

# Chapter 3

## 數值資料型態

# 3 Python的資料型態


- 數值資料型態(numeric type)：整數(int)、浮點數(float)、複數(complex number)
  - 布林值(Boolean)資料型態
  - 文字序列型態(text sequence type)：字串(string)
  - 字元組(bytes，有的書稱字節)資料型態
- 序列型態(sequence type)：list、tuple
  - 對映型態(mapping type)：dict
  - 集合型態(set type)：set、frozenset

Container

## 3.1 獲取變數型態

- Python 是動態語言 → 宣告變數時不用宣告變數的型態。
- 如何知道某個變數的資料型態？
  - `type()`

```
1  x = 10
2  y = x / 3
3  print(x)
4  print(type(x))
5  print(y)
6  print(type(y))
```

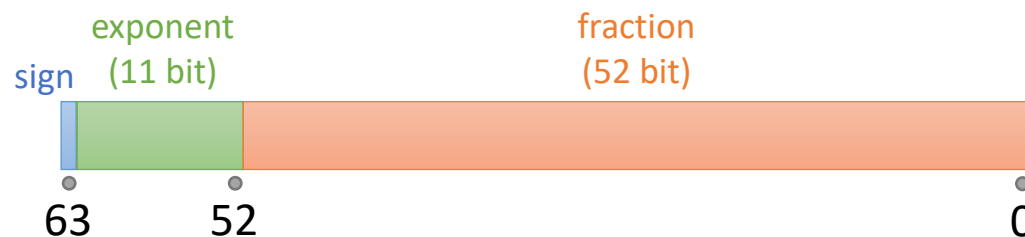


```
10
<class 'int'>
3.3333333333333335
<class 'float'>
```



## 3.2 數值資料型態

- 浮點數（書上內容有問題）
  - IEEE 754 規範，雙精度浮點數，15~17位數的精確度。



- 整數與浮點數的差異：
  - 宣告變數時不用指定資料型態 (動態語言)，所以當數值不含小數點就會被當成整數型態，數值包含小數點就會被當浮點數型態。

```
>>> x, y = 10, 10.0
>>> type(x)
<class 'int'>
>>> type(y)
<class 'float'>
```

## 3.2 數值資料型態

- 整數與浮點數的運算
  - 以資料範圍大的為主 ( float )
  - 計算時會先將整數轉換為浮點數再執行運算。

```
>>> x = 2 ** 55
>>> print(x)
36028797018963968
>>> print(type(x))
<class 'int'>
>>> y = 3602879701896397.0 / x
>>> print(y)
0.1
>>> print(type(y))
<class 'float'>
>>> print("%25.23f" % y)
0.100000000000000000000555112
```

```
>>> x = 2 ** 55
>>> print(x)
36028797018963968
>>> print(type(x))
<class 'int'>
>>> x = 3602879701896397.0 / x
>>> print(x)
0.1
>>> print(type(x))
<class 'float'>
```

## 3.2 數值資料型態

- 不同進位方式的整數
  - 我們常用的是 10 進位。例如：9527
  - 2 進位：0, 1。例如：**0b**10010100110111
  - 8 進位：0, 1, 2, 3, 4, 5, 6, 7。例如：**0o**22467
  - 16 進位：0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F（英文字母也可用小寫表示）。例如：**0x**2537

## 3.2 數值資料型態

- 2 進位整數與函數 bin()

```
1 x = 0b10010100110111    #2 進位數值
2 print(x)                 #印出 10 進位的值
3 y = 9527                  #10 進位數值
4 print(bin(y))             #印出 2 進位的值
```

```
9527
0b10010100110111
```

- 8 進位整數與函數 oct()

```
1 x = 0o22467               #8 進位數值
2 print(x)                  #印出 10 進位的值
3 y = 9527                  #10 進位數值
4 print(oct(y))             #印出 8 進位的值
```

```
9527
0o22467
```

- 16進位整數與函數 hex()

```
1 x = 0x2537                #16 進位數值
2 print(x)                  #印出 10 進位的值
3 y = 9527                  #10 進位數值
4 print(hex(y))             #印出 16 進位的值
```

```
9527
0x2537
```



