	40
5	Toetsenbord en outputvenster	43
5.1	Printen naar outputvenster	43
5.2	Input van toetsenbord	45
6	Conditionele programmeertechnieken	47
6.1	if ... elif ... else	47
6.2	for ... in	50
6.3	while	54
6.4	break en continue	55
7	Basisalgoritmen	57
7.1	Zoeken van informatie	57
7.2	Bewijs van het tegendeel	62

8	Funcities	71
8.1	Structuur van een functie	72
8.2	Funcities en variabelen	75
8.3	Funcities en functies	79
8.4	Mogelijkheden voor parameters	82
9	Modules	85
9.1	Modules importeren	85
9.2	Enkele wiskundige modules	88
9.2.1	De module <code>math</code>	88
9.2.2	De module <code>cmath</code>	92
9.2.3	De module <code>random</code>	94
9.2.4	De module <code>statistics</code>	97
9.3	Tweedimensionale grafieken	98
10	Strings	103
10.1	Stringfuncties	103
10.1.1	<code>len</code> , <code>min</code> , <code>max</code> , <code>in</code> en <code>not in</code>	103
10.1.2	String slicing	104
10.2	Stringmethodes	106
10.2.1	Overzicht	106
10.2.2	Hoofdletters en kleine letters	108
10.2.3	Begin en einde van strings	108
10.2.4	Zoeken en vervangen	109
10.2.5	Uitlijnen en opvullen	110
10.2.6	Opsplitsen	111
10.2.7	Testen op soort inhoud	112
11	Lijsten	115
11.1	Funcities voor lijsten	115
11.1.1	<code>len</code> , <code>min</code> , <code>max</code> , <code>in</code> , <code>not in</code> en <code>del</code>	115

11.1.2 Slicing van lijsten	116
11.2 Methodes voor lijsten	120
12 Verzamelingen	125
12.1 Functies voor verzamelingen	125
12.1.1 len, min, max, in en not in	125
12.2 Methodes voor verzamelingen	127
12.2.1 Overzicht	127
12.2.2 Bewerkingen	128
12.2.3 Gecombineerd toewijzen	129
12.2.4 Testen	130
12.2.5 Overige methodes	131
13 Woordenboeken	133
14 Foutafhandeling	137
14.1 Syntaxfouten	137
14.2 Semantische fouten	139
14.3 Logische fouten	140
14.4 Fouten bij uitvoering	141
15 Input en output met bestanden	145
15.1 IO met tekstbestanden	145
15.2 IO met andere programma's	149
16 Recursie	157
16.1 Principe en inleidende voorbeelden	157
16.2 Enumeratie	166
16.3 Backtracking	168
17 Klassen	179
17.1 Principes van objectgeoriënteerd programmeren	179

17.2 Objectgeoriënteerd programmeren met Python	181
18 Complexiteit van algoritmen	189
18.1 Experimenteel meten van efficiëntie	189
18.2 Theoretische benadering van efficiëntie	193
19 Sorteren	199
19.1 De Pythonmethode sort	199
19.2 Insertion sort	200
19.3 Merge sort	203
20 Binaire bomen	211
20.1 Binaire zoekboom	211
20.2 Binaire heap	218
21 Hashing	223
21.1 Zoeken via hashing	224
21.2 Hashing en beveiliging	229
22 Toepassingen uit Operationeel Onderzoek	233
22.1 Kortste pad	233
22.2 Minimaal opspannende boom	246
22.3 Handelsreizigerprobleem	253
22.4 Knapzakprobleem	258
23 Programmeeropdrachten	263
23.1 Inleidende opdrachten	263
23.2 Programmeertechnieken en basisalgoritmen	264
23.3 Programmeertechnieken en functies	265
23.4 Modules en strings	267
23.5 Lijsten, verzamelingen en woordenboeken	269
23.6 Foutafhandeling en IO	272

