Python Builtin Werkboek

D. Leeuw

4 juni 2025
1.0.0

1 Over dit Document

1.1 Leerdoelen

1. Het doel is om vertrouwd te raken met de built in functies van Python

1.2 Voorkennis

De lezer die deze opdrachten wil maken dient te weten

- hoe if/else/elif wordt gebruikt in Python
- hoe for/while wordt gebruikt in Python met gebruik van break en continue
- hoe builtin functies in Python worden aangesproken en waar er documentatie over gevonden kan worden

2 Decimaal naar binair

1. Maak gebruik van een built-in functie van Python om de volgende getallen om te zetten naar binair:
    - 654
    - 976
    - 64
    - 2048

3 Decimaal naar hexadecimaal

1. Maak gebruik van een built-in functie van Python om de volgende getallen om te zetten naar hexadecimaal:
    - 654
    - 976
    - 64
    - 2048

4 Conversie in een loop

1. Maak gebruik van een loop om de twee voorgaande opdrachten als één opdracht uit te voeren waarbij de uitvoer eruit moet zien als: `decimaalgetal - hexadeximaalgetal - binair getal`

5 Converteren naar decimaal

1. Maak een script waarin je de volgende binaire en hexadecimale getallen omzet naar decimaal. Maak gebruik van een built-in functie van Python.

   - 110000001111111111101110
   - 10111010010111101011101000010001
   - 11011110101011011011111011101111
   - 01011100101000011010101100011110

   - C0FFEE
   - BA5EBA11
   - DEADBEEF
   - 5CA1AB1E

Wat valt je op?

6 Caesar Cipher

Julius Caesar gebruikte encryptie om zijn orders door te geven aan zijn ondergeschikten. De reden dat hij encryptie gebruikte was om te zorgen dat zijn orders geheim beleven. Als de persoon die het bericht moest overbrengen gevangen werd genomen of werd gedood dan had de vijand alleen het ge-encrypte bericht en kende dus niet meteen de opdracht die uitgevoerd moest worden.

De techniek die Julius Caesar gebruikte was dat hij de letters van zijn bericht 1 of meer posities op schoof in het alphabet, dus een a werd dan bijvoorbeeld een d en een b werd dan een e. Door dit op een vaste manier te doen, konden de ontvangers het bericht relatief makkelijk decoderen, terwijl die vijand die niet wist hoevaak een letter verschoven was, als ze al wisten dat het om een verschuiving ging, niets met het bericht konden. In de tijd

dat Julius Caesar deze techniek gebruikte, ongeveer 58 voor Christus, waren er sowieso niet veel mensen die konden lezen.

Stel dat we een bericht hebben met de tekst: 'Dit is een geheim bericht' en we schuiven alle letters één plaats op in het alfabet dan wordt het geencrypte bericht 'Eju jt ffo hfjn cfsjdiu'. Dat is al behoorlijk onleesbaar. Als we ook nog de spaties verwijderen 'Ejujtffohfjncfsjdiu' dan is het totaal onleesbaar geworden.

Later werd deze techniek bekend onder de naam: Caesar Cipher

Extra informatie over het Caesar Cipher:

- https://www.splunk.com/en_us/blog/learn/caesar-cipher.html
- https://nl.wikipedia.org/wiki/Caesarcijfer

6.1 Opdracht

We hebben gezien dat de reeks nummers voor uppercase letters een aparte reeks is van de reeks met lowercase letters. Daar moeten we rekening mee houden. Ook als we bijvoorbeeld de letter z één plaats opschuiven dan moet het de a worden en niet { zoals dat volgens de ASCII-tabel zou worden. Om te weten te komen of een variabele uppercase of lowercase is kunnen we `islower` en `isupper` gebruiken. Voor het detecteren of een variabele een cijfer is is er `isnumeric`. Dat ziet er in Python code zo uit:

```
chars = [ 'a', 'A', '1', 'B', 'b', '2' ]
for char in chars:
    if char.islower():
        print(f"{char} is lowercase")
    elif char.isupper():
        print(f"{char} is uppercase")
    elif char.isnumeric():
        print(f"{char} is numeric")
```

Beschrijf in een flowdiagram hoe je de de verschuiving zou kunnen uitvoeren. Hou daarbij rekening met de volgende zaken:

- Gebruik een variabele voor het aantal keer dat er geschoven moet worden in het alfabet
- Geef deze variabele een waarde tussen 1 en 9 om de code niet te complex te maken
- Denk erom dat je na de Z weer bij A moet beginnen, na z weer bij a en na 9 komt weer 0.

