Numeral Systems

It is important that you understand in any numeral system the individual **digits** of a number are multiplied by some power of the **base** when you are computing what the number is. For example, in the **decimal** (or **base 10**) numeral system, the number 713 can be written as:

$$\begin{aligned} 713 &= (700) + (10) + (3) \\ &= (\mathbf{7} \times 100) + (\mathbf{1} \times 10) + (\mathbf{3} \times 1) \\ &= (\mathbf{7} \times 10^2) + (\mathbf{1} \times 10^1) + (\mathbf{3} \times 10^0) \end{aligned}$$

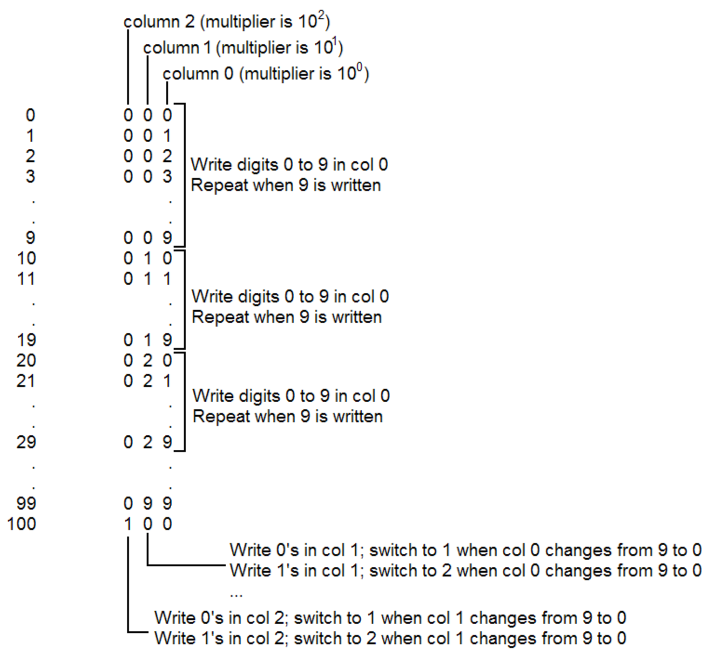
That is 7 hundreds, 1 ten, and 3 ones

Each digit of a decimal number is in some **power of 10** position. Starting from the right and working toward the left, the multipliers are 10^0, 10^1, 10^2, 10^3, 10^4, 10^5, and so on for however many digits are in the number.

When working with different numeral systems, numbers are sometimes written with a subscript following the number to indicate the base. This is done to eliminate confusion. Thus 713_{10} is the number 713 in the decimal (base 10) numeral system.

Writing Numbers in Decimal or Base 10

As you are writing successive numbers in the decimal numeral system, you follow specific rules that determine what the next number is You learned these rules at an early age, and it is so natural for you now that you have probably forgotten the rules. For example, consider the first 101 decimal numbers (0 through 100) written in the left column, and the same numbers but with leading zeros in the right column:



Starting at the top, and scanning down in columns 0, 1, and 2, notice that the pattern 0, 1, 2, …, 9 repeats in column 0 ten times (first with a preceding zero, then with a preceding one, then with a preceding two, …), while the pattern 000000000, 1111111111, …, 999999999 repeats in the middle column, while the pattern 0000000… (actually 100 zeroes) repeats in the third column. The basic algorithm for writing the first 1000 integers (0 to 999) is shown below:

```
1. c0_digit ← 0; c1_digit ← 0; c2_digit ← 0
2. write c2_digit followed by c1_digit followed by c0_digit
3. set c0_digit to (c0_digit + 1) modulo 10
4. if c0_digit is not 0 then go to step 2; otherwise go to step 5
5. set c1_digit to (c1_digit + 1) modulo 10
6. if c1_digit is not 0 then go to step 2; otherwise go to step 7
7. set c2_digit to (c2_digit + 1) modulo 10
8. if c2_digit is 0 and c1_digit is 0 and c0_digit is 0 go to step 9; otherwise go to step 2
9. stop
```

