W303 – Information representation

Digitizing information → representing it with discrete symbols.

Any set of discrete symbols will do

Examples

- dice
- Latin alphabet
- Greek alphabet
- bits "0" & "1"

etc...

- symbols A & B
- crazy symbols like ☺ and ☹ (smileys)

...

We are concerned with how information is represented in computers. Because how computers are built, we can only use 2 symbols. We usually write these symbols as 1 and 0, but this is just a convention.

→ a computer contains millions of devices that can be in one of 2 states

(ex) – a relay → up / down a switch → open / closed

mechanical computers like Konrad Zuse's.

- a vacuum tube → lets current to pass
  ↘ does not let current pas
  } generation I

- a transistor (same function as vacuum tube)

⋮

2 states <=> 2 symbols can be stored & manipulated.

Task: representing all kinds of info using only 2 symbols

Info
- English text
- Image
- Drawing
- Sound
- numbers

→ represent as: 1 0 1 0 1 1 .... (sequence of 2 symbols.)

→ will be studied when we look @ multimedia

Example
- use dice (6 symbols) which we can write as 1 2 3 4 5 6.

&

- represent English sentences.

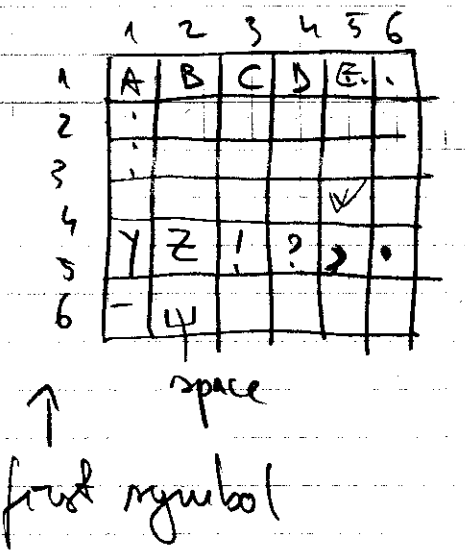
a) A → 1 B → 2 C → 3 D → 4 E → 5 F → 6

Not enough.

b) Use pairs of dices.

A → 1 1 B → 1 2 C → 1 3 - - - - etc.

Pairs = $6^2 = 36$, enough to represent all 26 letters & perhaps . , ; — ! ? , etc...



With this table, I can represent Hello world! using dice like this

2 2 1 5 2 6 2 6 3 3 6 2 4 5 3 3 3 6 2 6 1 4 5 3

L space

(we represented an English sentence with 6 symbols; a long sequence of symbols.

- What we can accomplish with 6 symbols, we can accomplish with 2, just the sequence (of 1s0) will

be even longer. All we need is just agree on a table like the one above & we can encode & decode text.

ASCII - table (standard)

(American Standard Code for information interchange)

- version 1 : each alphabet letter → sequence of 7 symbols. This gives 2^7 different sequences (128 alphabet characters)

  (2^7: 2 = # symbols, 7 = length of sequence)

- version 2 : → sequences of 8 bits (byte) (256 different characters)

- This approach = insufficient to deal with all languages & scripts in the world.

- UNICODE → standard meant to allow representations of any possible script of a living, dead, or imaginary language (like Klingon)

How it is done?

A) An index of all possible characters → (each receives a unique ID)

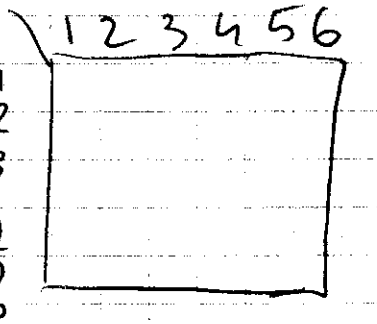
B) A way of translating each ID into bits ...

... called an encoding.

(ex) UTF-8 → encoding used for almost all e-mail, & most web pages.

- All the Latin alphabet encoded with 8 bits (one byte) like Ascii
- All other characters are encoded using a longer sequence of bits. The idea is to use an "ESCAPE" character.

(idea)



dice

- 2 dice symbols = $= 6^2 = 36$ characters to be translated.

- if I need both lowercase & uppercase Latin alphabet?

52 characters are too many for encoding with 2 dice.

- sol a) 3 dice symbols. (sequences of dice get longer for same text)
  b) 2 dice but also 2 tables with different encodings.

Table 1

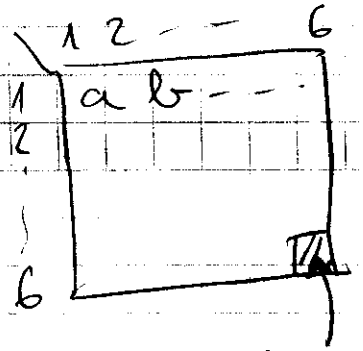
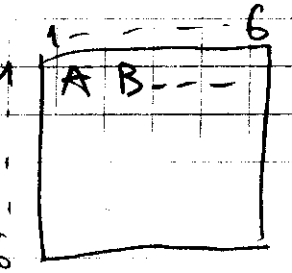


Table 2



call this a special escape character ESC

Convention:

- encode or decode as usually using table 1 except when you need to encode a capital letter. To encode Capital, use 4 dice symbols. First pair is 66 (ESC) that says next pair is encoded/decoded using table II.

Capital A is 6611

Lowercase a is 11