## 3.2 數值資料型態

- 整數與浮點數的強制轉換
  - `int()`：將資料型態強制轉換成整數型態
  - `float()`：將資料型態強制轉換成浮點數型態

```
1 x = 10.5
2 print(x)
3 print(type(x))
4 y = int(x) + 5
5 print(y)
6 print(type(y))
```

```
10.5
<class 'float'>
15
<class 'int'>
```

```
1 x = 10
2 print(x)
3 print(type(x))
4 y = float(x) + 5
5 print(y)
6 print(type(y))
```

```
10
<class 'int'>
15.0
<class 'float'>
```

## 3.2 數值資料型態

- 數值運算常用的函數
  - $\text{abs}(x)$ ：計算  $x$  的絕對值。
  - $\text{pow}(x, y)$ ：計算  $x$  的  $y$  次方。
  - $\text{round}(x)$ ：浮點數  $x$  轉換成整數，採用 Bankers Rounding。四捨六入五成雙。
    - $\text{round}(2.4) = 2$
    - $\text{round}(2.5) = 2$
    - $\text{round}(2.6) = 3$
    - $\text{round}(1.5) = 2$
  - $\text{round}(x, y)$ ：浮點數  $x$  做取到小數點第  $y$  位，第  $y$  位後的數字 5 以下捨去，51以上進位。
    - $\text{round}(2.155, 2) = 2.15$
    - $\text{round}(2.1551, 2) = 2.16$

## 3.2 數值資料型態

- 科學記號表示法
  - 所有浮點數都可以表達成  $a * 10^n$ 。
    - 其中  $1 \leq a < 10$ ,  $n$  是整數。
    - 例如：123456 可表達成  $1.23456 * 10^5$ 。
  - 將 10 以 E 或是 e 表達，指數部份改為一般數字，並省略掉 "\*" 號。
    - 123456 以科學記號表達為 1.23456E+5 ( "+" 號可省略 )
    - 0.000123 以科學記號表達為 1.23E-4

```
1  x = 1e4
2  print(x)
3  y = 1.3e-4
4  print(y)
```

```
10000.0
0.00013
```

## 3.2 數值資料型態

- 複數 (Complex Number)  $a+bj$ 
  - 由實部和虛部組成。
  - 實部和虛部都是浮點數。
  - $j$  是虛部的單位。
  - 可透過 `real` 跟 `imag` 屬性獲得實部和虛部的值

```
1 x = 3 + 5j
2 y = complex(6, 8)
3 print(x)
4 print(y.real)
5 print(y.imag)
```

```
(3+5j)
6.0
8.0
```

## 3.3 布林值資料型態

- 布林值(Boolean)資料型態的值有兩種，**True**(真)或**False**(偽)。
- 資料型態代號為 `bool`。
- 常用在流程控制。
- 強制轉換為整數，或是將布林值做運算，則 `True` 的值為 `1`，`False` 的值為 `0`。

```
1  x = True
2  y = False
3  print(type(x))
4  print(int(x))
5  print(int(y))
6  x_int = 1 + x
7  print(x_int)
8  print(type(x_int))
```

```
<class 'bool'>
1
0
2
<class 'int'>
```

## 3.3 布林值資料型態

- bool() 函數

- 將資料轉換為 True 或 False。

- 輸出為False，當

- 布林值False

- 整數0

- 浮點數0.0

- 空字串 "" or ""

- 空串列 []

- 空元組 ()

- 空字典 {}

- 空集合 set()

- None

- 其他狀況都會輸出 True。

```
1 x = None
2 print(type(x))
```

<class 'NoneType'>

## 3.4 字串資料型態

- 字串資料是兩個單引號(')之間或是兩個雙引號(")之間任意個數字元符號的資料。
- 資料型態代號為str。

```
1 x = "The Python Class."  
2 y = 'Today is Tuesday.'  
3 print(x)  
4 print(type(x))  
5 print(y)  
6 print(type(y))
```

```
The Python Class.  
<class 'str'>  
Today is Tuesday.  
<class 'str'>
```

## 3.4 字串資料型態

- 透過運算子 "+", 可以將兩個字串做連接, 產生新的字串。

```
1  num_x = 123
2  num_y = 456
3  num_z = num_x + num_y
4  print("數字相加為")
5  print(num_z)
6  str_x = "123"
7  str_y = "456"
8  str_z = str_x + str_y
9  print("字串相加為")
10 print(str_z)
11 str_z1 = str_x + " " + str_y
12 print(str_z1)
```

```
數字相加為
579
字串相加為
123456
123 456
```