Let's try this:

```
1. c0_digit ← 0; c1_digit ← 0; c2_digit ← 0
2. write c2_digit followed by c1_digit followed by c0_digit:  0  0  0
3. set c0_digit to 1
4. c0_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  0  1
3. set c0_digit to 2
4. c0_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  0  2
3. set c0_digit to 3
4. c0_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  0  3
... and so on until
3. set c0_digit to 9
4. c0_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  0  9
3. set c0_digit to 0
4. c0_digit is 0 so go to step 5
5. set c1_digit to 1
6. c1_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  1  0
3. set c0_digit to 1
4. c0_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  1  1
3. set c0_digit to 2
4. c0_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  1  2
3. set c0_digit to 3
4. c0_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  1  3
... and so on until
3. set c0_digit to 9
4. c0_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  1  9
3. set c0_digit to 0
4. c0_digit is 0 so go to step 5
5. set c1_digit to 2
6. c1_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  2  0
... and so on until both c1_digit and c0_digit are 9
2. write c2_digit followed by c1_digit followed by c0_digit:  0  9  9
3. set c0_digit to 0
4. c0_digit is 0 so go to step 5
5. set c1_digit to 0.
6. c1_digit is 0 so go to step 7
7. set c2_digit to 1
8. c2_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  1  0  0
... and so on until 9  9  9 is written
2. write c2_digit followed by c1_digit followed by c0_digit:  9  9  9
3. set c0_digit to 0
4. c0_digit is 0 so go to step 5
5. set c1_digit to 0
6. c1_digit is 0 so go to step 7
7. set c2_digit to 0
8. c2_digit is 0 and c1_digit is 0 and c0_digit is 0 so go to step 9
9. stop
```

All numeral systems use the same basic algorithm for writing successive numbers. It just seems confusing when you try to write the numbers in a numeral system other than decimal because you don't commonly think about the individual algorithm steps when you are writing the numbers out.

The Binary or Base 2 Numeral System

In the binary numeral system there are two digits: 0 and 1. The algorithm for writing the first eight binary numbers is:

```
1. c0_digit ← 0; c1_digit ← 0; c2_digit ← 0
2. write c2_digit followed by c1_digit followed by c0_digit
3. set c0_digit to (c0_digit + 1) modulo 2
4. if c0_digit is not 0 then go to step 2; otherwise go to step 5
5. set c1_digit to (c1_digit + 1) modulo 2
6. if c1_digit is not 0 then go to step 2; otherwise go to step 7
7. set c2_digit to (c2_digit + 1) modulo 2
8. if c2_digit is 0 and c1_digit is 0 and c0_digit is 0 go to step 9; otherwise go to step 2
9. stop
```

Let's try this:

```
1. c0_digit ← 0; c1_digit ← 0; c2_digit ← 0
2. write c2_digit followed by c1_digit followed by c0_digit:  0  0  0
3. set c0_digit to 1
4. c0_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  0  1
3. set c0_digit to 0
4. c0_digit is 0 so go to step 5
5. set c1_digit to 1
6. c1_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  1  0
3. set c0_digit to 1
4. c0_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  0  1  1
3. set c0_digit to 0
4. c0_digit is 0 so go to step 5
5. set c1_digit to 0
6. c1_digit is 0 so go to step 7
7. set c2_digit to 1
8. c2_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  1  0  0
3. set c0_digit to 1
4. c0_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  1  0  1
3. set c0_digit to 0
4. c0_digit is 0 so go to step 5
5. set c1_digit to 1
6. c1_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  1  1  0
3. set c0_digit to 1
4. c0_digit is not 0 so go to step 2
2. write c2_digit followed by c1_digit followed by c0_digit:  1  1  1
3. set c0_digit to 0
4. c0_digit is 0 so go to step 5
5. set c1_digit to 0
6. c1_digit is 0 so go to step 7
7. set c2_digit to 0
8. c2_digit is 0 and c1_digit is 0 and c0_digit is 0 so go to step 9
9. stop
```

Therefore, the first eight binary numbers are: 000_2, 001_2, 010_2, 011_2, 100_2, 101_2, 110_2, 111_2. The following table lists the first forty decimal numbers (0-39) and their binary equivalents:

