數字系統

D

Here are only numbers ratified.
—William Shakespeare

學習目標

在本章中，你將學到：

- 瞭解基數、位數值、以及符號值等基本數字系統的觀念。
- 瞭解如何操作二進位制、八進位制、跟十六進位制的數字。
- 將二進位制數值縮轉爲八進位制數值或十六進位制數值。
- 將八進位制數值以及十六進位制數值轉成二進位制數值。
- 來回轉換十進位制與二進位制、八進位制、以及十六進位制的數值。
- 瞭解二進位制的算術運算以及如何使用二補數法表示二進位負數。

本章綱要

摘要｜術語｜自我測驗｜自我測驗解答｜習題

D.1 簡介

在本附錄裡，我們介紹 C++ 程式設計師會用到的主要數字系統，特別是當這些程式設計師所從事的軟體專案與機器等級的硬體有密切互動時。像這樣的專案包括作業系統、電腦網路軟體、編譯器、資料庫系統，以及需要高執行效能的應用程式。

當我們在 C++程式裡寫了一個像是 227 或-63 的整數時，這個數是使用**十進位制 [基數為 10，decimal (base 10) number system]**。十進位制裡的**數字 (digit)** 有 0、1、2、3、4、5、6、7、8、與 9。最小的數字是 0，最大的數字是 9 (比基數 10 小 1)。至於電腦內部則是使用**二進位制 [基數為 2，binary number system(base 2)]**。二進位制只有兩個數字，即 0 與 1。它的最小數字是 0，最大數字是 1 (比基數 2 小 1)。

我們將會看到，二進位數比等值的十進位數要長得多。工作上會使用到各種組合語言以及諸如 C++等高階語言向下觸及到機器層級的程式設計師都知道，以二進位數值工作眞的很累贅。因此，另外兩種數字系統：**八進位數字系統 [基數為 8，octal number system (base 8)]** 跟**十六進位數字系統 [基數為 16，hexadecimal number system (base 16)]**，深爲大眾所愛用，主要是因爲這兩種數字系統在縮轉二進位數值上十分便利。

在八進位制裡，數字範圍爲 0 到 7。因爲二進位制與八進位制所擁有的數字都比十進位制的數字要少，所以只需從十進位制裡取相對應的數字來用即可。

由於十六進位制需要有 16 個數字，所以會發生一種困難：最低數字是 0 但最高數字 (比基數 16 小 1) 等同於十進位制的 15。根據協定，我們使用字母 A 到字母 F 表示十六進位的數字，這些字母 A 到字母 F 所表示的十六進位數字分別對應十進位數字的 10 到 15。因此在十六進位制裡，我們可以看到有些值就如同十進位制一樣只用數字即可表示，例如 876；也有同時用到數字跟字母才能表示的值，像是 8A55F；還有僅以字母組成的數值，像是 FFE。偶爾，十六進位數會拼出一個常見的字詞，例如 FACE 或 FEED，

這對那些慣於處理數字的程式設計師而言，會感到有點奇怪。二進位制、八進位制、十進位制、跟十六進位制所含的數字概述於圖 D.1–D.2。

這些數字系統都使用**位數標記法 (positional notation)**，各個寫有數字的位數都有其獨特的**位數值 (positional value)**。舉例來說，在十進位數 937 中 (9、3、7 稱爲符號值)，我們稱 7 是寫在個位數 (ones position) 上，3 是寫在十位數 (tens position) 上，而 9 是寫在百位數 (hundreds position) 上。請注意，這些位數每個都是基數 (10) 的次方，這些次方從最右邊的 0 次方，隨著數值位數每次左移一位，次方數就遞增 1 (請參閱圖 D.3)。

二進位數字	八進位數字	十進位數字	十六進位數字
0	0	0	0
1	1	1	1
	2	2	2
	3	3	3
	4	4	4
	5	5	5
	6	6	6
	7	7	7
		8	8
		9	9
			A(十進位數的 10)
			B(十進位數的 11)
			C(十進位數的 12)
			D(十進位數的 13)
			E(十進位數的 14)
			F(十進位數的 15)

圖 D.1　二進位制、八進位制、十進位制及十六進位制的數字系統

屬性	二進位	八進位	十進位	十六進位
基數	2	8	10	16
最小數字	0	0	0	0
最大數字	1	7	9	F

圖 D.2　　二進位制、八進位制、十進位制及十六進位制的比較

十進位制內的位數值。			
十進位數字	9	3	7
位數名	百位數	十位數	個位數
位數值	100	10	1
以基數 10 的次方表示位數值	10^2	10^1	10^0

圖 D.3　　十進位制內的位數值

至於更長的十進位數，往左算的下幾位數分別是千位數 (10 的 3 次方)、萬位數 (10 的 4 次方)、十萬位數 (10 的 5 次方)、百萬位數 (10 的 6 次方)、千萬位數 (10 的 7 次方)，依此類推。

就二進位數 101 而言，我們則說最右邊的 1 是 1 位數 (ones position)、0 是 2 位數 (twos position)、最左邊的 1 是 4 位數 (fours position)。這些位數每個都是基數 (2) 的次方，這些次方從最右邊的 0 次方，隨著數值位數每次左移一位，次方數就遞增 1 (請參閱圖 D.4)。因此，$101 = 2^2 + 2^0 = 4 + 1 = 5$。

二進制內的位數值。			
二進位數字	1	0	1
位數名	4 位數	2 位數	個位數
位數值	4	2	1
以基數 2 的次方表示位數值	2^2	2^1	2^0

圖 D.4　　二進制內的位數值

至於更長的二進位數，往左算的下幾位數分別爲 8 位數 (2 的 3 次方)、16 位數 (2 的 4 次方)、32 位數 (2 的 5 次方)、64 位數 (2 的 6 次方)，依此類推。

就八進位數 425 而言，我們會說最右邊的 5 是 1 位數 (ones position)、2 是 8 位數 (eights position)、最左邊的 4 是 64 位數 (sixty-fours position)。請注意，這些位數每個都是基數 (8) 的次方，這些次方從最右邊的 0 次方，隨著數值位數每次左移一位，次方數就遞增 1 (請參閱圖 D.5)。

八進制內的位數值。

十進位數字	4	2	5
位數名	64 位數	8 位數	個位數
位數值	64	8	1
以基數 8 的次方表示位數值	8^2	8^1	8^0

圖 D.5　八進制內的位數值

至於更長的八進位數，往左算的下幾位數分別爲 512 位數 (8 的 3 次方)、4096 位數 (8 的 4 次方)、32768 位數 (8 的 5 次方)等等，依此類推。

十六進位數 3DA 中，我們會說最右邊的 A 是 1 位數 (ones position)、D 是 16 位數 (sixteens position)、最左邊的 3 是 256 位數 (two-hundred-and-fifty-sixes po sition)。請注意，這些位數每個都是基數 (16) 的次方，這些次方從最右邊的 0 次方，隨著數值位數每次左移一位，次方數就遞增 1 (請參閱圖 D.6)。

至於更長的十六進位數，往左算的下幾位數分別爲 4096 位數 (16 的 3 次方)、65536 位數 (16 的 4 次方)，依此類推。

十六進制內的位數值。

十進位數字	3	D	A
位數名	256 位數	16 位數	個位數
位數值	256	16	1
以基數 16 的次方表示位數值	16^2	16^1	16^0

圖 D.6　十六進制內的位數值

D.2　二進位數轉為八進位及十六進位

計算機領域使用八進位和十六進位的主要目的在於縮短二進位表示法。圖 D.7 說明了冗長的二進位數可以用大於 2 的基數數字系統，以較簡潔的方式來表達。

十進位 數字	二進位表示法	八進位 表示法	十六進位 表示法
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	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
16	10000	20	10

圖 D.7　　十進位、二進位、八進位、十六進位同等值的對照

八進位制與十六進位制都與二進位制有相當重要的關係，就是八進位制和十六進位制的基數 (分別爲 8 與 16) 都是二進位制基數 (2) 的次方。請考慮下面所列的 12 位數二進位制數值，以及其對應的八進位制和十六進位制的同等值。你能否看出這個關係是如何使將二進位數縮轉成八進位數及十六進位數具有便利性。這些數字的後面會說明答案。

```
二進位數          對應的八進位數  對應的十六進位數
100011010001    4321            8D1
```

要瞭解如何輕易地將二進位數轉換爲八進位數，只要從右邊開始，把這 12 位數的二進位數以每 3 個連續的位元加以分組即可，我們將分組的結果和各組對應的八進位數寫出，如下所示：

```
100     011     010     001
4       3       2       1
```

請注意：針對每組 3 個位元的 3 位數二進制數字，其所對應的八進位數字都可以在圖 D.7 裡找到。

同樣的關係也可以在將二進位數字轉爲十六進位時看到。將這 12 個位數的二進位數由右邊開始，以每 4 個連續的位元來分組，我們將分組的結果和各組對應的十六進位數寫出，結果如下：

```
1000    1101    0001
8       D       1
```

請注意：寫在每組 4 個位元下方的十六進位數字，可以在圖 D.7 裡清楚地找到該組二進位數的十六進位對應值。

D.3　八進位與十六進位轉為二進位

在前一節，我們看到如何透過二進位數字的分組，將各組數字化成對應的八進位與十六進位的方法，藉此將二進位數轉成八進位與十六進位的同等值。這個處理程序可以反過來使用，這樣就可以針對八進位數或十六進位數，產生其二進位同等值。

舉例來說，要將八進位數 653 轉換成二進位數，只要把 6 寫成 3 位數的二進位同等值 110、把 5 寫成 3 位數的二進位同等值 101、把 3 寫成 3 位數的二進位同等值 011，再組合成一個含有 9 位數的二進位數 110101011 就可以了。

十六進位數 FAD5 轉換爲二進位數時，只要把 F 寫成 4 位數的二進位同等值 1111，把 A 寫成 4 位數的二進位同等值 1010，把 D 寫成 4 位數的二進位同等值 1101，把 5 寫成 4 位數的二進位同等值 0101，再組合成一個含有 16 位數的二進位數 1111101011010101 就可以了。

D.4　二進位、八進位或十六進位轉為十進位

因爲我們習慣以十進位計算，所以常常會將二進位、八進位、或十六進位轉爲十進位，來看看這個數字「眞正」的值。在第 D.1 節的各個圖裡列出了以十進位表示的位數值。

要將數字由其它基數轉爲十進位時，只要將各位數上的符號值乘以其所在位數的位數值(十進位等同值)，再將這些乘積加總就可以了。例如，圖 D.8 裡的二進位數 110101 轉成十進位的 53。

二進位轉為十進位						
位數值	32	16	8	4	2	1
符號值	1	1	0	1	0	1
乘積	1*32=32	1*16=16	0*8=0	1*4=4	0*2=0	1*1=1
總和：	= 32 + 16 + 0 + 4 + 0s + 1 = 53					

圖 D.8　　二進位轉為十進位

圖 D.9 裡，將八進位的 7614 轉成十進位的 3980，我們也用了相同的技巧，這時要使用適當的八進位位數值。

八進位轉為十進位				
位數值	512	64	8	1
符號值	7	6	1	4
乘積	7*512=3584	6*64=384	1*8=8	4*1=4
總和：	= 3584 + 384 + 8 + 4 = 3980			

圖 D.9　　八進位轉為十進位

圖 D.10 裡，將十六進位的 AD3B 轉成十進位的 44347，我們也用了相同的技巧，這時要使用適當的十六進位位數值。

十六進位轉為十進位				
位數值	4096	256	16	1
符號值	A	D	3	B
乘積	A*4096=40960	D*256=3328	3*16=48	B*1=11
總和：	= 40960 + 3328 + 48 + 11 = 44347			

D10　十六進位轉為十進位

D.5　十進位轉為二進位、八進位、或十六進位

前一節的換算很自然地使用位數標記換算法產生出結果。從十進位轉爲二進位、八進位、或十六進位也可以採用這種換算法。

假設要將十進位的 57 轉成二進位。我們開始從右到左寫出(二進位)每一行(位數)的位數值，直到有一行(位數)的位數值大於這個十進位數爲止。由於已經超過，我們也可以將該行加以省略。因此，我們先寫出：

```
位數值：    64    32    16    8    4    2    1
```

然後略去位數值爲 64 的那一行，剩下：

```
位數值：          32    16    8    4    2    1
```

接著，我們從最左那行往右開始計算。我們把 32 除 57 會發現 57 裡有一個 32 還剩 25，所以我們在 32 那行寫下 1。我們把 16 除 25 會發現 25 裡有一個 16 還剩 9，所以我們在 16 那行寫下 1。我們把 8 除 9 會發現 9 裡有一個 8 還剩 1，所以我們在 8 那行寫下 1。當我們把下兩行的位數值除 1 時都會產生商數爲 0 的情況，所以我們在 4 和 2 那兩行寫下 0。最後，1 除 1 得 1，所以我們在 1 那行寫下 1。這樣就會得到：

```
位數值：    32    16    8    4    2    1
符號值：    1     1     1    0    0    1
```

因此十進位的 57 等於二進位的 111001。

如果要將十進位的 103 轉換爲八進位，我們一開始先寫出每一行的位數值，直到有一行的位數值大於這個十進位數爲止。由於已經超過，我們也可以將該行加以省略。因此，我們先寫出：

```
位數值：    512    64    8    1
```

然後略去位數值爲 512 的那一行，剩下：

```
位數值：          64    8    1
```

接著，我們從最左那行往右開始計算。我們把 64 除 103 會發現，103 裡有一個 64 還剩 39，所以我們在 64 那行寫下 1。我們把 8 除 39 會發現，39 裡有四個 8 還剩 7，所以我們在 16 那行寫下 4。最後，1 除 7 會發現，7 裡有七個 1 而且沒有餘數，所以我們在 1 那行寫下 7。這樣就會得到：

```
位數值：        64      8       1
符號值：        1       4       7
```

因此十進位的 103 等於八進位的 147。

如果要將十進位的 375 轉換爲十六進位，我們一開始先寫出每一行的位數值，直到有一行的位數值大於這個十進位數爲止。由於已經超過，我們也可以將該行加以省略。因此，我們先寫出：

```
位數值：        4096    256     16      1
```

然後略去位數值爲 4096 的那一行，剩下：

```
位數值：                256     16      1
```

接著，我們從最左那行往右開始計算。我們把 256 除 375 會發現，375 裡有一個 256 還剩 119，所以我們在 256 那行寫下 1。我們把 16 除 119 會發現，119 裡有七個 16 還剩 7，所以我們在 16 那行寫下 7。最後，1 除 7 會發現，7 裡有七個 1 而且沒有餘數，所以我們在 1 那行寫下 7。這樣就會得到：

```
位數值：        256     16      1
符號值：        1       7       7
```

因此十進位的 375 等於十六進位的 177。

D.6　二進位負數：二補數法

截至目前爲止，本附錄的討論都側重於正數。在這一節，我們會解釋電腦如何使用**二補數法 (two's complement notation)** 來表示二進位負數。首先，我們解釋二進位數的二補數是如何產生的，然後再說明爲何二補數能表示這個二進位數的負值。

設想有一台機器，裡面的整數爲 32 位元。假設

```
int value = 13;
```

`value` 的 32 位元表示法爲：

```
00000000 00000000 00000000 00001101
```

要得到 `value` 的負值的話，我們先透過 C++ 的**位元補數運算子 (~，bit-wise complement operator)** 來產生它的 **1 補數 (one's complement)**：

```
onesComplementOfValue = ~value;
```

~value 內已成爲 value 中每個位元反轉 (0 變 1，1 變 0) 後所產生的值，如下所示：

```
value:
00000000 00000000 00000000 00001101
~value (也就是value的1補數):
11111111 11111111 11111111 11110010
```

要得到 value 的 2 補數，我們只要把 value 的 1 補數加 1 就可以了。所以：

```
value的 2 補數:
11111111 11111111 11111111 11110011
```

現在，假如說這個數眞的等於-13，那麼我們把它和二進位數的 13 相加，應該會得到 0 這個結果。讓我們試一下：

```
 00000000 00000000 00000000 00001101
+11111111 11111111 11111111 11110011
-----------------------------------
 00000000 00000000 00000000 00000000
```

超過最左行的進位位元被捨掉，而且我們的確得到了 0 這個結果。如果把這個數加上它的 1 補數，會得到全都是 1 的結果。得到結果全爲 0 的關鍵在於 2 補數會比 1 補數大 1。加上 1 將會使得每行原來的 1 都加上進位的值 1，結果變成 0，並且進位 1。進位會一直往左移，直到它超過最左位元被捨掉爲止，因此得到位元全都是 0 的結果。

實際上，電腦要計算一個減法，例如：

```
x = a - value;
```

會透過 a 加上 value 的 2 補數來計算，如下所示：

```
x = a + (~value + 1);
```

假設 a 是 27，value 和前面一樣是 13。如果 value 的 2 補數確實是 value 的負值的話，那麼 a 加上 value 的 2 補數應該會得到 14，讓我們試一下：

```
a (也就是27)       00000000 00000000 00000000 00011011
+(~value + 1)     +11111111 11111111 11111111 11110011
                  -----------------------------------
                   00000000 00000000 00000000 00001110
```

結果的確等於 14。

摘要

- 當我們在 C++程式裡寫了一個像是 19、227、或是-63 的整數時，這個數是使用十進位制 (基數爲 10)。十進位制裡的數字有 0、1、2、3、4、5、6、7、8、與 9。最小的數字是 0，最大的數字是 9 (比基數 10 小 1)。
- 電腦內部使用二進位制 (基數爲 2) 數字系統。二進位制只有兩個數字，即 0 與 1。它的最小數字是 0，最大數字是 1 (比基數 2 小 1)。
- 八進位數字系統 (基數爲 8) 跟十六進位數字系統 (基數爲 16)，深爲大眾所愛用，主要是因爲這兩種數字系統在縮轉二進位數值上十分便利。
- 八進位數字系統的數字範圍是從 0 到 7。
- 由於十六進位制需要有 16 個數字，所以會發生一種困難：最低數字是 0 但最高數字 (比基數 16 小 1) 等同於十進位制的 15。根據協定，我們使用字母 A 到字母 F 表示十六進位的數字，這些字母 A 到字母 F 所表示的十六進位數字分別對應十進位數字的 10 到 15。
- 這些數字系統都使用位數標記法，各個寫有數字的位數都有其獨特的位數值。
- 八進位制與十六進位制都與二進位制有相當重要的關係，就是八進位制和十六進位制的基數 (分別爲 8 與 16) 都是二進位制基數 (2) 的次方。
- 將八進位轉換到二進位系統時，可以用 3 個位數的二進位制相等數去代換掉各個八進位制的數值。
- 將十六進位轉換到二進位系統時，可以用 4 個位數的二進位制相等數去代換掉各個十六進位制的數值。
- 因爲我們習慣以十進位計算，所以常常會將二進位、八進位、或十六進位轉爲十進位來看看這個數字眞正的值。
- 要將數字由其它基數轉爲十進位時，只要把每個數字的十進位同等值乘上它的位數值，再把這些乘積加總就可以了。
- 電腦使用二補數法 (two's complement notation) 來表示負數。
- 要得到某二進位值的負數的話，我們先透過 C++的位元補數運算子 (~) 來產生它的 1 補數 (one's complement)：這樣便反轉了該數值各位元的值。要得到某數值的 2 補數，我們只要把 `value` 的 1 補數加 1 就可以了。

術語

基數 (base)

八進位的數字系統 (base 8 number system)

二進位的數字系統 (base 2 number system)

十進位的數字系統 (base 10 number system)

十六進位的數字系統 (base 16 number system)
二進位制 (binary number system)
位元補數運算子 (~) [bitwise complement operator (~)]
十進位數字系統 (decimal number system)
數字 (digital)
十六進位數字系統 (hexadecimal number system)
負值 (negative value)
八進位數字系統 (octal number system)
1 補數法 (one's complement notation)
位數標記法 (positional notation)
位數值 (positional value)
符號值 (symbol value)
2 補數法 (two's complement notation)

自我測驗

D.1 二進位制、八進位制、十進位制跟十六進位制的基數分別爲_____，_____，_____與_____。

D.2 一般說來，一個二進位制的十進位制、八進位制跟十六進位制表示法所含的數字會比二進位制數字所含的來得 (多／少)。

D.3 （是非題）使用十進制的一般理由是它可以很容易地透過一個十進位數字代換一組四個二進位位元的方法將二進位數縮轉爲十進位。

D.4 以 (八進位／十六進位/十進位) 來表示二進位數值是三者中最簡潔的。

D.5 （是非題）任何數字系統基數內的最大數字都會比基數大 1。

D.6 (是非題) 任何數字系統基數內的最小數字都會比基數小 1。

D.7 不管在二進位、八進位、十進位、或十六進位，所有數值最右邊那個數字的位數值都爲_____。

D.8 二進位、八進位、十進位、或十六進位，所有數值最右數字左邊那個數字的位數值都爲_____。

D.9 在表中每個指定的數字系統內填入最右四位數的位數值：

十進位	1000	100	10	1
十六進位	...	256	...	...
二進位	...	...	...	...
八進位	512	...	8	...