## 3.4 字串資料型態

- 如果字串長度多於一行，有下列幾種方式做設定：

```
1  str1 = '''This is string
2  number 1'''
3  print(str1)      #用三個單引號形成跨行字串，但易與註解混搖
4  str2 = '''This is string \
5  number 2'''
6  print(str2)      #用三個單引號加上"\ "
7  str3 = "This is string " \
8  "number 3"
9  print(str3)
10 str4 = ("This is string "
11 "number 4")
12 print(str4)
```

```
This is string
number 1
This is string number 2
This is string number 3
This is string number 4
```

## 3.4 字串資料型態

- 逸出字元 ( Escape Character )
  - 若字串內有特殊字元，例如：單引號、雙引號...等，必須在此特殊字元前面加上\"(反斜線)，字串才會正常。這種含有\"的字元稱作逸出字元。

逸出字元	Hex 值	意義	逸出字元	Hex 值	意義
\'	27	單引號	\n	0A	換行
\"	22	雙引號	\o		8 進位表示
\\	5C	反斜線	\r	0D	游標移至最左位置
\a	07	響鈴	\x		16 進位表示
\b	08	BackSpace 鍵	\t	09	Tab 鍵效果
\f	0C	換頁	\v	0B	垂直定位

## 3.4 字串資料型態

- 逸出字元 ( Escape Character )
  - 若字串內有特殊字元，例如：單引號、雙引號...等，必須在此特殊字元前面加上\"(反斜線)，字串才會正常。這種含有\"的字元稱作逸出字元。

```
1 str1 = 'This is Mr. Wu's Class'
2 print(str1)
```

SyntaxError: invalid syntax

```
1 str1 = 'This is Mr. Wu\'s Class'
2 print(str1)
3 str2 = "This is Mr. Wu's Class"
4 print(str2)
5 str3 = "This is \tMr. Wu's\n Class"
6 print(str3)
7 str4 = "The keyword \"Python\"."
8 print(str4)
```

```
This is Mr. Wu's Class
This is Mr. Wu's Class
This is      Mr. Wu's
Class
The keyword "Python".
```

## 3.4 字串資料型態

- `str()` 函數
  - 設定空字串
  - 設定字串
  - 將數值資料轉換為字串

```
1 num1 = 123
2 num2 = 456
3 num3 = num1 + num2
4 print("數字相加")
5 print(num3)
6 str1 = str(num1) + str(num2)
7 print("將數字轉換成字串再相加")
8 print(str1)
```

```
數字相加
579
將數字轉換成字串再相加
123456
```

```
1 str1 = str("ABC")
2 print(str1)
3 str2 = str("")
4 print(str2)
5 num_int = 12345
6 str3 = str(num_int)
7 print(str3)
8 num_float = 10.1234
9 str4 = str(num_float)
10 print(str4)
```

```
ABC
12345
10.1234
```

## 3.4 字串資料型態

- 將字串轉為整數做運算

```
1 str1 = "123"  
2 str2 = "456"  
3 str3 = str1 + str2  
4 print(str3)  
5 num3 = int(str1) + int(str2)  
6 print(num3)  
7 str4 = "12a"  
8 num4 = int(str4)  
9 print(num4)
```

```
123456  
579  
Traceback (most recent call last):  
  File "e:\PythonTest\test.py", line 8, in <module>  
    num4 = int(str4)  
ValueError: invalid literal for int() with base 10: '12a'
```

- 字串與整數相乘產生字串複製效果

```
1 str1 = "123"  
2 str2 = "456"  
3 str3 = str1 * 3 + str2 * 2  
4 print(str3)
```

```
123123123456456
```

## 3.4 字串資料型態

- 使用逸出字元做字串輸出的排版。

```
1  str1 = "Topic 1: Variables\n"  
2  str1_1 = "\t1. int\n\t2. float\n"  
3  str1_2 = "\t3. string.\n\n"  
4  str2 = "Topic 2: Loop\n"  
5  str2_1 = "\t1. For Loop.\n\t2. While Loop\n"  
6  str_all = str1 + str1_1 + str1_2 + str2 + str2_1  
7  print(str_all)
```

Topic 1: Variables

1. int
2. float
3. string.