0_{10}	000000_2	8_{10}	001000_2	16_{10}	010000_2	24_{10}	011000_2	32_{10}	100000_2
1_{10}	000001_2	9_{10}	001001_2	17_{10}	010001_2	25_{10}	011001_2	33_{10}	100001_2
2_{10}	000010_2	10_{10}	001010_2	18_{10}	010010_2	26_{10}	011010_2	34_{10}	100010_2
3_{10}	000011_2	11_{10}	001011_2	19_{10}	010011_2	27_{10}	011011_2	35_{10}	100011_2
4_{10}	000100_2	12_{10}	001100_2	20_{10}	010100_2	28_{10}	011100_2	36_{10}	100100_2
5_{10}	000101_2	13_{10}	001101_2	21_{10}	010101_2	29_{10}	011101_2	37_{10}	100101_2
6_{10}	000110_2	14_{10}	001110_2	22_{10}	010110_2	30_{10}	011110_2	38_{10}	100110_2
7_{10}	000111_2	15_{10}	001111_2	23_{10}	010111_2	31_{10}	011111_2	39_{10}	100111_2

Now, given a binary number, it is possible to determine what the number is in decimal without having a table in front of you. The trick is to remember that each digit of a binary number is multiplied by some **power of 2** (because the binary numeral system is **base 2**). For example,

$$
\begin{aligned}
11010110_2 &= (\mathbf{1} \times 2^7) + (\mathbf{1} \times 2^6) + (\mathbf{0} \times 2^5) + (\mathbf{1} \times 2^4) + (\mathbf{0} \times 2^3) + (\mathbf{1} \times 2^2) + (\mathbf{1} \times 2^1) + (\mathbf{0} \times 2^0) \\
&= (\mathbf{1} \times 128) + (\mathbf{1} \times 64) + (\mathbf{0} \times 32) + (\mathbf{1} \times 16) + (\mathbf{0} \times 8) + (\mathbf{1} \times 4) + (\mathbf{1} \times 2) + (\mathbf{0} \times 1) \\
&= (128) + (64) + (0) + (16) + (0) + (4) + (2) + (0) \\
&= 214_{10}
\end{aligned}
$$

It is also possible to convert a decimal number into an equivalent binary number. One simple algorithm is to use the following **repeated-divide-by-two** method:

```
-- Assume N is a decimal integer you wish to convert to binary.
1. if N is less than 2 then write N and go to step 8; otherwise go to step 2.
2. divide N by 2 obtaining a quotient (Q) and a remainder (R).
3. if this was the first division, then write R and go to step 5; otherwise go to step 4.
4. write the remainder R to the left of the previous remainder.
5. set N to the quotient Q.
6. if N is greater than or equal to 2 then go to step 2; otherwise go to step 7.
7. if N is nonzero then write N to the left of the previous remainder
8. stop.
```

For example, convert the decimal number 34_{10} to binary:

```
-- N is 43₁₀
1. N is not less than 2 so go to step 2
2. divide N = 43 by 2 obtaining the quotient Q = 21 and the remainder R = 1
3. this was the first division so write R = 1 and go to step 5                    1
5. N is set to Q which is 21
6. N = 21 is greater than or equal to 2 so go to step 2
2. divide N = 21 by 2 obtaining the quotient Q = 10 and the remainder R = 1
3. this was not the first division so go to step 4.
4. write R = 1 to the left of the previous remainder                             11
5. set N to Q which is 10
6. N = 10 is greater than or equal to 2 so go to step 2
2. divide N = 10 by 2 obtaining the quotient Q = 5 and the remainder R = 0
3. this was not the first division so go to step 4.
4. write R = 0 to the left of the previous remainder                            011
5. set N to Q which is 5
6. N = 5 is greater than or equal to 2 so go to step 2
2. divide N = 5 by 2 obtaining the quotient Q = 2 and the remainder R = 1
3. this was not the first division so go to step 4.
4. write R = 1 to the left of the previous remainder                           1011
5. set N to Q which is 2
6. N = 2 is greater than or equal to 2 so go to step 2
2. divide N = 2 by 2 obtaining the quotient Q = 1 and the remainder R = 0
3. this was not the first division so go to step 4.
4. write R = 0 to the left of the previous remainder                          01011
5. set N to Q which is 1
6. N = 1 is less than 2 so go to step 7
7. N is 1 which is nonzero so write N to the left of the previous remainder  101011
8. stop
```