23.7 Recursie	274
23.8 Klassen	276
Bibliografie	281
Index	285

Hoofdstuk 2

Installatie en eerste programma's

2.1 Installatie van Python

Python wordt beheerd door de Pythongemeenschap en heeft als centraal aanspreekpunt de website <https://www.python.org>. Daar vindt men een bron van informatie over die populaire programmeertaal. We nodigen de lezer uit om die website grondig te bekijken. Vooral de onderdelen 'About' en 'Documentation' zijn aanraders wanneer specifieke informatie moet worden gezocht. Maar doorgaans komt men ook op de juiste pagina's door eenvoudige zoekopdrachten op internet via jouw gebruikelijke zoekmachine.

Omdat programmeertalen nog steeds verder evolueren, worden versienummers toegevoegd. Er wordt meestal wel geprobeerd om compatibiliteit met een vroegere versie te behouden maar de praktijk leert dat dit toch niet steeds het geval is. Bij grote revisies gebeurt het wel eens dat er beslissingen worden genomen die niet compatibel zijn met eerdere ontwikkelingen. Meestal zijn die niet erg dramatisch maar het komt wel voor dat een programma geschreven voor een oudere versie lichtjes moet worden aangepast aan de nieuwere versie. Op het ogenblik van dit schrijven is de laatste stabiele versie van Python 3.6. Dat is de versie waarmee de voorbeelden uit deze cursustekst zijn geschreven en dus ook de versie waarmee bij voorkeur gewerkt wordt. Maar omdat er geen wezenlijke verschillen zijn tussen bijvoorbeeld versies 3.4, 3.5, 3.6 en 3.7 zijn ook deze perfect bruikbaar. Merk op dat dit niet kan gezegd worden van versie 2.7, de laatste stabiele versie van Python 2. Anders gezegd, het eerste volgnummer van de versie verwijst naar een grote revisie, de andere nummers naar kleinere, minder belangrijke revisies. Die kleinere revisies worden benut om de gebruikelijke bugs weg te werken.

Python 3 kan worden gedownload vanaf de website <https://www.python.org>. Volg daarvoor de link 'Download' en kies de distributie die voor het besturingssysteem van de computer geschikt is. Er bestaan distributies voor de courante platforms zoals Windows, Mac OS X en Linux. Installeer de download en kies hierbij de gebruikelijke opties (o.a. de map waarin alles moet worden geïnstalleerd en voor welke gebruikers die installatie geldt). Zorg dat men de nodige rechten heeft om de software

in de gekozen locatie te installeren. Op internet (YouTube) zijn er instructievideo's te vinden die alle facetten van de installatie duidelijk illustreren.

Bij de installatie van Python hoort de eenvoudige programmeeromgeving IDLE. IDLE is een zogenaamd shell-programma dat toelaat om eenvoudige instructies bestaande uit één enkele regel te laten vertalen en uitvoeren. Het resultaat wordt als nieuwe regel in hetzelfde venster getoond. Eenvoudige instructies kunnen dus met IDLE snel worden gecontroleerd. Ter controle van een correcte installatie starten we IDLE op. Er verschijnt een venster met daarin het versienummer van Python gevolgd door een commandolijn die begint met `>>>`. Typ het volgende commando:

```
print("Hallo, hier ben ik ...")
```

Python reageert door "Hallo, hier ben ik ..." op de volgende regel te plaatsen en een nieuwe commandolijn te tonen. Python wacht nu op verdere instructies. Wat experimenteren met IDLE kan zeker geen kwaad. Probeer enkele eenvoudige berekeningen zoals het optellen, vermenigvuldigen en delen van getallen. Klassieke ronde haakjes kunnen daarbij worden gebruikt. Wordt er iets ingevoerd dat Python niet begrijpt, dan wordt er een foutmelding gegenereerd.

