

# 以下的程式為數字系統的範例

## 參考資料

### 鳥哥私房菜 - 第零章、計算機概論

網址：[https://linux.vbird.org/linux\\_basic/centos7/0105computers.php](https://linux.vbird.org/linux_basic/centos7/0105computers.php)

鳥哥私房菜之 "第零章、計算機概論" 介紹很多電腦相關的知識，極力推薦大家仔細閱讀。



## 資料表示方式

我們目前使用的電腦只能認識 0 與 1，所有的數位化資料不論是：Word文件、PPT簡報檔、數位相機的照片、聲音、影片，在電腦都是以 0 和 1 表示，所以電腦常用的數位化資料是以二進位表示的。

**一個二進位元 (binary digit, 簡稱 bit, 位元) 可以表示 0 或 1，電腦儲存資料時的最小單元是 byte (位元組)，一個 byte 由 8 個 bit 所組成。**

**1 byte = 8 bits**

二進位就是逢二進位，由 1~8 個 bit 所組成的記憶體以二進位表示，可以表示的數值範圍如下圖所示：

	1-bit	2-bit	3-bit	4-bit
1	0	00	000	0000
2	1	01	001	0001
3		10	010	0010
4		11	011	0011
5			100	0100
6			101	0101
7			110	0110
8			111	0111
9				1000
10				1001
11				1010
12				1011
13				1100
14				1101
15				1110
16				1111

Number of bits	Minimum	Maximum	Number of combinations
5	00000	11111	32
6	000000	111111	64
7	0000000	1111111	128
8	00000000	11111111	256

Bytes	Bits	Number of combinations
1	8	256
2	16	65,536
3	24	16,777,216
4	32	4,294,967,296
5	40	1,099,511,627,776
6	48	281,474,976,710,656
7	56	72,057,594,037,927,900
8	64	18,446,744,073,709,600,000

由表格可知， $n$  個 bit 由 0 和 1 構成的組合有  $2^n$  種，上面的表格由 Excel 產生，其精度有限制，以下程式列出 1~8 bytes 可以表示的組合數

```
In [5]: from math import pow

for i in range(1,9):
    num_bits = i * 8
    print('bytes: {}, bits: {:>2}, pow(2, {:>2}): {:>30,'.format(i, num_bits, num_bits, int(po

bytes: 1, bits: 8, pow(2, 8): 256
bytes: 2, bits: 16, pow(2, 16): 65,536
bytes: 3, bits: 24, pow(2, 24): 16,777,216
bytes: 4, bits: 32, pow(2, 32): 4,294,967,296
bytes: 5, bits: 40, pow(2, 40): 1,099,511,627,776
bytes: 6, bits: 48, pow(2, 48): 281,474,976,710,656
bytes: 7, bits: 56, pow(2, 56): 72,057,594,037,927,936
bytes: 8, bits: 64, pow(2, 64): 18,446,744,073,709,551,616
```

## 數字系統

參考資料：二、八、十與十六進位 (數字系統) 轉換教學

<https://www.footmark.com.tw/news/introduction-to-computer/digital-system-conversion/>

此網頁中有很詳細的介紹數字系統，和轉換的方式，建議詳讀

## + 概念

### 二進位 (Binary , bin)

- 基數為 2 的系統：逢 2 進位。
- 數字符號：0, 1。

### 八進位 (Octal , oct)

- 基數 8 的系統：逢 8 進位。
- 數字符號：0, 1, 2, 3, 4, 5, 6, 7。

### 十進位 (Decimal , dec)

- 基數 10 的系統：逢 10 進位。
- 數字符號：0, 1, 2, 3, 4, 5, 6, 7, 8, 9。

### 十六進位 (Hexadecimal , hex)

- 基數 16 的系統：逢 16 進位。
- 數字符號：0, 1, 2, 3, 4, 5, 6, 7, 8, 9 與 10 → A, 11 → B, 12 → C, 13 → D, 14 → E, 15 → F。

