**01\_Converting to Binary.docx**

<http://www.youtube.com/watch?v=Erk9x8fmOaU&list=PL1DE477438120C9EF&index=5>

Let’s convert the number “93” to its binary equivalent.

128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255

These 4 bits form 1 byte of memory (RAM)

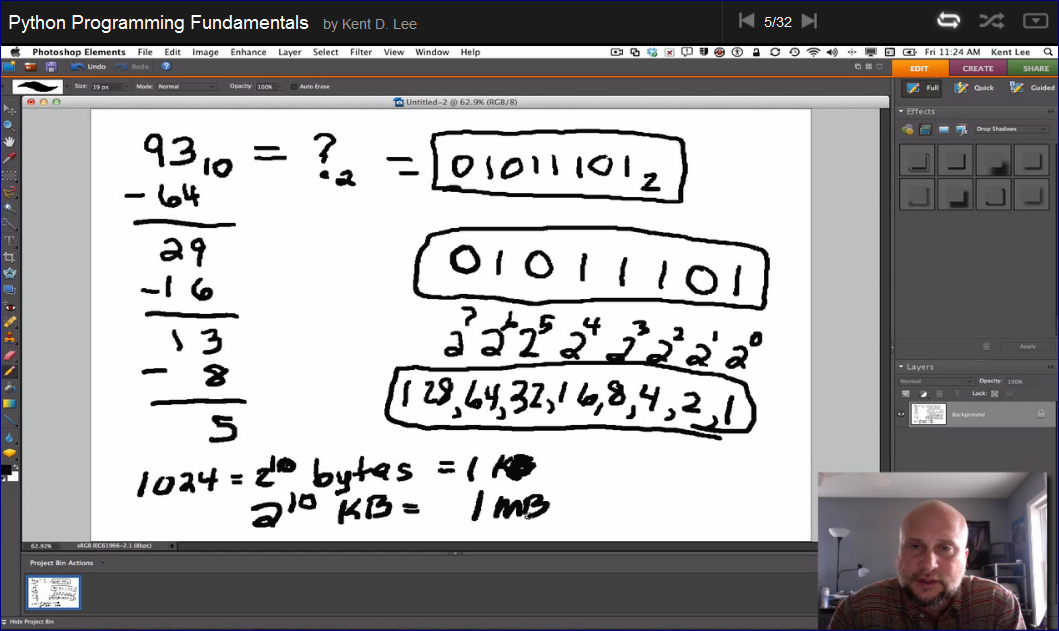
1byte can store the integer up to 255

In order to store bigger numbers 4 bytes are combined in one word (2^31)