- Leestekens veranderen niet.

Maak een script dat een substitutie uitvoert (encryptie). Een variable bevat het te encrypten bericht en de output van het script is het encrypte bericht.

6.2 Tip: ASCII-tabel

Computers zijn rekenaars, ze werken alleen met getallen. Mensen werken graag met tekst. Om een computer om te laten gaan met tekst moet tekst voor de computer omgezet worden in getallen. Als elk programma andere getallen gebruikt voor deze omzetting, dan wordt het een zooitje. We hebben een standaard vertaal tabel nodig om tekst om te zetten in getallen. Eén van deze tabellen is de ASCII-tabel. ASCII staat voor American Standard Code for Information Interchange en dat is precies wat het doet. Het is een standaard om informatie uitwisseling te vergemakkelijken.

	000	001	010	011	100	101	110	111
...0000	NUL	DLE	SPC	0	@	P	‘	p
...0001	SOH	DC1	!	1	A	Q	a	q
...0010	STX	DC2	"	2	B	R	b	r
...0011	ETX	DC3	#	3	C	S	c	s
...0100	EOT	DC4	$	4	D	T	d	t
...0101	ENQ	NAK	%	5	E	U	e	u
...0110	ACK	SYN	&	6	F	V	f	v
...0111	BEL	ETB	’	7	G	W	g	w
...1000	BS	CAN	(	8	H	X	h	x
...1001	HT	EM	)	9	I	Y	i	y
...1010	LF	SUB	*	:	J	Z	j	z
...1011	VT	ESC	+	;	K	⌊	k	{
...1100	FF	FS	,	<	L	\	l	|
...1101	CR	GS	-	=	M	⌋	m	}
...1110	SO	RS	.	>	N	^	n	~
...1111	SI	US	/	?	O	_	o	DEL

Om characters om te zetten naar een binaire code nemen we een character, zoeken deze op in te tabel. De kolom geeft de eerste 3 bits en de rij geeft de tweede 4 bits, zo komen we op 7-bits:

A 100 0001 = 65

a 110 0001 = 97

0 011 0000 = 48+0 = 48

9 011 1001 = 48+9 = 57

Het alfabet is op volgorde, dus B is 66 en b is 98 etc.

6.3 TiP: Python conversie functies

Python heeft builtin functies die nummers kunnen omzetten naar characters en characters naar nummers. Deze functies kan je vinden op o.a. de site van W3Schools: https://www.w3schools.com/python/python_ref_functions.asp of op de site van Python (https://docs.python.org/3/library/functions.html). Zoek de functie op die je nodig hebt om een character om te zetten naar een number en de functie die het omgekeerde doet. Test de functies in een script.

We hebben gezien dat de reeks nummers voor uppercase letters een aparte reeks is van de reeks met lowercase letters. Daar moeten we rekening mee houden. Ook als we bijvoorbeeld de letter z één plaats opschuiven dan moet het de a worden en niet { zoals dat volgens de ASCII-tabel zou worden. Om te weten te komen of een variabele uppercase of lowercase is kunnen we `islower` en `isupper` gebruiken. Voor het detecteren of een variabele een cijfer is is er `isnumeric`. Dat ziet er in Python code zo uit:

```
chars = [ 'a', 'A', '1', 'B', 'b', '2' ]
for char in chars:
    if char.islower():
        print(f"{char} is lowercase")
    elif char.isupper():
        print(f"{char} is uppercase")
    elif char.isnumeric():
        print(f"{char} is numeric")
```

Extra informatie over uppercase, lowercase en numeric detectie:

- https://www.w3schools.com/python/pythn_ref_string.asp
- https://docs.python.org/3/library/stdtypes.html#string-methods