Therefore the decimal number 43_{10} is equivalent to the binary number 101011_2. Of course we can verify this:

$$
\begin{aligned}
101011_2 &= (\mathbf{1} \times 2^5) + (\mathbf{0} \times 2^4) + (\mathbf{1} \times 2^3) + (\mathbf{0} \times 2^2) + (\mathbf{1} \times 2^1) + (\mathbf{1} \times 2^0) \\
&= (\mathbf{1} \times 32) + (\mathbf{0} \times 16) + (\mathbf{1} \times 8) + (\mathbf{0} \times 4) + (\mathbf{1} \times 2) + (\mathbf{1} \times 1) \\
&= (32) + (0) + (8) + (0) + (2) + (1) \\
&= 43_{10}
\end{aligned}
$$

Octal or Base 8

In the octal numeral system there are eight digits: 0, 1, 2, 3, 4, 5, 6, and 7. The following table lists the first forty decimal numbers and their octal equivalents. Notice the pattern or algorithm that is used when writing the numbers in octal. See how we write 0-7 repeatedly on column 0, and increment column 1 when column 0 goes from 7 to 0.

0_{10}	00_8	8_{10}	10_8	16_{10}	20_8	24_{10}	30_8	32_{10}	40_8
1_{10}	01_8	9_{10}	11_8	17_{10}	21_8	25_{10}	31_8	33_{10}	41_8
2_{10}	02_8	10_{10}	12_8	18_{10}	22_8	26_{10}	32_8	34_{10}	42_8
3_{10}	03_8	11_{10}	13_8	19_{10}	23_8	27_{10}	33_8	35_{10}	43_8
4_{10}	04_8	12_{10}	14_8	20_{10}	24_8	28_{10}	34_8	36_{10}	44_8
5_{10}	05_8	13_{10}	15_8	21_{10}	25_8	29_{10}	35_8	37_{10}	45_8
6_{10}	06_8	14_{10}	16_8	22_{10}	26_8	30_{10}	36_8	38_{10}	46_8
7_{10}	07_8	15_{10}	17_8	23_{10}	27_8	31_{10}	37_8	39_{10}	47_8

To convert an octal number to decimal, remember that each digit of a octal number is multiplied by some **power of 8**. For example,

$$\begin{aligned}
71325_8 &= (\mathbf{7} \times 8^4) + (\mathbf{1} \times 8^3) + (\mathbf{3} \times 8^2) + (\mathbf{2} \times 8^1) + (\mathbf{5} \times 8^0) \\
&= (\mathbf{7} \times 4096) + (\mathbf{1} \times 512) + (\mathbf{3} \times 64) + (\mathbf{2} \times 8) + (\mathbf{5} \times 1) \\
&= (28672) + (512) + (192) + (16) + (5) \\
&= 29397_{10}
\end{aligned}$$

To convert a decimal number to octal, use the following **repeated-divide-by-eight** method:

```
-- Assume N is a decimal integer you wish to convert to octal.
1. if N is less than 8 then write N and go to step 8; otherwise go to step 2.
2. divide N by 8 obtaining a quotient (Q) and a remainder (R).
3. if this was the first division, then write R and go to step 5; otherwise go to step 4.
4. write the remainder R to the left of the previous remainder.
5. set N to the quotient Q.
6. if N is greater than or equal to 8 then go to step 2; otherwise go to step 7.
7. if N is nonzero then write N to the left of the previous remainder
8. stop.
```

For example, convert the decimal number 172_{10} to octal:

```
-- N is 172₁₀
1. N is not less than 8 so go to step 2
2. divide N = 172 by 8 obtaining the quotient Q = 21 and the remainder R = 4
3. this was the first division so write R = 4 and go to step 5                      4
5. N is set to Q which is 21
6. N = 21 is greater than or equal to 8 so go to step 2
2. divide N = 21 by 8 obtaining the quotient Q = 2 and the remainder R = 5
3. this was not the first division so go to step 4.
4. write R = 5 to the left of the previous remainder                               54
5. set N to Q which is 2
6. N = 2 is less than 8 so go to step 7
7. N = 2 is nonzero so write N to the left of the previous remainder              254
8. stop
```

