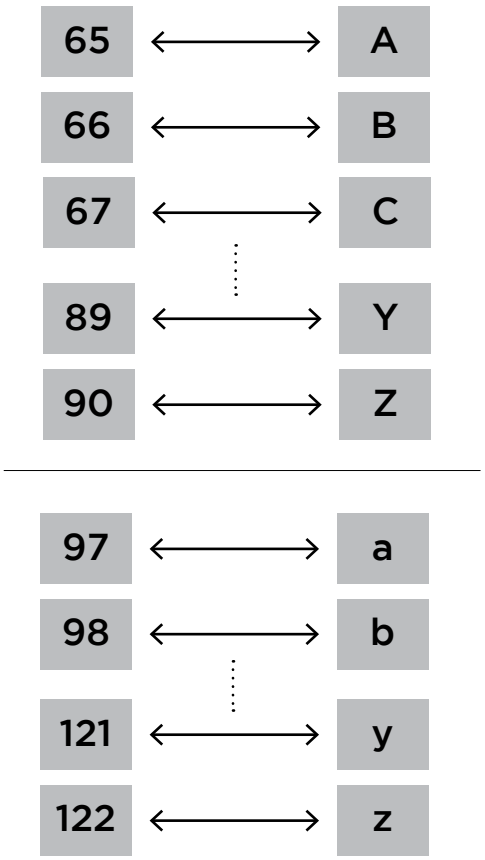
CS50 ASCII

Overview

Computers need a way of storing a variety of types of information, including text. However, since computers can only store data as 0s and 1s, computers need a way of using those 0s and 1s to represent characters in text. ASCII is a standard way of translating characters to and from sequences of binary digits that computers can understand.

Key Terms

- encoding
- ASCII
- ASCII table



ASCII Encoding Standard

In order to represent characters as numbers, a character **encoding** standard is used, which gives common characters a unique number to identify them. **ASCII** is a common encoding standard, which computers use in order to store text-based data. In the standard, the number 65 corresponds to the capital letter 'A'. Thus, if a computer wanted to store the capital letter 'A', it would need to store the number 65 in binary (which happens to be 1000001). The next 25 values in the ASCII encoding standard represent the other 25 letters, in order: so 66 represents 'B', 67 represents 'C', and so on.

Lowercase letters also have numerical representations in ASCII. The lowercase letter 'a' is represented by the number 97, 'b' is represented by 98, and so on. Thus, for a computer to store the lowercase letter 'a', it would need to store the number 97 in binary, which is 1100001. Note that this binary number differs from the binary representation of capital 'A' by just one bit: the value in the 32s place. This is because, in ASCII, lowercase letters are always represented as numbers 32 greater than their respective uppercase letters. As a result, letters can easily be switched from lowercase to uppercase or vice versa just by switching a single bit—the one in the 32s place—to 1 or 0 (1 for lowercase, 0 for uppercase).

There's no reason why ASCII has to use these exactly values: ultimately, the decision as to what number maps to which letter is arbitrary. What's important is that the standard is consistent: any computer can read and understand the numbers the same way.

ASCII's Limits

ASCII is frequently represented on an **ASCII table**: which is just a table that shows all possible ASCII characters, and which numbers correspond to them.

The original ASCII table represents all characters using just 7 bits: which means that there are 2^7, or 128, possible characters that can be represented in ASCII. Several extensions to ASCII exist which add an 8th bit, allowing for a total of 256 possible characters to be represented. Since there are only 52 letters, this means that ASCII has space to represent other types of characters: like punctuation, numbers, and some basic symbols (like the $ sign or the % sign).

However, event with 8-bit ASCII encoding, there are still a lot of characters that can't be represented, because there are more than 256 possible characters. For example, many mathematical symbols and characters in other languages do not fit into the standard ASCII table. As a result, other character encoding standards exist that have far more possible character options: Unicode, for example, is a character encoding standard that allows for more than 1 million possible characters to be represented. The first 128 characters in Unicode are identical to the 128 characters in ASCII, which makes them compatible with one another.