## + 對照表

十進位	二進位	八進位	十六進位
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

- 二進位(binary)：逢2進位，數字符號：0, 1。
- 八進位(octal)：逢8進位，數字符號：0, 1, 2, 3, 4, 5, 6, 7。
- 十進位(decimal)：逢10進位，數字符號：0, 1, 2, 3, 4, 5, 6, 7, 8, 9。

- 十六進位(hexadecimal)：逢16進位，數字符號：0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 → A, 11 → B, 12 → C, 13 → D, 14 → E, 15 → F。

## 二進位和十進位的互相轉換

我們平常使用的數字系統使用十進位，電腦則使用二進位，兩者之間的轉換可參考 [鳥哥私房菜 - 第零章、計算機概論](#) 的文章如下：

**0.3.1 數字系統**

早期的電腦使用的是利用通電與否的特性的真空管，如果通電就是1，沒有通電就是0，後來沿用至今，我們稱這種只有0/1的環境為二進位制，英文稱為binary的哩。所謂的十進位指的是逢十進一位，因此在個位數歸為零而十位數寫成1。所以所謂的二進位，就是逢二就前進一位的意思。

那二進位怎麼用呢？我們先以十進位來解釋好了。如果以十進位來說，3456的意義為：

$$3456 = 3 \times 10^3 + 4 \times 10^2 + 5 \times 10^1 + 6 \times 10^0$$

特別注意：『任何數值的零次方為1』所以 $10^0$ 的結果就是1囉。同樣的，將這個原理帶入二進位的環境中，我們來解釋一下1101010的數值轉為十進位的話，結果如下：

$$1101010 = 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$$

$$= 64 + 32 + 0 \times 16 + 8 + 0 \times 4 + 2 + 0 \times 1 = 106$$

這樣你瞭解二進位的意義了嗎？二進位是電腦基礎中的基礎哩！瞭解了二進位後，八進位、十六進位就依此類推啦！那麼知道二進位轉成十進位後，那如果有十進位數值轉為二進位的環境時，該如何計算？剛剛是乘法，現在則是除法就對了！我們同樣的使用十進位的106轉成二進位來測試一下好了：

**圖0.3.1、十進位轉二進位的方法**

最後的寫法就如同上面的紅色箭頭，由最後的數字向上寫，因此可得到1101010的數字囉！這些數字的轉換系統是非常重要的，因為電腦的加減乘除都是使用這些機制來處理的！有興趣的朋友可以再參考一下其他計算機概論的書籍中，關於1的補數/2的補數等運算方式喔！

**0.3.2 文字編碼系統**

## 以下程式將十進位轉換為二進位 (以字串表示)

```
In [1]: decimal_number = 106

n = decimal_number
s = ""
while n >= 2:
    remainder = n % 2

    s = str(remainder) + s

    n = n // 2

s = str(n) + s
print('decimal: {}, binary: {}'.format(decimal_number, s))
```

decimal: 106, binary: 1101010

```
In [2]: decimal_number = 10

n = decimal_number
s = ""
while n >= 2:
    remainder = n % 2

    s = str(remainder) + s

    n = n // 2

s = str(n) + s
```

```
print('decimal: {}, binary: {}'.format(decimal_number, s))
```

```
decimal: 10, binary: 1010
```

In [3]:

```
decimal_number = 5

n = decimal_number
s = ""
while n >= 2:
    remainder = n % 2

    s = str(remainder) + s

    n = n // 2

s = str(n) + s
print('decimal: {}, binary: {}'.format(decimal_number, s))
```

```
decimal: 5, binary: 101
```

上面的範例是測試將 **10** 進位轉換為 **2** 進位的表示法，分別用 **106, 10, 5** 進行測試

以下程式將轉換的步驟寫成 **function**，就可以重複利用程式碼

In [5]:

```
def decimal_2_binary(n):
    s = ""
    while n >= 2:
        remainder = n % 2

        s = str(remainder) + s

        n = n // 2

    s = str(n) + s
    return s

for i in range(3):
    dec = int(input('please enter a decimal integer:'))
    result = decimal_2_binary(dec)
    print('decimal: {}, binary: {}'.format(dec, result))
```

```
please enter a decimal integer:106
decimal: 106, binary: 1101010
please enter a decimal integer:10
decimal: 10, binary: 1010
please enter a decimal integer:5
decimal: 5, binary: 101
```