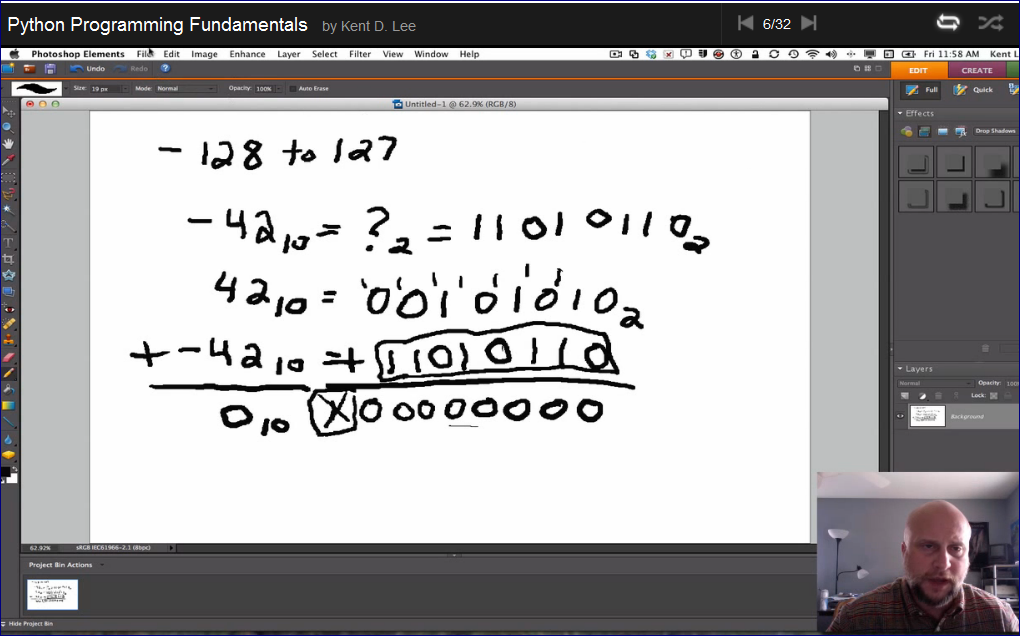
1024 (2^10) = 1 Kb

2^10 Kb = 1Mb

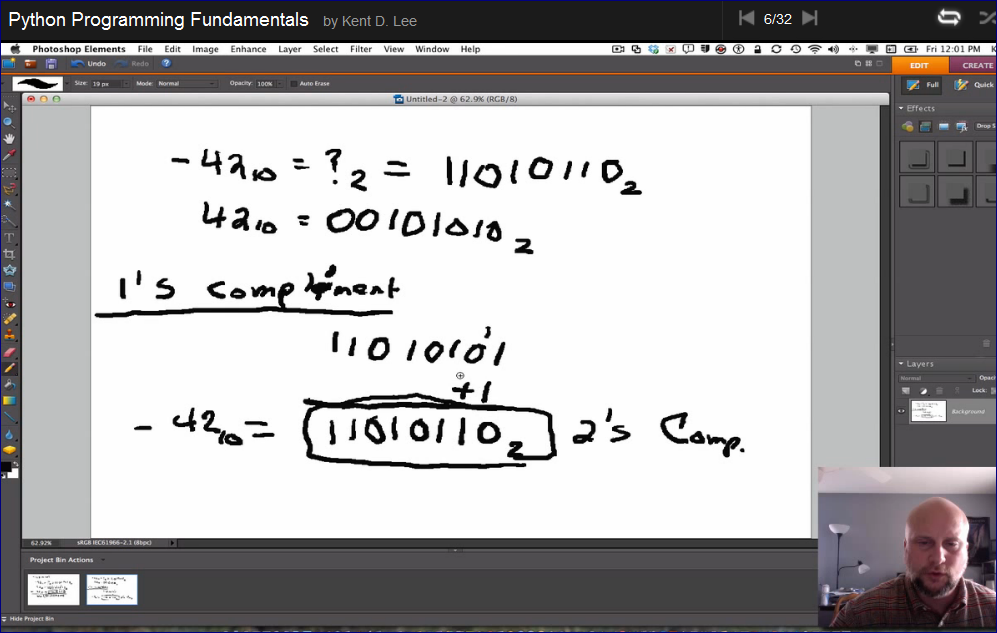
2^10 Mb = 1Gb



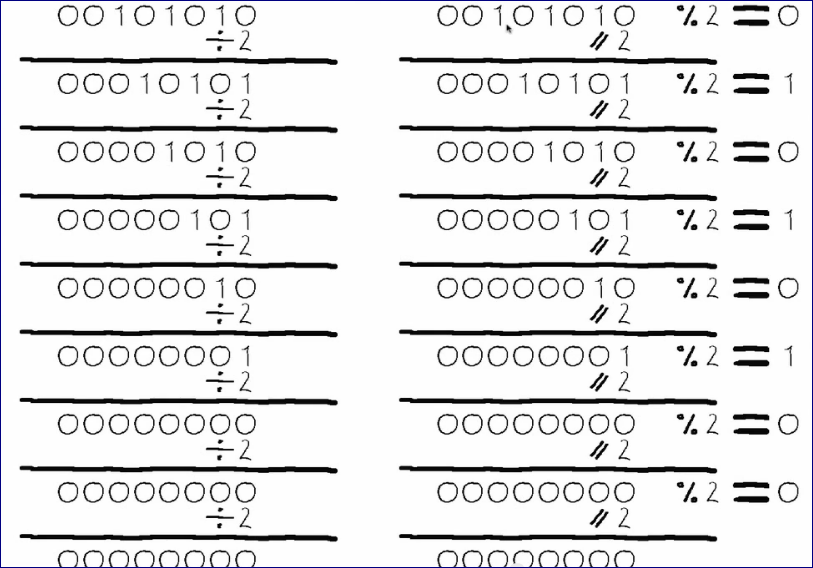
# 1-4: Signed and Unsigned Binary



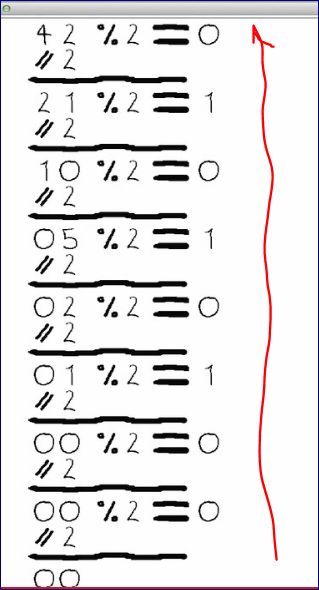
Another method to convert to a negative number



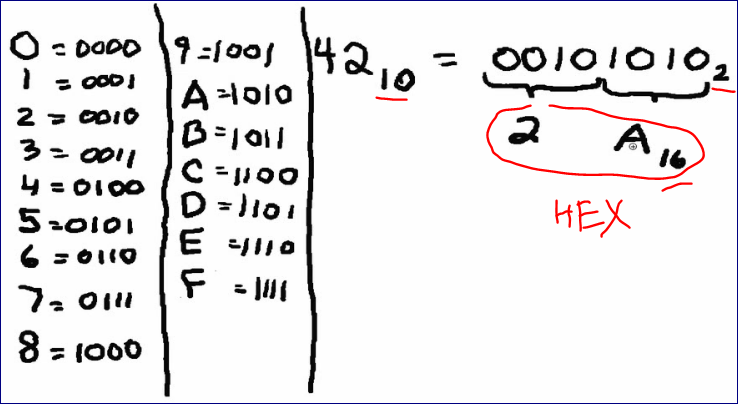
# 1-5: An Algorithm



Read the bottom up and that is the binary representation if “42”



HEXD – hex decimal number representation



# 1-6: The Importance of Types in Python Programs

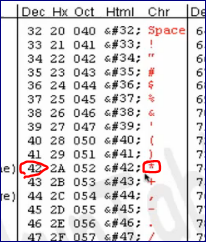
ASCII Table and Description

<http://www.asciitable.com/>

**ASCII** stands for **American Standard Code for Information Interchange**. Computers can only understand numbers, so an ASCII code is the numerical representation of a character such as 'a' or '@' or an action of some sort. ASCII was developed a long time ago and now the non-printing characters are rarely used for their original purpose. Below is the ASCII character table and this includes descriptions of the first 32 non-printing characters. ASCII was actually designed for use with teletypes and so the descriptions are somewhat obscure. If someone says they want your CV however in ASCII format, all this means is they want 'plain' text with no formatting such as tabs, bold or underscoring - the raw format that any computer can understand. This is usually so they can easily import the file into their own applications without issues. Notepad.exe creates ASCII text, or in MS Word you can save a file as 'text only

If we would be able to see in the bits of the computer memory we won’t be able to distinguish between

The number 42 and the character “\*”



They represent the same sequence of bits.



We tell Python how we want to interpret these bits:

* As an Integer
* As a character
* As a Real Number