Therefore the decimal number 172_{10} is equivalent to the octal number 254_8. Of course we can verify this:

$$\begin{aligned}
254_8 &= (\mathbf{2} \times 8^2) + (\mathbf{5} \times 8^1) + (\mathbf{4} \times 8^0) \\
&= (\mathbf{2} \times 64) + (\mathbf{5} \times 8) + (\mathbf{4} \times 1) \\
&= (128) + (40) + (4) \\
&= 172_{10}
\end{aligned}$$

Hexadecimal or Base 16

In the hexadecimal numeral system (often called **hex**) there are sixteen digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F. The important thing to remember is that the hexadecimal digit **A** is equivalent to **10** in decimal, **B** is equivalent to **11** in decimal, **C** is **12** in decimal, **D** is **13** in decimal, **E** is **14** in decimal, and **F** is **15** in decimal.

The following table lists the first forty decimal numbers and their hexadecimal equivalents:

0_{10}	00_{16}	8_{10}	08_{16}	16_{10}	10_{16}	24_{10}	18_{16}	32_{10}	20_{16}
1_{10}	01_{16}	9_{10}	09_{16}	17_{10}	11_{16}	25_{10}	19_{16}	33_{10}	21_{16}
2_{10}	02_{16}	10_{10}	$0A_{16}$	18_{10}	12_{16}	26_{10}	$1A_{16}$	34_{10}	22_{16}
3_{10}	03_{16}	11_{10}	$0B_{16}$	19_{10}	13_{16}	27_{10}	$1B_{16}$	35_{10}	23_{16}
4_{10}	04_{16}	12_{10}	$0C_{16}$	20_{10}	14_{16}	28_{10}	$1C_{16}$	36_{10}	24_{16}
5_{10}	05_{16}	13_{10}	$0D_{16}$	21_{10}	15_{16}	29_{10}	$1D_{16}$	37_{10}	25_{16}
6_{10}	06_{16}	14_{10}	$0E_{16}$	22_{10}	16_{16}	30_{10}	$1E_{16}$	38_{10}	26_{16}
7_{10}	07_{16}	15_{10}	$0F_{16}$	23_{10}	17_{16}	31_{10}	$1F_{16}$	39_{10}	27_{16}

To convert a hexadecimal number to decimal, remember that each digit of a hexadecimal number is multiplied by some **power of 16**. For example,

$$
\begin{array}{lllllll}
1FA2C_{16} & = & (\mathbf{1} \times 16^4) & + (\mathbf{F} \times 16^3) & + (\mathbf{A} \times 16^2) & + (\mathbf{2} \times 16^1) & + (\mathbf{C} \times 16^0) \\
 & = & (\mathbf{1} \times 16^4) & + (\mathbf{15} \times 16^3) & + (\mathbf{10} \times 16^2) & + (\mathbf{2} \times 16^1) & + (\mathbf{12} \times 16^0) \\
 & = & (\mathbf{1} \times 65536) & + (\mathbf{15} \times 4096) & + (\mathbf{10} \times 256) & + (\mathbf{2} \times 16) & + (\mathbf{12} \times 1) \\
 & = & (65536 & + (61440) & + (2560) & + (32) & + (12) \\
 & = & 129580_{10} & & & &
\end{array}
$$

To convert a decimal number to hex, use the following **repeated-divide-by-sixteen** method:

```
-- Assume N is a decimal integer you wish to convert to hex.
1. if N is less than 16 then write N and go to step 8; otherwise go to step 2.
2. divide N by 16 obtaining a quotient (Q) and a remainder (R).
3. if this was the first division, then write R and go to step 5; otherwise go to step 4.
4. write the remainder R to the left of the previous remainder.
5. set N to the quotient Q.
6. if N is greater than or equal to 16 then go to step 2; otherwise go to step 7.
7. if N is nonzero then write N to the left of the previous remainder
8. stop.
```