以下程式將二進位 (以字串表示) 轉換為十進位

In [13]:

```
bits = '1101010'

num_bits = len(bits)

decimal = 0
s = bits + ' = '
s2 = 'decimal = '

for i in range(num_bits):
    power = num_bits - i - 1

    bit = bits[i]
```

```

s = s + ' {}x2**{}'.format(bit, power)

n = int(bit) * 2 ** power

s2 = s2 + ' {}'.format(n)

decimal = decimal + n

if i < num_bits - 1:
    s = s + ' +'
    s2 = s2 + ' +'

print('i = {}, s = {}'.format(i,s))

print('result: ', s)
print('{} = {}'.format(s2, decimal))

i = 0, s = 1101010 = 1x2**6 +
i = 1, s = 1101010 = 1x2**6 + 1x2**5 +
i = 2, s = 1101010 = 1x2**6 + 1x2**5 + 0x2**4 +
i = 3, s = 1101010 = 1x2**6 + 1x2**5 + 0x2**4 + 1x2**3 +
i = 4, s = 1101010 = 1x2**6 + 1x2**5 + 0x2**4 + 1x2**3 + 0x2**2 +
i = 5, s = 1101010 = 1x2**6 + 1x2**5 + 0x2**4 + 1x2**3 + 0x2**2 + 1x2**1 +
i = 6, s = 1101010 = 1x2**6 + 1x2**5 + 0x2**4 + 1x2**3 + 0x2**2 + 1x2**1 + 0x2**0
result:  1101010 = 1x2**6 + 1x2**5 + 0x2**4 + 1x2**3 + 0x2**2 + 1x2**1 + 0x2**0
decimal = 64 + 32 + 0 + 8 + 0 + 2 + 0 = 106

```

八進位(**octal**)就是逢八進位，可以將二進位的**bits**每3個一組，以八進位表示

十六進位(**hexadecimal**)就是逢十六進位，可以將二進位的**bits**每4個一組，以十六進位表示

數字系統	數值	說明
10進位	215	
2進位	11010111	將10進位轉換為2進位
8進位	11 010 111 <b>3 2 7</b>	將2進位的 bits 每3個一組，以八進位表示
16進位	1101 0111 <b>d 7</b>	將2進位的 bits 每4個一組，以十六進位表示

以下程式示範 2進位(**binary**)、8進位(**octal**)、16進位(**hexadecimal**) 的表示法

In [22]: `bin(215)` # 將數值以 2 進位的字串表示

Out[22]: `'0b11010111'`

In [21]: `oct(215)` # 將數值以 8 進位的字串表示

Out[21]: `'0o327'`

In [20]: `hex(215)` # 將數值以 16 進位的字串表示

Out[20]: '0xd7'

在 Python 中表示2進位、8進位、16進位的方式

2進位：**0b**11010111

8進位：**0o**327

16進位：**0x**d7

In [23]:

```
print(215)           # decimal
print(0b11010111)    # binary
print(0o327)          # octal
print(0xd7)           # hexadecimal
```

```
215
215
215
215
```

## 練習1

閱讀 **鳥哥私房菜 - 第零章、計算機概論** 關於數字系統的文章，自行找一個10進位的整數，轉換為 binary, octal, hexadecimal，將計算的過程用 PowerPoint 或 Word 呈現然後截圖，或是用手寫的方式把演算過程寫出來，然後拍照，將影像插入筆記本中。

## 練習2

撰寫一個Python程式測試數字系統的轉換

1. 設計一個function名為 **number\_system**，可以接受一個10進位整數的argument，然後印出這個整數的對應的 2進位、8進位、16進位表示方式 (可參考上面的程式範例)，function沒有傳回值 (return value)。
2. 利用 for loop 測試 **number\_system()** 函式，在迴圈裡面接受使用者輸入測試的資料，然後呼叫 **number\_system()** 函式測試其功能，迴圈執行共3次。

In [ ]: