

Typy danych w języku Python

Typy proste

- Logiczny (bool)
- Całkowity (int)
- Zmiennopozycyjny (float)
- Zespółony (complex)
- Napisowy (str)

Typy strukturalne

- Zbiór (set)
- Krotka (tuple)
- Lista (list)
- Słownik, tabela (dict)

1

Działania na typach prostych

- typ logiczny (bool)

True False
not or and

- typ całkowity (int)

12 -21

+ - * // % **
& ^ ~ >> <<

- typ zmiennopozycyjny (float)

12.3 2e-23

+ - * / **

- typ zespolony (complex)

3+4j 3.0+4.0j 2j
+ - * / **

2

Funkcje wbudowane

abs(n) - wartość bezwzględna

chr(n) - znak o kodzie n

ord(zn) - kod znaku

min(a,b,...) - najmniejsza z liczb

max(a,b,...) - największa z liczb

round(x) - zaokrąglenie wartości

len(s) - długość napisu, listy, krotki

type(x) - typ zmiennej

Wbudowane
biblioteczne
dostęp z pkgutil
zestawienie
math, random...
numpy
→ pip3 3.11

3

Biblioteki funkcji

from math import sqrt, ln, sin, cos

from math import *

import math

import math as m

cos = 0

niezalecane

math.sqrt(x)

m.sqrt(x)

Funkcja	Znaczenie
ceil(value)	rounds up
cos(value)	cosine, in radians
floor(value)	rounds down
log(value)	logarithm, base e
log10(value)	logarithm, base 10
max(value1, value2)	larger of two values
min(value1, value2)	smaller of two values
sin(value)	sine, in radians
sqrt(value)	square root

4

Kod ASCII

	0	1	2	3	4	5	6	7
0	NUL	DLE	space	0	@	P	.	p
1	SOH	DC1	!	1	(A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	\$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(8	H	X	h	x
9	HT	EM)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[k	{
C	FF	FS	,	<	L	\	l	
D	CR	GS	=	=	M]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	del

256

5

Rozszerzenie kodu ASCII

128	Ç	144	É	160	á	176	ð	192	À	208	À	224	à	240	à
129	ü	145	æ	161	í	177	ñ	193	Á	209	Á	225	á	241	á
130	é	146	æ	162	ó	178	ñ	194	Â	210	Â	226	â	242	â
131	á	147	ó	163	ú	179	ñ	195	Ã	211	Ã	227	ã	243	ã
132	ä	148	ö	164	û	180	ñ	196	Ä	212	Ä	228	ä	244	ä
133	å	149	ö	165	ü	181	ñ	197	Å	213	Å	229	å	245	å
134	ä	150	ü	166	å	182	ñ	198	Ä	214	Ä	230	ä	246	ä
135	ç	151	ù	167	ö	183	ñ	199	Å	215	Å	231	å	247	å
136	è	152	ý	168	ç	184	ñ	200	Ä	216	Ä	232	ä	248	ä
137	é	153	Ö	169	ç	185	ñ	201	Ä	217	Ä	233	ä	249	ä
138	é	154	Ü	170	ç	186	ñ	202	Ä	218	Ä	234	ä	250	ä
139	í	155	ö	171	ç	187	ñ	203	Ä	219	Ä	235	ä	251	ä
140	í	156	É	172	ç	188	ñ	204	Ä	220	Ä	236	ä	252	ä
141	í	157	ç	173	ç	189	ñ	205	Ä	221	Ä	237	ä	253	ä
142	Ä	158	ç	174	ç	190	ñ	206	Ä	222	Ä	238	ä	254	ä
143	Ä	159	ç	175	ç	191	ñ	207	Ä	223	Ä	239	ä	255	ä

ISO 8859-2 PN

Source: www.LookupTables.com

6

Unicode

UTF-8

Znaki ASCII kodujemy za pomocą 1 bajta.
Alfabet: łaciński, grecki, armeński, hebrajski, arabski, koptyjski i cyrylica
kodujemy za pomocą 2 bajtów.
Kolejne znaki (m.in. alfabet chiński i japoński) kodowanych jest na 3 i 4 bajtach.