Topic 2: Loop

1. For Loop.
2. While Loop

- 字串前加上r，可防止逸出字元發生效果。

```
1  str1 = r"Topic 1: Variables\n"  
2  str1_1 = r"\t1. int\n\t2. float\n"  
3  str1_2 = r"\t3. string.\n\n"  
4  str_all = str1 + str1_1 + str1_2  
5  print(str_all)
```

```
Topic 1: Variables\n\t1. int\n\t2. float\n\t3. string.\n\n
```

# 3.5 字串與字元

## • ASCII Code

Ctrl	Dec	Hex	Char	Code	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
^@	0	00		NUL	32	20		64	40	@	96	60	'
^A	1	01		SOH	33	21	!	65	41	A	97	61	a
^B	2	02		STX	34	22	..	66	42	B	98	62	b
^C	3	03		ETX	35	23	#	67	43	C	99	63	c
^D	4	04		EOT	36	24	\$	68	44	D	100	64	d
^E	5	05		ENQ	37	25	%	69	45	E	101	65	e
^F	6	06		ACK	38	26	&	70	46	F	102	66	f
^G	7	07		BEL	39	27	,	71	47	G	103	67	g
^H	8	08		BS	40	28	(	72	48	H	104	68	h
^I	9	09		HT	41	29	)	73	49	I	105	69	i
^J	10	0A		LF	42	2A	*	74	4A	J	106	6A	j
^K	11	0B		VT	43	2B	+	75	4B	K	107	6B	k
^L	12	0C		FF	44	2C	,	76	4C	L	108	6C	l
^M	13	0D		CR	45	2D	-	77	4D	M	109	6D	m
^N	14	0E		SO	46	2E	.	78	4E	N	110	6E	n
^O	15	0F		SI	47	2F	/	79	4F	O	111	6F	o
^P	16	10		DLE	48	30	0	80	50	P	112	70	p
^Q	17	11		DC1	49	31	1	81	51	Q	113	71	q
^R	18	12		DC2	50	32	2	82	52	R	114	72	r
^S	19	13		DC3	51	33	3	83	53	S	115	73	s
^T	20	14		DC4	52	34	4	84	54	T	116	74	t
^U	21	15		NAK	53	35	5	85	55	U	117	75	u
^V	22	16		SYN	54	36	6	86	56	V	118	76	v
^W	23	17		ETB	55	37	7	87	57	W	119	77	w
^X	24	18		CAN	56	38	8	88	58	X	120	78	x
^Y	25	19		EM	57	39	9	89	59	Y	121	79	y
^Z	26	1A		SUB	58	3A	:	90	5A	Z	122	7A	z
^[	27	1B		ESC	59	3B	;	91	5B	[	123	7B	{
^\	28	1C		FS	60	3C	<	92	5C	\	124	7C	
^]	29	1D		GS	61	3D	=	93	5D	]	125	7D	}
^^	30	1E	▲	RS	62	3E	>	94	5E	^	126	7E	~
^-	31	1F	▼	US	63	3F	?	95	5F	_	127	7F	~*

## IBM的擴展ASCII碼表 (Extended ASCII)

128	Ç	144	É	160	á	176	☐	193	±	209	™	225	ß	241	±
129	ü	145	æ	161	í	177	☐	194	ˆ	210	™	226	Γ	242	≥
130	é	146	Æ	162	ó	178	☐	195	ˆ	211	™	227	π	243	≤
131	â	147	ô	163	ú	179		196	—	212	™	228	Σ	244	∫
132	ä	148	ö	164	ñ	180	†	197	†	213	™	229	σ	245	∫
133	à	149	ò	165	Ñ	181	‡	198	†	214	™	230	μ	246	÷
134	â	150	û	166	ª	182	‡	199	‡	215	™	231	τ	247	≈
135	ç	151	ù	167	º	183	™	200	™	216	™	232	Φ	248	°
136	ê	152	—	168	¿	184	™	201	™	217	™	233	Θ	249	.
137	ë	153	Ö	169	—	185	‡	202	™	218	™	234	Ω	250	.
138	è	154	Ü	170	¬	186	‡	203	™	219	■	235	δ	251	√
139	í	156	£	171	½	187	™	204	‡	220	■	236	∞	252	—
140	î	157	¥	172	¾	188	™	205	=	221	■	237	φ	253	²
141	ï	158	—	173	¡	189	™	206	‡	222	■	238	ε	254	■
142	Ä	159	f	174	«	190	‡	207	±	223	■	239	∩	255	
143	Å	192	L	175	»	191	™	208	™	224	α	240	≡		

