How numbers, images and texts are represented in the computer

Christophe@pallier.org

Sept. 2015

Contents

There are 10 kinds of people: those who count in binary and the others

Computers represent all objects as series of 0 and 1, also known as bits (for "binary digits").

Representation of integers

Just like a number can be written in base 10, it can be written in base 2:

E.g.

```
12 : 10  + 2  = 8 + 4 = 2**3 + 2**1 : 1010
33 : 30  + 3  = 32 + 1 = 2**5 + 1 : 100001

1 :   1
2 :  10
3 :  11
```

```
4 : 100
5 : 101
6 : 110
7 : 111
...
```

To learn more about how integer numbers can be represented in binary format, see http://csunplugged.org/binary-numbers

(1) Convert (manually) into decimal the following binary numbers:

- 101
- 1000
- 1011
- 11111111

(2) Write a function that, given the binary representation of a number as a string of '0' and '1', returns its value as a integer.

(3) Now we will go in the other direction: Our aim is to write a program that, given a number (in decimal), computes its binary representation.

If you have an idea how to program it, please proceed. If not, we propose that you follow the following steps:

(4) Study the program below. Execute it with various values of the variable *num*. Do you understand the last line? Do you see a limitation of this program?

```
num = 143
d3 = int(num/1000) % 10 # thousands
d2 = int(num/100)  % 10 # hundreds
d1 = int(num/10)   % 10 # dec
d0 =  num % 10
print(str(d3) + str(d2) + str(d1) + str(d0))
```

(5) Adapt the above program to print the binary representation of num

(6) Modify the above program to print the binary representations of every number between 0 and 255.

(7) (Advanced) Write an improved version that uses a loop and does not have a limitation in size.

(8) Study the following code. Do you understand why it works?

```
def binary(num):
    if num == 0:
        return "0"
    if num == 1:
        return "1"
    return(binary(int(num /2)) + binary(num % 2))

print(binary(1234))
```

Remark: measures of memory size

- 1 byte = 8 bits
- 1 Kilobyte (KB) = 1024 bytes
- 1 Megabyte (MB) = 1024 kbytes = 1048576 bytes
- 1 Gigabytes (GB) = 1024 Mbyes
- Terabyte, Petabyte, Exabyte...

Exercice (advanced): Write a function that return the hexadecimal representation (base 16) of a number.

To go further:

- If you want to know how negative integer numbers are represented, see http://en.wikipedia.org/wiki/Two%27s_complement
- Execute 0.1 + 0.2 on the python command line. If you are surprised, read What Every Programmer Should Know About Floating-Point Arithmetic

---

Representation of text

A text file is nothing but a sequences of characters (a word document is not a text file).

For a long time, characters were encoded using ASCII code.

(9) lookup the ASCII representation of your first name in the table and use the chr function of Python to print it.

Remark: ASCII codes use one byte per characters. This is fine for English, but cannot cover all the caracters of all alphabets. It cannot even encode french accented letters. Unicode was invented that associate a unique 2 bytes number to each character of any human script. It is possible to write text files using these number, but more economic to encode the most common letters with one byte, and keep the compatibility with ASCII (UTF-8).

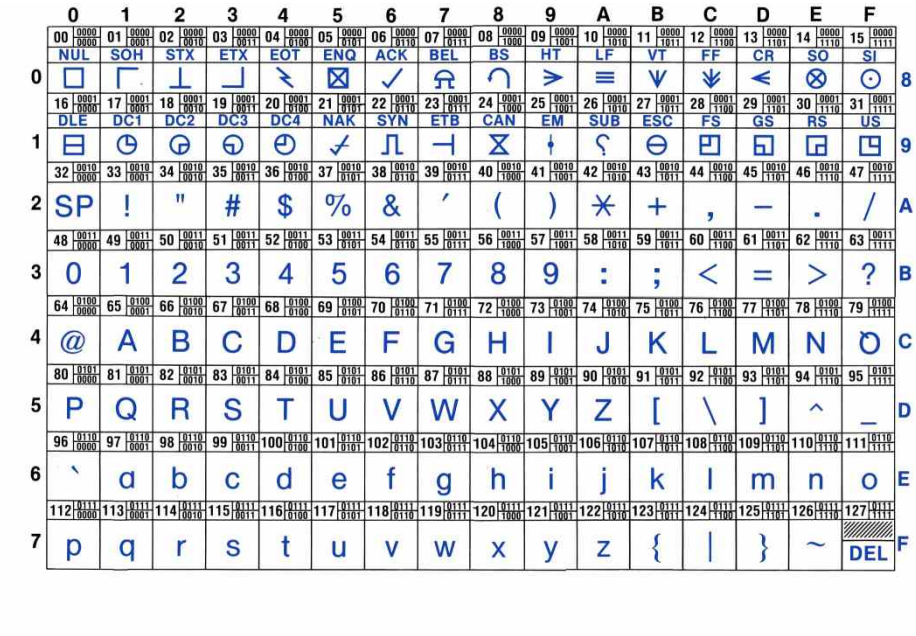


Figure 1: ascii table

How to read a text file in Python

Download Alice in Wonderland

```
f = file('alice.txt')
o = f.read()
print(o)
lines = o.split("\n")
print(lines)
```

(10) Write a program that counts the number of lines, and number of words in alice.txt (we suppose that words are separated by spaces).