De mogelijkheid om Python interactief te gebruiken via een shell-programma zoals IDLE volgt uit het feit dat Python behoort tot de familie van de interpreters. Eén enkele instructieregel volstaat om vertaald en uitgevoerd te worden. Bij compilers is het veel minder evident om een dergelijk gedrag te genereren. De interactieve mogelijkheid is vooral interessant voor het snel uittesten van bepaalde instructies.

2.2 Installatie van PyCharm

IDLE kan ook als volwaardige programmeeromgeving worden gebruikt maar gezien de eerder beperkte mogelijkheden opteren we voor een andere gebruikersinterface genaamd PyCharm. De thuisbasis van dit programma is te vinden op de website <https://www.jetbrains.com/pycharm>. Via de downloadknop komt men terecht op een keuzepagina. Kies het gewenste besturingssysteem en vervolgens de (gratis) Communityversie. Deze versie bevat voor de toepassingen in deze cursus meer dan voldoende mogelijkheden. Installeer vervolgens het programma en volg opnieuw de verschillende stappen. Ook van deze installatieprocedure zijn instructievideo's te vinden op YouTube. Sommige illustreren zelfs het gecombineerd installeren van Python en PyCharm. Zorg er evenwel voor dat Python steeds als eerste wordt geïnstalleerd. Op die manier is het mogelijk om in PyCharm de juiste verwijzingen naar Python aan te brengen.

Start vervolgens PyCharm op. De eerste maal zal PyCharm vragen naar een aantal initiële settings. Het eerste venster heeft betrekking op hoe het programma moet aanvoelen maar aangezien we nog geen ervaring hebben, behouden we de defaultinstellingen. Creëer vervolgens een nieuw project in een locatie naar keuze en met de naam 'Test'. Op de lijn 'Interpreter' staat de verwijzing naar de locatie waarin het uitvoerbare bestand van python zich bevindt. Mocht dat niet het geval

zijn, dan geeft men die locatie manueel in via de settingsknop (tandwiel), gevolgd door de optie 'Add local'. Er is nu een nieuw project gecreëerd in de bijhorende map. Deze map is momenteel nog leeg maar via 'File' → 'New' → 'File' kan een nieuw bestand worden toegevoegd. Creëer op deze manier het bestand 'test.py'.¹ Dat bestand wordt ook meteen geopend waardoor we meteen een instructie kunnen invoeren. Als test typen we hier opnieuw

