Programming Practice: Numeral Representations

1. Write a C program to input a positive integer n less than or equal to 4,294,967,295. Print the 32-bit representation as a binary numeral, an octal numeral, and a hexadecimal numeral. For each case, show the leading zeros. The binary numeral is printed in the following format $b_{31}b_{30}b_{29}b_{28}$ $b_{27}b_{26}b_{25}b_{24}$ $b_{23}b_{22}b_{21}b_{20}$ $b_{19}b_{18}b_{17}b_{16}$ $b_{15}b_{14}b_{13}b_{12}$ $b_{11}b_{10}b_9b_8$ $b_7b_6b_5b_4$ $b_3b_2b_1b_0$, that is, every four bits are separated by a space. The octal numeral is printed as $o_{10}o_9$ $o_8o_7o_6$ $o_5o_4o_3$ $o_2o_1o_0$, that is, every three octal digits are separated by a space and the left-most-end contains two digits only. The hexadecimal numeral is printed as $x_7x_6x_5x_4$ $x_3x_0x_1x_0$, that is, every four digits are separated by a space. Solutions: integer_to_2_8_16_numeral.c. Sample output:

```
Input a positive integer: 847309329
Binary numeral:      0011 0010 1000 0000 1110 1010 0001 0001
Octal numeral:       06 240 165 021
Hexadecimal numeral: 3280 EA11
```

2. Repeat Problem 1, but without leading zero's.
   Solutions: multiplication_table_hexadecimal.c. Sample output:

```
Input a positive integer: 847309329
Binary numeral:      11 0010 1000 0000 1110 1010 0001 0001
Octal numeral:       6 240 165 021
Hexadecimal numeral: 3280 EA11
```

3. Write a C program to perform conversion of binary, octal, decimal and hexadecimal numerals. Each iteration input an integer x and a string s. Input integer x is one of 0, 2, 8, 10, and16, denoting stop program, binary numeral, octal numeral, decimal numeral, and hexadecimal numeral, respectively. That is, the program terminates, when the input is 0. The second input s is a string representing a numeral of base_x, where x is the value of the first input 2, 8, 10, or 16. The numeral of the second input is no more than the 32-bit value of base_x. Convert the input numeral of the other three bases of 2, 8, 10, and 16 that is different for x. Solutions: numeral_conversion.c. Sample output:

```
Input the numeral base and the numeral as a string: 2 10101010100011100101010
The input numeral 10101010100011100101010 of base 2 has the decimal value: 2794389
Octal numeral:       12 521 625
Decimal numeral:     2794389
Hexadecimal numeral: 2A A395

Input the numeral base and the numeral as a string: 8 30241
The input numeral 30241 of base 8 has the decimal value: 12449
Binary numeral:      11 0000 1010 0001
Decimal numeral:     12449
Hexadecimal numeral: 30A1

Input the numeral base and the numeral as a string: 10 1000000
The input numeral 1000000 of base 10 has the decimal value: 1000000
Binary numeral:      1111 0100 0010 0100 0000
Octal numeral:       3 641 100
Hexadecimal numeral: F 4240

Input the numeral base and the numeral as a string: 16 5AC09B
The input numeral 5AC09B of base 16 has the decimal value: 5947547
Binary numeral:      101 1010 1100 0000 1001 1011
Octal numeral:       26 540 233
Decimal numeral:     5947547

Input the numeral base and the numeral as a string: 0 0
```

4. Base 62 numeral is a number system of 62 digits as the following encoding scheme:

digit	value	digit	value	digit	value	digit	value	digit	value
0	0	D	13	Q	26	d	39	q	52
1	1	E	14	R	27	e	40	r	53
2	2	F	15	S	28	f	41	s	54
3	3	G	16	T	29	g	42	t	55
4	4	H	17	U	30	h	43	u	56
5	5	I	18	V	31	i	44	v	57
6	6	J	19	W	32	j	45	w	58
7	7	K	20	X	33	k	46	x	59
8	8	L	21	Y	34	l	47	y	60
9	9	M	22	Z	35	m	48	z	61
A	10	N	23	a	36	n	49		
B	11	O	24	b	37	o	50		
C	12	P	25	c	38	p	51		

Write a C programs to convert a base$_{62}$ numeral to a decimal number or backwrd. Read an integer x and a numeral string s. If x is 10, the numeral string s is a decimal numeral and convert it to the equivalent base$_{62}$ numeral; if x is 62, the numeral string is base$_{62}$ and convert it to the equivalent decimal numeral. Repeat the conversion step until x is 0. Declare the numeral value of type **unsigned long long**. An **unsigned long long** number is a 64-bit number and its maximum value is 18,446,744,073,709,551,615 (decimal, 2^{64}-1). Sample testing data:

```
62 ABC123xyz
62 FunnyNumber
62 100000
10 9999999
10 218340105584896
10 13358441988829475865
0 0
```

Solutions:base62_numeral.c. Sample output:

```
Input the numeral base and the numeral as a string: 62 ABC123xyz
Base62 numeral ABC123xyz is equivalent to decimal numeral: 2222821365975321

Input the numeral base and the numeral as a string: 62 FunnyNumber
Base62 numeral FunnyNumber is equivalent to decimal numeral: 13358441988829475865

Input the numeral base and the numeral as a string: 62 100000
Base62 numeral 100000 is equivalent to decimal numeral: 916132832

Input the numeral base and the numeral as a string: 10 9999999
Decimal number 9999999 is equivalent to base62 numeral: fxSJ

Input the numeral base and the numeral as a string: 10 218340105584896
Decimal number 218340105584896 is equivalent to base62 numeral: 100000000

Input the numeral base and the numeral as a string: 10 13358441988829475865
Decimal number 13358441988829475865 is equivalent to base62 numeral: FunnyNumber

Input the numeral base and the numeral as a string: 0 0
```