Number Representation

1. Converting a positive binary integer to decimal essentially amounts to converting a base-2 number to a base-10 number. In the base-2 number system, each digit represents a power of 2 (just as in the base-10 system, each digit represents a power of 10, e.g. 435 can be thought of as $4(10^2) + 3\ (10^1) + 5\ (10^0)$, or 4(hundreds) + 3(tens) + 5(units)).

Let's work through an example. We'll convert the positive integer represented by the binary number 10110 to decimal.

	1 * 2^4	0 * 2^3	1 * 2^2	1 * 2^1	0 * 2^0	
=	1*16	0*8	1*4	1*2	0*1	
=	16	0	4	2	0	row sum = 22

Hence, the binary number 10110 is equivalent to the positive integer 22.

Your turn: convert the binary number 10111110 to decimal.

1 0 1 1 1 1 1 0

$2^7\ 2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

128 0 32 16 8 4 2 0

190_{10}

2. Converting a positive decimal integer to its equivalent binary representation involves repeatedly dividing by 2 and figuring out the remainder. Let's work through an example. We'll convert the positive integer 53 to its equivalent binary representation.

	remainder	
2 | 53		
26	1	divide 2 into 53 to get 26 with remainder 1
13	0	divide 2 into 26 to get 13 with remainder 0
6	1	divide 2 into 13 to get 6 with remainder 1
3	0	divide 2 into 6 to get 3 with remainder 0
1	1	divide 2 into 3 to get 1 with remainder 1
0	1	divide 2 into 1 to get 0 with remainder 1
		stop when division by 2 gives you 0

Now we read the digits in the remainder column from bottom to top to get the binary representation of 53: 110101

Your turn: convert the decimal number 147 to binary.

2 | 147	
73	1
36	1
18	0
9	0
4	1
2	0
1	0
0	1

$\boxed{10010011}$

3. Converting a positive integer represented in hexadecimal to the equivalent decimal value amounts to converting a base-16 number to a base-10 number. As we work through the following example, notice the similarity between this process and the one used to convert base-2 to base-10 – they're identical apart from the fact that each digit now represents a power of 16 rather than a power of 2!

Let's convert the hexadecimal number 1CA to decimal.
Recall: A has the value 10, B – 11, C – 12, … F – 15)

	1 * 16^2	C * 16^1	A * 16^0	
=	1*256	12*16	10*1	
=	256	192	10	Row sum = 458

Therefore, $1CA_{16} = 458_{10}$

Your turn: convert the hexadecimal number 2EC to decimal.

$2 * 16^2 + E * 16^1 + C * 16^0$

$2 * 256 + 14 * 16 + 12 = \boxed{748}$

4. Converting a positive decimal integer to its equivalent hexadecimal representation involves a process that's identical to that for converting a positive decimal integer to its equivalent binary representation – the only difference is that we repeatedly divide by 16. Let's convert the decimal number 316 to hexadecimal.

		remainder	
16	316		
	19	12	divide 16 into 312 to get 19 with remainder 12
	1	3	divide 16 into 19 to get 1 with remainder 3
	0	1	divide 16 into 1 to get 0 with remainder 1
			stop when division by 16 gives you 0

Now take the numbers in the remainder column and read them bottom to top. In the process we must convert numbers 10 and higher to their equivalent hexadecimal digit (10 – A, 11 – B, …, 15 – F). Hence, the hexadecimal representation of the decimal integer 316 is 13C.

Your turn: convert the decimal number 141 to hexadecimal.

16	141	
	8	13 → D
	0	8

bottom to top

$\boxed{8D}$

for Reference

A - 10
B - 11
C - 12
D - 13
E - 14
F - 15

5. We know that a single decimal digit can represent 10 different numbers: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Two decimal digits can represent 100 different numbers: 0, 1, 2, 3, 4, …, 98, 99

Three decimal digits can represent 1000 different numbers: 0, 1, 2, 3, 4, …, 998, 999

a) In general, N decimal digits can represent 10^N different numbers: 0, 1, 2, 3, 4, …, <?>

What number should go in place of <?> 2^N 10^(n-1)

b) In this exercise, we investigate how many different *binary* numbers can be represented using N binary digits.

Suppose you have binary codes consisting of two binary digits. You can write down exactly 4 different binary codes that can be used to represent 4 different integer values:

00 01 10 11

What positive decimal integer values do these binary numbers represent?

0 1 2 3

c) Now suppose you have binary codes consisting of three binary digits. Write down all of the different binary codes that can be generated. Here's a start:

000 001 010 011 100 101 110 111

What positive decimal integer values do these binary numbers represent?

0 1 2 3 4 5 6 7

d) Now suppose you have binary codes consisting of four binary digits. Write down all of the different binary codes that can be generated. Here's a start:

0000 0001 0010 0011 0100 0101 0110 0111

1000 1001 1010 1011 1100 1101 1110 1111

What positive decimal integer values do these binary numbers represent?

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

e) Now let's generalize. Suppose you have binary codes consisting of N binary digits. How many different binary codes can be generated? (Don't attempt to write them all down!)

6. A binary digit is also referred to as a **bit** and eight bits are referred to as a **byte**. Your program instructions and the data they use are stored in memory. Memory is separated into bytes, where each byte has a unique address. Different types of data can be different sizes. Some types are 1 byte while others can be 2, 4 and even 8 bytes in size:

Integer types:

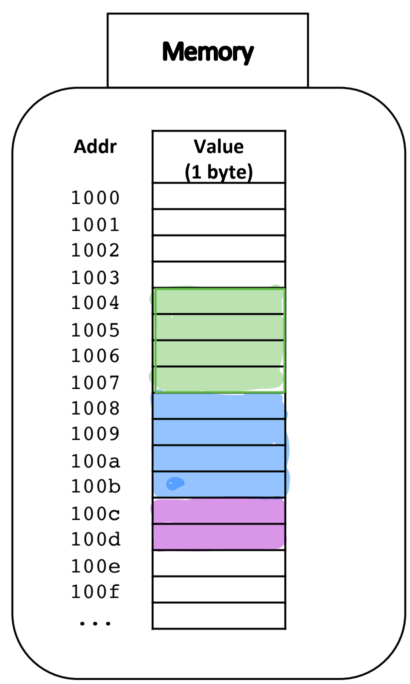
Type	Storage size	Value range
char	1 byte	-128 to 127 or 0 to 255
unsigned char	1 byte	0 to 255
signed char	1 byte	-128 to 127
int	2 or 4 bytes	-32,768 to 32,767 or -2,147,483,648 to 2,147,483,647
unsigned int	2 or 4 bytes	0 to 65,535 or 0 to 4,294,967,295
short	2 bytes	-32,768 to 32,767
unsigned short	2 bytes	0 to 65,535
long	8 bytes	-9223372036854775808 to 9223372036854775807
unsigned long	8 bytes	0 to 18446744073709551615

Floating point types:

Type	Storage size	Value range	Precision
float	4 byte	1.2E-38 to 3.4E+38	6 decimal places
double	8 byte	2.3E-308 to 1.7E+308	15 decimal places
long double	10 byte	3.4E-4932 to 1.1E+4932	19 decimal places

https://www.tutorialspoint.com/cprogramming/c_data_types.htm

To the right is a visual representation of a small chunk of memory. Each memory location is labeled with a unique address written in hexadecimal form.



a) If we knew we had 2 pieces of data in memory that were each 4 bytes in size and the first piece of data was stored at address 1004, what address do you think the second piece of data would be stored at?

Green

1008

b) If we added another piece of data 2 bytes in size, what starting address do you think it would be stored at and which addresses would it take up?

100C
100D