D.10 將二進位的 110101011000 轉爲八進位和十六進位。

D.11 將十六進位的 FACE 轉爲二進位。

D.12 將八進位的 7316 轉爲二進位。

D.13 將十六進位的 4FEC 轉爲八進位。[提示：先將 4FEC 轉爲二進位，再將二進位數值轉爲八進位。)

D.14 將二進位的 1101110 轉爲十進位。

D.15 將八進位的 317 轉爲十進位。

D.16 將十六進位的 EFD4 轉爲十進位。

D.17 將十進位的 177 轉爲二進位、八進位和十六進位。

D.18 寫出十進位 417 的二進位表示法，然後再寫出 417 的 1 補數和 2 補數。

D.19 某個數值與其 2 補數相加之後，所得結果爲何？

自我測驗解答

D.1 10, 2, 8, 16.

D.2 少。

D.3 錯。十六進位。

D.4 十六進位。

D.5 錯。任何基數內的最大數字都會比基數小 1。

D.6 錯。任何基數內的最小數字都爲 0。

D.7 1 (基數的 0 次方)。

D.8 數字系統的基數。

D.9 填好的圖表如下所示：

十進位	1000	100	10	1
十六進位	4096	256	16	1
二進位	8	4	2	1
八進位	512	64	8	1

D.10 八進位 6530：十六進位 D58。

D.11 二進位 1111 1010 1100 1110。

D.12 二進位 111 011 001 110。

D.13 二進位 0 100 111 111 101 100；八進位 47754。

D.14 十進位 2+4+8+32+64＝110。

D.15 十進位 7 + 1 * 8 + 3 * 64 = 7 + 8 + 192 = 207。

D.16 十進位 4+13*16+15*256+14*4096＝61396。

D.17 十進位 177。

轉爲二進位：

```
256 128 64 32 16 8 4 2 1
128 64 32 16 8 4 2 1
(1*128)+(0*64)+(1*32)+(1*16)+(0*8)+(0*4)+(0*2)+(1*1)
10110001
```

轉爲八進位：

```
512 64 8 1
64 8 1
(2*64)+(6*8)+(1*1)
261
```

轉爲十六進位：

```
256 16 1
16 1
(11*16)+(1*1)
(B*16)+(1*1)
B1
```

D.18 二進位：

```
512 256 128 64 32 16 8 4 2 1
256 128 64 32 16 8 4 2 1
(1*256)+(1*128)+(0*64)+(1*32)+(0*16)+(0*8)+(0*4)+(0*2)+(1*1)
110100001
```

One's complement: 001011110
Two's complement: 001011111
Check: Original binary number + its two's complement

```
110100001
001011111
---------
000000000
```

D.19 零。

習題

D.20 因爲 12 比 10 (基數 10) 能夠被更多數除盡，所以有些人士主張採用基數 12 的數值系統，如此一來我們許多計算將會更簡易。基數 12 的數字系統中最小數字爲何？基數 12 的數字系統中最大數字將是甚麼數字？基數 12 的數字系統中，任何數值最右邊四位數的位數值爲何？

D.21 根據下列數值系統的指示，完成下表最右邊四位數的位數值。

十進位	1000	100	10	1
基數 6	...	...	6	...
基數 13	...	169	...	...
基數 3	27	...	...	...

D.22 將二進位的 100101111010 轉爲八進位和十六進位。

D.23 將十六進位的 3A7D 轉爲二進位。

D.24 將十六進位的 765F 轉爲八進位。[提示：先將 765F 轉爲二進位，再將二進位數值轉爲八進位。]

D.25 將二進位的 1011110 轉爲十進位。

D.26 將八進位的 426 轉爲十進位。

D.27 將十六進位的 FFFF 轉爲十進位。

D.28 將十進位的 299 轉爲二進位、八進位和十六進位。

D.29 寫出十進位 779 的二進位表示法，然後再寫出 779 的 1 補數和 2 補數。

D.30 某台電腦的整數佔 32 個位元，請寫出在此電腦中，整數－1 的 2 補數。