<http://kevin.hwai.edu.tw/~kevin/material/JAVA/Sample2016/ASCII.htm>

\* ASCII 碼 127 具有代碼 DEL。在 MS-DOS 下，這個代碼與 ASCII 8 (BS) 的效果相同。DEL 代碼可以由 CTRL + BKSP 鍵產生。

## 3.5 字串與字元

- chr() 函數
  - Python 裡面沒有所謂的字元(character)資料，如果字串含一個字元，我們稱這是含一個字元的字串。
  - 透過 chr() 取得一個字元的字串的ASCII 碼。

```
>>> x=97
>>> print(chr(x))
a
>>> x -= 32
>>> print(chr(x))
A
```



## 3.5 字串與字元

### • Unicode

- 官方中文名稱為統一碼，也譯名為萬國碼、國際碼、單一碼，是電腦科學領域的業界標準。它整理、編碼了世界上大部分的文字系統，使得電腦可以用更為簡單的方式來呈現和處理文字。
- <http://www.unicode.org/charts>，其中East Asian Scripts欄位可以看到CJK，這是Chinese、Japanese與Korean的縮寫。
- 用16 bits 定義文字，可定義65536個字元。定義方式是以“\u”開頭後面有4個16進位的數字，所以是從“\u0000”至“\uFFFF”之間。
- 漢字編碼是在4E00~9FBB之間（共20923 個字）。

#### CJK Unified Ideographs (Han) (35MB)

CJK Extension A (6MB)  
 CJK Extension B (40MB)  
 CJK Extension C (3MB)  
 CJK Extension D  
 CJK Extension E (3.5MB)  
 CJK Extension F (4MB)  
 CJK Extension G (2MB)  
 (see also Unihan Database)

#### CJK Compatibility Ideographs

CJK Compatibility Ideographs Supplement

#### CJK Radicals / Kangxi Radicals

CJK Radicals Supplement  
 CJK Strokes  
 Ideographic Description Characters

## 3.5 字串與字元

- ord() 函數
  - 回傳字元的 Unicode 。

```
1  x1 = 97
2  x2 = chr(x1)
3  print(x2)
4  x3 = ord(x2)
5  print(x3)
6  x4 = "成"
7  print(hex(ord(x4)))
```

```
a
97
0x6210
```

## 3.5 字串與字元

- utf-8
  - 針對 Unicode字集的可變長度編碼方式。
  - 用一至四個位元組對Unicode字元集中的所有有效編碼點進行編碼，屬於Unicode標準的一部分。
  - 逐漸成為電子郵件、網頁及其他儲存或傳送文字優先採用的編碼方式。
  - ASCII 用 utf-8 編碼
    - ASCII字元只需一個位元組編碼。

## 3.5 字串與字元

- utf-8
  - 中文的 utf-8 編碼規則
    - 用 3 bytes ( 24 bits ) 編碼。
    - 第一個 byte 的最高 4 bits 為 1110，第二跟第三個 byte 的最高 2 bits 為 10。
    - 其餘 16 bits (24 – 4 – 2 – 2) 填入原本的 Unicode。

BYTE	2								1								0							
bit	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Utf-8 中文編碼規則	1	1	1	0	x	x	x	x	1	0	x	x	x	x	x	x	1	0	x	x	x	x	x	x
成的Unicode	x	x	x	x	0	1	1	0	x	x	0	0	1	0	0	0	x	x	0	1	0	0	0	0
成的utf-8編碼	1	1	1	0	0	1	1	0	1	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0

