Binary Numbers

Numbers: the language computers understand

- At the heart, a computer isn't that different from a lightbulb: Both will eventually be translated into a series of "on" (1) and "off" (0) signals. In fact, everything your computer works with is encoded as a series of 1s and 0s and stored in its memory.
- Recall that for decimal numbers (the numbers we are use to reading), we use the digits 0 through 9, and each digit represents a new power of ten. Thus 34 is really shorthand for $3*10^1 + 4*10^0$, which is 3*10 + 4.
- This table shows how we create a number by adding together digits (columns in the table) after multiplying each digit by the appropriate power of 10.

Decimal	10^2 (=100)	10^1 (=10)	10^0 (=1)
34	100 > 34 so we don't need any digits at the 100 level	10<= 34, we can say $3*10^1$ (3) here	We still need 4, so we can say 4 $*10^0$ (4) here
The final number:	0	3	4

Binary numbers

- Binary numbers (Definition):
  - Are made up of digits that can only have two values (1 or 0) instead of 10 values (0 ... 9).
  - Instead of each digit representing a power of 10, each digit represents a new power of two.
  - Binary numbers are very useful to computers because a binary digit (called a bit) is easy to represent (since it only has two values, which in electronic terms can correspond to "on" and "off").

- Here's a video demonstration of this concept.
- Below is an example of a table like the one above showing what 34 looks like in binary. Again, this table can be read as adding together digits, in this case after multiplying each of them by an appropriate power of 2.

Binary	2^5 (= 32)	2^4 (= 16)	2^3 (= 8)	2^2 (= 4)	2^1 (= 2)	2^0 (= 1)
34	32 < 34 so we can say $1*2^5$ (32) here	We still need 2. 16 > 2, so 0 here	We still need 2. 8 > 2, so 0 here	We still need 2. 4 > 2, so 0 here	We still need 2, so we can say 1*2 here	We're done. So this digit is 0
the final number:	1	0	0	0	1	0

- In other words 34 translates to 100010, which is really shorthand for $1*2^5 + 0*2^4 + 0*2^3 + 0*2^2 + 1*2^1 + 0*2^0$, which is 32 + 0 + 0 + 0 + 2 + 0.
- We can also translate from binary numbers back into decimal. To convert from a binary number to an integer, you multiply each digit by a power of 2 and add up the results.

Translating from binary to an integer

- Consider the number 101. The location of each number indicates what power of two it represents.
- Here we have $1*2^0 + 0*2^1 + 1*2^2 = 1+0+4 = 5$

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12
Binary	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100

Learn by doing

- How many different digits are used in the binary number system?
- How many different digits are used in the decimal number system?
- How would you represent 25 in binary?
- How would you represent 26 in binary?

Our Answer

- How many different digits are used in the binary number system?
  - 2
- How many different digits are used in the decimal number system?
  - 10
- How would you represent 25 in binary?
  - In binary, each digit is multiplied by 2^n. With n increasing from 0. Thus, 1 = 1, 10 = 2, 100 = 4, 1000 = 8, 10000 = 16. Moreover, any binary number smaller than 16 will have at most 4 digits.
  - Therefore 25 = 16*1 + 8*1 + 4*0 + 2*0 +1*1 == 11001
- How would you represent 26 in binary?
  - 26 = 16*1 + 8*1 + 4*0 + 2*1 +1*0 == 11010

- *Every program* that is written for a computer is *eventually translated into a series of 1s and 0s* before the computer can understand it. This helps to explain why we write programs in languages such as Java instead of English -- a formal, well-specified language helps to bridge the gap between what we want, and what computers understand.
- English is not always clear. For example, the word *close* could be interpreted as “to close the door” or “to be close to someone”.