(11) Write a program that detects if a text file contains the word 'NSA'

Representation of images

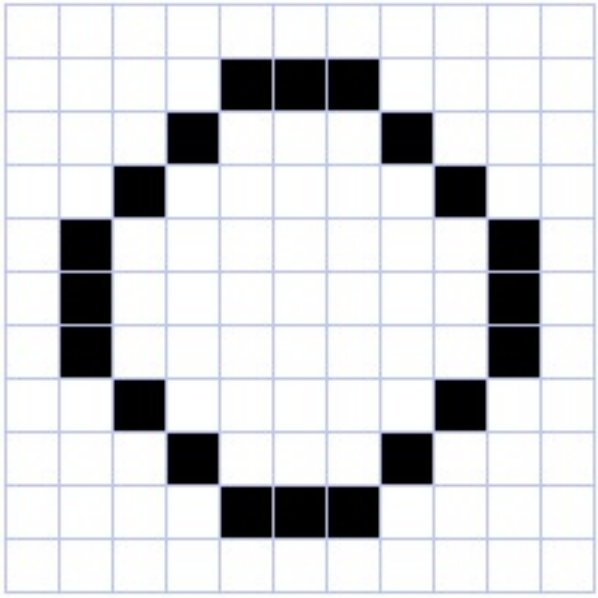
Images can be stored either:

- as bitmaps, that is a two dimensional arrays of dots (formats: bmp, png, gif, jpeg...)
- as vectorized formats, the image contain instruction for drawing objects (eps, pdf, svg, ...).

Here we are just going to manipulate bitmaps.

Black and white bitmaps

Each dot (pixel) is either '0' (black) or '1' (white).



(12) What is the size in kilobytes of a 1024x768 pixels images ?

(13) Execute the follwoing code (it requires the modules numpy and matplotlib).

```
import numpy as np
import matplotlib.pyplot as plt

a = np.array([[0, 0, 0, 0, 0, 0, 0],
              [0, 0, 1, 1, 1, 0, 0],
              [0, 0, 1, 1, 1, 0, 0],
              [0, 0, 1, 1, 1, 0, 0],
              [0, 0, 1, 1, 1, 0, 0],
              [0, 0, 1, 1, 1, 0, 0],
```

```
              [0, 0, 0, 0, 0, 0, 0]])
plt.imshow(a, cmap=plt.cm.gray, interpolation='nearest')
plt.show()
```

Numpy's arrays are a new type of object. There are similar to lists, but optimised for mathematical computations. Notably, they can be multidimensional (i.e. you can use a[i,j] notation). You can learn more about arrays in the documents http://www.pallier.org/cours/AIP2013/python4science.pdf and http://wiki.scipy.org/Tentative_NumPy_Tutorial.

(14) Exercice:

1. Create a cross.
2. create a 200x200 bitmap:
    1. add a diagonal
    2. make two crosses imitating the British Flag

Grey level pictures

Each dot is now associated to an integer value, e.g. ranging from 0 to 255 for 8-bits codes, coding for a grey level (smaller=darker). Each dot needs one byte.

How large is the file for a 1024x768 image pixels with 256 grey levels?

The following code displays an image:

```
import scipy.misc
l = scipy.misc.lena()
plt.imshow(l,  cmap=plt.cm.gray)
plt.show()
```

This code runs a low pass (averaging) filter on it:

```
import scipy.ndimage
bl = scipy.ndimage.gaussian_filter(l, 3)
plt.imshow(bl,  cmap=plt.cm.gray)
plt.show()
```

Edge detector. It is easy to implement an edge detector with a neural network. See https://courses.cit.cornell.edu/bionb2220/UnderstandingLateralInhibition.html.

Using the ndimage.convolve function, apply the following filters to the image and diplay the results.

```
kernel1 = np.array([[-1, -1, -1],
                    [-1,  8, -1],
                    [-1, -1, -1]])


kernel2 = np.array([[-1, -1, -1, -1, -1],
                    [-1,  1,  2,  1, -1],
                    [-1,  2,  4,  2, -1],
                    [-1,  1,  2,  1, -1],
                    [-1, -1, -1, -1, -1]])
```

More manipulations are available at http://scipy-lectures.github.io/advanced/image_processing/.

Colored bitmaps

Each dot is now associated to three bytes, representing the Red, Gree and Blue intensities (see http://www.colorpicker.com/).

How large is the file for a 1024x768 RGB image?

Exercice: What are the RGB triplets for BLACK, WHITE, RED, YELLOW?