成 Unicode = 0x6210 = 0b0110\_0010\_0001\_0000

成 utf-8 = 0xE68890

## 3.6 bytes 資料

- 資料傳遞都是透過 bytes 資料，要從 bytes 資料透過正確的編碼喘換成字串，否則會有亂碼產生。
- Bytes資料格式是在字串前加上b，例如：用utf-8編碼"成"的 bytes資料為b'\xe6\x88\x90'。
- 若是英文字串的 bytes 資料格式，相對單純會顯示原始的字元，例如用ASCII、Unicode、utf-8 編碼的 "ABC"對應的 bytes 資料都是 b'ABC'。

## 3.6 bytes 資料

- 字串轉成 bytes 資料
  - 透過編碼(encode)將字串編碼為bytes 資料，使用 encode() 函數。

編碼	說明
'ascii'	標準的 ASCII 編碼。
'utf-8'	Unicode 可變長度編碼，也是最常用的編碼方式。
'cp-1252'	一般英文 Windows 作業系統編碼。
'cp950'	一般中文 Windows 作業系統編碼。
'unicode-escape'	Unicode 的常數格式 \uxxxx 或是 \Uxxxxxxxx

```

1 name = "abc"
2 print(len(name))
3 nameBytes = name.encode("utf-8")
4 print(len(nameBytes))
5 print(type(nameBytes))
6 print(nameBytes)
    
```

```

3
3
<class 'bytes'>
b'abc'
    
```

```

1 name = "成功大學"
2 print(len(name))
3 nameBytes = name.encode("utf-8")
4 print(len(nameBytes))
5 print(type(nameBytes))
6 print(nameBytes)
    
```

```

4
12
<class 'bytes'>
b'\xe6\x88\x90\xe5\x8a\x9f\xe5\xa4\xa7\xe5\xad\xb8'
    
```

## 3.6 bytes 資料

- bytes 資料轉成字串
  - 透過解碼(decode)將 bytes 資料解碼為字串，使用 decode() 函數。

```
1 name = "abc"
2 print(len(name))
3 nameBytes = name.encode("utf-8")
4 print(len(nameBytes))
5 print(type(nameBytes))
6 print(nameBytes)
7 nameUnicode = nameBytes.decode("utf-8")
8 print(len(nameUnicode))
9 print(nameUnicode)
```

```
3
3
<class 'bytes'>
b'abc'
3
abc
```

```
1 name = "成功大學"
2 print(len(name))
3 nameBytes = name.encode("utf-8")
4 print(len(nameBytes))
5 print(type(nameBytes))
6 print(nameBytes)
7 nameUnicode = nameBytes.decode("utf-8")
8 print(len(nameUnicode))
9 print(nameUnicode)
```

```
4
12
<class 'bytes'>
b'\xe6\x88\x90\xe5\x8a\x9f\xe5\xa4\xa7\xe5\xad\xb8'
4
成功大學
```

## 3.6 bytes 資料

- bytes 資料轉成字串
  - 透過解碼(decode)將 bytes 資料解碼為字串，使用 decode() 函數。

```
1 name = "abc"
2 print(len(name))
3 nameBytes = name.encode("utf-8")
4 print(len(nameBytes))
5 print(type(nameBytes))
6 print(nameBytes)
7 nameUCode = nameBytes.decode("unicode-escape")
8 print(len(nameUCode))
9 print(nameUCode)
```

```
3
3
<class 'bytes'>
b'abc'
3
abc
```

```
1 name = "成功大學"
2 print(len(name))
3 nameBytes = name.encode("utf-8")
4 print(len(nameBytes))
5 print(type(nameBytes))
6 print(nameBytes)
7 nameUCode = nameBytes.decode("unicode-escape")
8 print(len(nameUCode))
9 print(nameUCode)
```

```
4
12
<class 'bytes'>
b'\xe6\x88\x90\xe5\x8a\x9f\xe5\xa4\xa7\xe5\xad\xb8'
12
æøå .
```



## 3.7 動手練習

- `divmod()` 函數
  - 商, 餘數 = `divmod(被除數, 除數)`
- 高鐵以固定時速 250公里從台北到高雄行經380公里，請問需時幾小時幾分？
- 直角三角形兩邊長度各為 6 與 10，請為斜邊長度為多少？
- 兩點座標為(1, 8) 與 (3, 10)，請問兩點距離遠？