For example, convert the decimal number 49571_{10} to hexadecimal:

```
-- N is 49571_10
1. N is not less than 16 so go to step 2
2. divide N = 49571 by 16 obtaining the quotient Q = 3098 and the remainder R = 3
3. this was the first division so write R = 3 and go to step 5                     3
5. N is set to Q which is 3098
6. N = 3098 is greater than or equal to 16 so go to step 2
2. divide N = 3098 by 16 obtaining the quotient Q = 193 and the remainder R = A (10_10)
3. this was not the first division so go to step 4.
4. write R = A to the left of the previous remainder                              A3
5. N is set to Q which is 193
6. N = 193 is greater than or equal to 16 so go to step 2
2. divide N = 193 by 16 obtaining the quotient Q = C (12_10) and the remainder R = 1
3. this was the first division so write R = 1 and go to step 5                   1A3
5. N is set to Q which is C (12_10)
6. N = C (12_10) is less than 16 so go to step 7
7. N = C (12_10) is nonzero so write N to the left of the previous remainder    C1A3
8. stop
```

Therefore the decimal number 49571_{10} is equivalent to the hex number $C1A3_{16}$. Of course we can verify this:

$$
\begin{array}{lllllll}
C1A3_{16} & = & (\mathbf{C} \times 16^3) & + (\mathbf{1} \times 16^2) & + (\mathbf{A} \times 16^1) & + (\mathbf{3} \times 16^0) & \\
 & = & (\mathbf{12} \times 4096) & + (\mathbf{1} \times 256) & + (\mathbf{10} \times 16) & + (\mathbf{3} \times 1) & \\
 & = & (49152) & + (256) & + (160) & + (3) & = 49571_{10}
\end{array}
$$

Binary ↔ Hexadecimal

It is possible and very easy to convert a number written in binary into an equivalent number written in hexadecimal. To convert from binary to hex, know that each group of **four** binary digits is written as one hex digit. For example, to convert the number 11001011011111_2 to hex, group the binary digits into groups of four:

0011	0010	1101	1111
↓	↓	↓	↓
$0+0+2+1 = 3_{10}$	$0+0+2+0 = 2_{10}$	$8+4+0+1 = 13_{10}$	$8+4+2+1 = 15_{10}$

Remember that 13_{10} in D_{16} and 15_{10} is F_{16}, so the answer is $32DF_{16}$. To convert from hexadecimal to binary, simply write each hex digit as **four** binary digits. For example, to convert the number $B7AF4_{16}$ to binary:

B	7	A	F	4
11_{10}	7_{10}	10_{10}	15_{10}	4_{10}
8+0+2+1	0+4+2+1	8+0+2+0	8+4+2+1	0+4+0+0
1010	0111	1010	1111	0100

Thus the answer is 10100111101011110100_2.

Binary ↔ Octal

To convert from binary to octal, know that each group of **three** binary digits is written as one octal digit. For example, convert the number 11001011011111_2 to octal:

011	001	011	011	111
↓	↓	↓	↓	↓
0+2+1 = 3	0+0+1 = 1	0+2+1 = 3	0+2+1 = 3	4+2+1 = 7

Thus the answer is 31337_8. To convert from octal to binary, simply write each octal digit as **three** binary digits. For example, convert the number 713245_8 to binary:

7	1	3	2	4	5
4+2+1	0+0+1	0+2+1	0+2+0	4+0+0	4+0+1
111	001	011	010	100	101

Thus the answer is 111001011010100101_2.

Octal ↔ Hexadecimal

To convert from octal to hexadecimal, rewrite the octal number as binary and then write every group of four bits as a hexadecimal digit. Likewise, to convert from hexadecimal to octal, rewrite the hexadecimal number as binary, and then write every group of three bits as an octal digit.

Base *b*

I hope you noticed that there is a pattern to each numeral system. In general, for base b, $b \geq 2$, there are b digits 0 through $(b - 1)$[1]. When writing the number, each digit in the number is multiplied by a power of b (that is, the rightmost digit is multiplied by b^0, the digit to the left of that one by b^1, the digit to the left of that one by b^2, and so on.