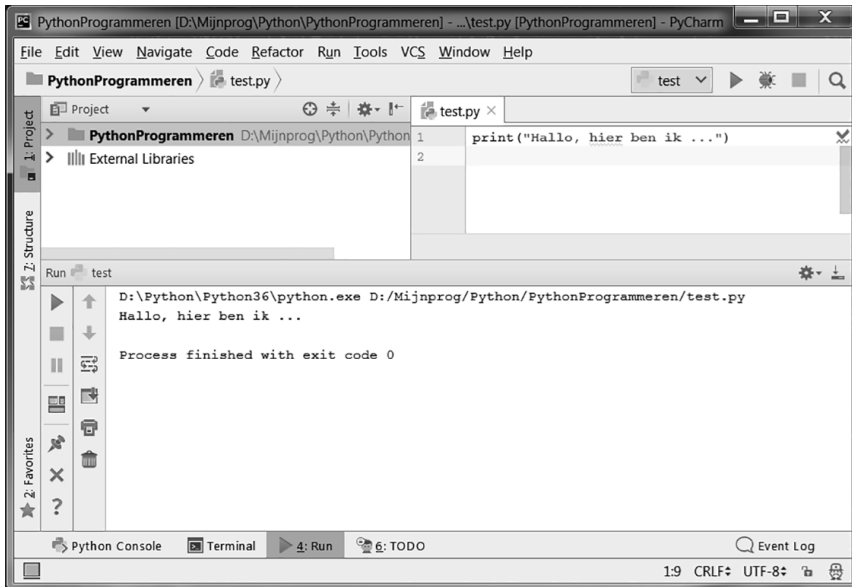
```
print("Hallo, hier ben ik ...")
```

Merk op dat hier geen commandolijn te zien is zoals in IDLE maar eerder een editor zoals we die kennen van tekstverwerkers. Deze omgeving laat dus toe om meerdere instructies en dus volledige programma's te schrijven in Python. Om het korte programma bestaande uit één enkele regel hierboven te laten lopen, kiezen we via het menu 'Run' → 'Run 'test'' of 'Run' → 'Run...'. Mocht de koppeling met de interpreter niet correct zijn uitgevoerd, dan is hier ook nog de mogelijkheid (bijvoorbeeld via 'Edit Configurations...') om de juiste locatie van Python aan te geven. Als het programma al eens is uitgevoerd, kan het op een alternatieve wijze opnieuw gestart worden met de "Play"-knop. De ene regel uit het programma wordt uitgevoerd en er verschijnt een outputvenster onderaan met daarin o.a. het zinnetje "Hallo, hier ben ik ...", gevolgd door de exit code 0. Deze code geeft aan dat het programma normaal is geëindigd.

2.3 Eerste programma's

Nadat we in PyCharm het eerste Pythonprogramma hebben uitgevoerd, ziet de grafische omgeving eruit als in Figuur 2.1. We kunnen drie grote regio's onderscheiden. In het venster linksboven (projectvenster) wordt een overzicht van het project getoond. In het voorbeeld is de naam van het project 'PythonProgrammeren' en het project bevat momenteel één bestand genaamd 'test.py'. Een project kan gezien worden als een container die alle bestanden bevat gerelateerd aan dat project. Het venster rechts van het projectoverzicht (editvenster) toont de inhoud van het Pythonbestand dat momenteel open is ('test.py'). Dat is ook de plaats waarin we Pythoncode zullen schrijven. PyCharm gedraagt zich in dat venster zoals een tekstverwerker maar dan aangepast aan de typische Pythonregels. Dat betekent onder meer dat bepaalde samenhangende onderdelen van de code worden ingekleurd, wat de leesbaarheid verhoogt. Bovendien zal men merken dat er syntaxhulp wordt getoond van zodra een aantal karakters van een instructie zijn getypt. Die hulp bestaat uit een lijst waaruit een specifieke syntax kan worden geselecteerd. Door vervolgens op "enter" te drukken, zal de overeenkomstige syntax integraal worden overgenomen. Het vraagt enige oefening om met een dergelijk systeem te werken, maar eens je het gewoon bent, versnelt het de productie van code enorm.

¹Noteer dat elk bestand dat Pythonprogrammacode bevat, de zogenaamde broncode, steeds de extensie 'py' bevat. Dat is strikt genomen niet essentieel maar het is een gangbare praktijk en maakt het terugvinden van programmabestanden gemakkelijker. Die extensie voegen we zelf toe zoals hier is beschreven.



Figuur 2.1: Grafische interface van PyCharm

Het onderste venster (outputvenster) bevat de output die door Python wordt gegenereerd als antwoord op het uitvoeren van de code. Zoals reeds aangegeven zal Python de exit code 0 geven wanneer er zich geen onregelmatigheden hebben voorgedaan. Het outputvenster bevat een aantal knoppen. Hun functionaliteit is doorgaans duidelijk aan de hand van hun pictogram en de contextgevoelige hulpinfo die wordt getoond. Vooral de lokale 'Play'-knop is interessant om het programma opnieuw te laten uitvoeren. Een dergelijke knop is overigens ook rechts bovenaan te vinden.

Het menu bevat allerlei hulpmiddelen bij het programmeren en mogelijkheden om de omgeving aan te passen aan specifieke gewoontes. Sommige ervan zullen we verderop bespreken bij de behandeling van een toepasselijk geval. Neem gerust de tijd om eens door de verschillende mogelijkheden te lopen. Ook de helpfunctie kan nuttig zijn als bron voor verdere informatie.