00000000 – 0000007F: 0xxxxxxx
00000080 – 000007FF: 110xxxxx 10xxxxxx
00000800 – 0000FFFF: 1110xxxx 10xxxxxx 10xxxxxx
00010000 – 001FFFFF: 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx

UCS-2

Wszystkie znaki zapisywane są za pomocą 2 bajtów. Kodowanie to pozwala na zapisanie tylko 65536 początkowych znaków Unikuodu.

UCS-4

Wszystkie znaki zapisywane są za pomocą 4 bajtów.

Kodowanie „polskich” znaków

Znak	ISO 8859-2	Unicode	UTF-8
ą	161	261	196 133
ć	198	263	196 135
ę	202	281	196 153
ł	163	322	197 130
ń	209	324	197 132
ó	211	211	195 179
ś	166	347	197 155
ż	172	378	197 186
ź	175	380	197 188
Ą	177	260	196 132
Ć	230	262	196 134
Ę	234	280	196 152
Ł	179	321	197 129
Ń	241	323	197 131
Ó	243	243	195 147
Ś	182	346	197 154
Ź	188	377	197 185
Ż	191	379	197 187



7

8

Liczby całkowite

Java	C/C++	zakres	rozmiar
byte	char	-128..127	1
short	short int	-32768..32767	2
int	int	-2147483648..214748647	4
long	long int	-2^63..2^63-1	8

W języku Python 3.x zakres typu int jest ograniczony jedynie dostępną pamięcią.

Liczby zmiennopozycyjne

Java, C/C++

typ	zakres	dokładność	rozmiar
float	1.5E-45 .. 3.4E38	7-8	4
double	5.0E-324 .. 1.7E308	15-16	8

W języku Python typ float jest równoważny z typem double w języku C/C++.

9

10

Dokładność obliczeń

>>> 2.7+1.1
3.8000000000000003

>>> 0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1+0.1
x=0.9999999999999999

Obliczenia iteracyjne

```
r = 3.0
p = 0.01
for i in range(0,101):
    q = p+r*p*(1-p)
    print(q)
    p = q;
# end for
```

iter	float (64bit)	wartość dokładna
1	0.0397000000000000	0.0397000
2	0.1540717300000000	0.15407172999
3	0.545072626044421	0.54507262604442129999999999999999999999999999999999
4	1.288978001188801	1.28897800118880056414607744431892999999999999999999
5	0.171519142109176	0.171519142109175610913273761193669501531848
...		
50	1.313996746606757	1.314489760648214812383780183964860775887564
51	0.076224636147602	0.074309050045856049321771955401870933940599
52	0.287467959122905	0.280670695427271580509655379893949005186949
53	0.901958353924756	0.886354663894201577758577930930197101949547
54	1.167246799055058	1.188544884955797172985886514749765353448385
55	0.581591926507394	0.516262709159421280638670823473078496363871
56	1.311620199093830	1.265469282031809506189821162818058704556637
57	0.085438156362505	0.257639616828927735003403861501470932678094
...		
93	1.217632446396239	0.86478556843432384355119764618126915551368
94	0.422643462034285	1.215580049398713266568656814984364161700898
95	1.154691360136162	0.429415628106318510122842974986646350734709
96	0.618829029025349	1.164469167439441909702911183666688004618360
97	1.326468014608027	0.589911344006446644012487937540734261222550
98	0.027319877097624	1.315659194663309877447649200395210188456166
99	0.107040381336610	0.069759429146912168060579275901754241874755
100	0.393788595636378	0.264438582722939495435219626057398282867365

11

12

```
print('#' * 80)
```

- [
Iala I "Sunderore" 806
II do II I ala I ob- "exo" ---
III de III
IV do IV ala
] = [I ala I ala II
 I ala do ala II] priist (9)

13

[illegible]

```
>>> greet[0:3]
'Hel'
>>> greet[5:9]
'-Bob'
>>> greet[:5]
'Hello'
>>> greet[5:]
'-Bob'
>>> greet[:]
'Hello-Bob'
```

14

• Lower (A) – male Pterocera

- $\zeta = \text{"ole", "ole"}$ $\rightarrow \text{split}() \rightarrow [\text{"ole"}, \text{"", "ole"}]$
~~CSV~~ CSV

15