- Bit (definition)
  - A bit (short for *b*inary dig*it*) is a single digit of a binary number stored in a computer's memory. Just like a single decimal digit can only hold a small amount of information (9 numbers), a single bit holds a small amount of information (2 numbers, since it can only have the values 1 and 0). However, groups of bits, can be used to represent much larger numbers.
- Byte (definition)
  - A byte is eight bits. Thus, a byte can represent numbers ranging from 00000000 to 11111111 (in binary) or 0 to 255 (in decimal). These numbers represent information used by the computer to manipulate data, including numbers, the colors of the pixels of an image, or text characters.

- Let's talk a little bit about how computers count.
- People tend to start counting from 1, so if you count to 5, you have counted something 5 times. However, computers usually start counting from 0. This means that when a computer has counted to 5, it has actually counted *6* times.
- This may take a while to get used to, but once you do, it becomes second nature.

Learn by doing

- We've discussed the fact that a byte is made up of 8 bits. But how big is it -- how many different numbers could a byte represent?
  a. 257
  b. 8
  c. 7
  d. 255
  e. 256
  f. 1
  g. 9

Our Answer

- A byte contains 8 bits
- This means it can store 2^8 numbers.
- Therefore, 256.

- When data is stored as a binary number, in a byte, some sort of encoding is used to help translate from what the programmer intends to that number. A great example is text.
- Text is one of the most common types of data that is manipulated and encoded by computer programs. For text to be manipulated by programs, there must be a way of encoding each letter that a computer can understand. This is a problem that comes up over and over again.
- As a result, there is a standard solution that is used by pretty much every programming language and computer: a translation table giving a number for each character on your keyboard.
- The solution is called ASCII (American Standard Code for Information Interchange). ASCII represents each letter using 1 byte (meaning 8 bits).

- Recall that a byte can represent up to 256 separate numbers (or alphabetic characters in the case of ASCII).
- This is great for English, but not big enough to cover all of the characters used in languages around the globe.
- ASCII was internationalized into a new system called Unicode, which is designed to support an extended alphabet needed to represent other languages.
- Because this involves many more different types of characters, Unicode is not limited to 8 bits. In fact, Unicode can representation of 256 times as many characters as ASCII. New character encoding schemes based on the Unicode character standard now allow us to represent characters used in almost all written languages.

Translating from text to binary

- In ASCII, for each character in the alphabet, there is a pre-specified number used to represent it.
- This is called an encoding.
- Using the translation table provided at ascii-table.com, we will translate the word 'Wow' into binary numbers. We can look up each letter in turn. 'W' is 01010111. 'o' is 01101111. Since uppercase and lowercase letters get different codes, 'w' is different from 'W', with the code 01110111. So we can write Wow as 01010111 01101111 01110111.
- Needless to say, the process of translating things into binary is laborious for a person to complete. And frequently less straightforward than the simple direct translation of text to binary that you just experienced. For the most part, this is a job that is done for you by the compiler.

Learn by doing

The following program converts text to ASCII/Unicode In the demo provided, "Hello" is translated into ASCII, while the next two lines are translated into Unicode. Use this program to explore the differences between ASCII and Unicode.

1. How do the ASCII and Unicode characters differ when represented in binary?
2. How many bytes does each use to represent a character?

Our Answer

- The translated characters are shown one to a line in the results box.
- Each binary digit represents one bit.
- Eight bits make up a byte.
- The Unicode characters are longer than the ASCII characters.
- The ASCII characters are one byte long, the Unicode characters are between one and two bytes long.

Learn by doing

- Give an example of a computer program you use that manipulates text in some way.

Our Answer

- A word processor, a web browser, and a print manager are all examples of computer programs you probably use regularly that manipulate text. So is textpad, or any computer program that you use to program (like a compiler)!

Summary

- In summary, we've learned about numbers encoded in binary as 1s and 0s. We've also learned about other encodings such as numeric codes used to represent ASCII and Unicode text. As we will see later in this course, it is necessary to have a basic understanding of the underlying representations of things like pictures and text if you wish to manipulate them programmatically.