In het vervolg van deze tekst zullen tal van voorbeelden worden opgenomen ter illustratie van het gebruik van welbepaalde commando's. We gebruiken daarvoor steeds dezelfde stijl zoals gebruikt in het kleine voorbeeld hierna.

Voorbeeld 2.1.

```

1 a = [0 for i in range(11)]
2 for i in range(11):
3     a[i] = i**2
4 for i in range(11):
5     print(a[i], " ", end="")
6 print()

```

```
0 1 4 9 16 25 36 49 64 81 100
Process finished with exit code 0
```

Eerst ziet men het programma, dat is het deel dat wordt ingevoerd in het editvenster. Specifieke Pythoncommando's worden vetjes weergegeven. Er wordt ook een nummering van de verschillende lijnen voorzien om in de tekst gemakkelijk te kunnen refereren. Maar voor het uitvoeren van de code heeft die nummering geen nut. Indien men in PyCharm ook graag de nummering ziet, dan kan die worden geactiveerd via bijvoorbeeld een 'rechtsklik' in de linkermarge van het editvenster en het onderdeel 'Show Line Numbers' te activeren.

Vervolgens wordt de output getoond. Het voorbeeldprogramma berekent de kwadraten van de natuurlijke getallen tot en met 10 en toont die achter elkaar. De gebruikte instructies en structuren komen verderop nog aan bod maar als inleiding kan het nuttig zijn om aan de hand van dit voorbeeld enkele eenvoudige principes van Python uit te leggen.

- In lijn 1 komt het commando `range(11)` voor. Voor Python is dit een lijst van de natuurlijke getallen van 0 tot en met 10. Merk op dat 11 zelf hier niet bijzit. Het gaat dus om de natuurlijke getallen vanaf 0 tot één minder dan de waarde die is opgegeven. Via de volledige eerste lijn wordt een lijst, genaamd `a` gecreëerd bestaande uit tien keer 0. Het object aangegeven met de letter `a` is een variabele en die bevat de inhoud die aan de rechterkant van het gelijkheidsteken (`=`) voorkomt. Het gelijkheidsteken wordt in Python dus gebruikt om inhoud 'toe te wijzen' aan een variabele. Uit de constructie van het rechterlid wordt duidelijk dat het om een lijst gaat. Dat is te merken aan de uiterste vierkante haken. Binnen die haken lezen we dat 0 moet worden ingevuld voor alle posities `i` in de lijst gelegen tussen 0 en 10. Noteer dat de nummering van items in een lijst in Python steeds vanaf 0 begint.
 - In lijnen 2 en 3 komt een lus voor. De instructie die ingesprongen voorkomt op lijn 3 zal worden uitgevoerd voor alle natuurlijke waarden van `i` gelegen tussen 0 en 10. Concreet betekent dit dat elke positie beschikbaar in de lijst `a` zal voorzien worden van het kwadraat van `i`. Machtsverheffing wordt in Python immers aangegeven met `**`. Verder is aan lijn 3 ook duidelijk te merken dat de nummering van lijsten steeds vanaf 0 begint. Aangezien `a` reeds een inhoud gekregen heeft in lijn 1, wordt deze nu overschreven op basis van de lus die voorkomt op lijnen 2 en 3. Voor meer informatie over het gebruik van lussen verwijzen we naar Sectie 6.2 op pagina 50.
- In lijn 2 staat de syntax voor de creatie van de lus. De instructie `in` geeft aan dat `i` een element moet zijn van de lijst met natuurlijke getallen van 0 tot en met 10. Bemerkt ook het dubbelpunt aan het einde van regel 2 die aangeeft dat wat ingesprongen volgt op lijn 3 moet worden herhaald voor alle mogelijke waarden van `i`.
- Lijnen 4 en 5 bevatten opnieuw een lus. De variabele `i` varieert opnieuw voor alle mogelijke natuurlijke waarden gelegen tussen 0 en 10. Voor elke waarde