To convert a n-digit base b number $(d_{n-1}...d_2d_1d_0)_b$ into decimal use the formula,

$$(d_{n-1}...d_2d_1d_0)_b = (\boldsymbol{d_{n-1}} \times b^{n-1}) + (\boldsymbol{d_{n-2}} \times b^{n-2}) + ... + (\boldsymbol{d_1} \times b^1) + (\boldsymbol{d_0} \times b^0)$$

[1] If b > 10, then symbols other than 0, 1, 2, ..., 9 must be used to represent the remaining b - 10 digits in the numeral system; e.g., in hex we use the symbols A, B, C, D, E, and F to represent the last six hex digits. English letters were chosen because it seems natural to English speakers, but there is no reason the last six digits could not be ☺, ⬟, ◈, ▷, ➞, and 🕷.

To convert a decimal number *N* into an equivalent base *b* number, use the **repeated-divide-by-*b*** algorithm:

```
-- Assume N is a decimal integer you wish to convert to base b
1. if N is less than b then write N and go to step 8; otherwise go to step 2.
2. divide N by b obtaining a quotient (Q) and a remainder (R).
3. if this was the first division, then write R and go to step 5; otherwise go to step 4.
4. write the remainder R to the left of the previous remainder.
5. set N to the quotient Q.
6. if N is greater than or equal to b then go to step 2; otherwise go to step 7.
7. if N is nonzero then write N to the left of the previous remainder
8. stop.
```

For example, consider base 5 (i.e., $b = 5$). First, the digits in the base 5 numeral system are 0, 1, 2, 3, and 4 (note that $b - 1 = 4$). To convert a number such as 31240_5 written in base 5 into an equivalent decimal number,

$$
\begin{aligned}
31240_5 &= (\mathbf{3} \times 5^4) + (\mathbf{1} \times 5^3) + (\mathbf{2} \times 5^2) + (\mathbf{4} \times 5^1) + (\mathbf{0} \times 5^0) \\
&= (\mathbf{3} \times 625) + (\mathbf{1} \times 125) + (\mathbf{2} \times 25) + (\mathbf{4} \times 5) + (\mathbf{0} \times 1) \\
&= (1875) + (125) + (50) + (20) + (0) \\
&= 2070_5
\end{aligned}
$$

To convert a decimal number into an equivalent base 5 number use the repeated-divide-by-5-algorithm. For example, convert 819_{10} into base 5:

```
-- N is 819_10
1. N is not less than 5 so go to step 2
2. divide N = 819 by 5 obtaining the quotient Q = 163 and the remainder R = 3
3. this was the first division so write R = 3 and go to step 5                   4
5. N is set to Q which is 163
6. N = 163 is greater than or equal to 5 so go to step 2
2. divide N = 163 by 5 obtaining the quotient Q = 32 and the remainder R = 3
3. this was not the first division so go to step 4.
4. write R = 3 to the left of the previous remainder                             34
5. N is set to Q which is 32
6. N = 32 is greater than or equal to 5 so go to step 2
2. divide N = 32 by 5 obtaining the quotient Q = 6 and the remainder R = 2
3. this was the first division so write R = 2 and go to step 5                  234
5. N is set to Q which is 6
6. N = 6 is greater than or equal to 5 so go to step 2
2. divide N = 6 by 5 obtaining the quotient Q = 1 and the remainder R = 1
3. this was the first division so write R = 1 and go to step 5                 1234
5. N is set to Q which is 1
6. N = 1 is less than 5 so go to step 7
7. N = 1 is nonzero so write N to the left of the previous remainder          11234
8. stop
```

Therefore the decimal number 819_{10} is equivalent to the base 5 number 11234_5. Of course we can verify this:

$$
\begin{aligned}
11234_5 &= (\mathbf{1} \times 5^4) + (\mathbf{1} \times 5^3) + (\mathbf{2} \times 5^2) + (\mathbf{3} \times 5^1) + (\mathbf{4} \times 5^0) \\
&= (\mathbf{1} \times 625) + (\mathbf{1} \times 125) + (2 \times 25) + (\mathbf{3} \times 5) + (\mathbf{4} \times 1) \\
&= (625) + (125) + (50) + (15) + (4) \\
&= 819_{10}
\end{aligned}
$$