A brief introduction to computer science, to help you out

Orena GAPUISSI

Table of contents

What is cryptography?

Cryptography is a branch of cryptology and is the art of **protecting messages** (ensuring confidentiality, authenticity and integrity), often by using *secrets* and *keys*. It is different from steganography, which involves hiding one message within another, in that cryptography **renders a message unintelligible** to anyone other than the intended recipient.

It has been used since ancient times, but some of the main methods, such as asymmetric cryptography, have only been around since the late 20th century.

(Source: Wikipedia)

An example of a symmetric cipher: the Vigenère cipher

The **Vigenère cipher** is a method of encrypting text by substitution, whereby a single letter of the message to be transmitted (**plaintext**), can, based on its position in the latter, be replaced by different letters in the **codetext**.

The Vigenère cipher was broken by the Prussian major Friedrich Kasiski, who published his method in 1863. Since then, it no longer offers any security, but its methodology continues to inform modern, key-based encryption techniques.

Key cryptography uses a chain of digits or characters (**the key**) to indicate the substitute letter for each letter in the plaintext. Modern methods use unique, random alphanumeric keys.

In the **Vigenère cipher**, the key is a word that is repeated several times (this is its inherent weakness and what makes it easy to develop a decoding method). Each letter in the plaintext corresponds to a letter in the key, which determines the substitution. Coding is done with the help of a table, which indicates, in the form of a cross tabulation, the letter to be used for coding.

Blaise de Vigenère
Wikimedia commons

(Adapted from Wikipedia)

The binary system – why all those 0s and 1s?

Binary arithmetic (or simply, binary calculation) is used by everyday electronic systems (calculators, computers, etc.), in which the two digits 0 and 1 are translated into either **a flow or absence of electric current** (or into a **voltage** difference).

Generally, the 1 represents the flow of current, while the 0 represents the absence of current.

In **logic**, the 1 represents “true” and the 0, “false”.

Just for my own knowledge, regarding the origins of binary:

In December 1937, physicist John Atanasoff, fed up with performing endless calculations, jumped into his car and headed for Des Moines, the state capital of Iowa in the United States. In the early afternoon, he stopped at a bar

*on the Mississippi border, where he downed a few scotches and, suddenly, it came to him: "**1 and 0** – **binary**. That's the answer." And right there and then, on a napkin on the table in front of him, he wrote out the basis of the key characteristics of the digital computers that would take over from mechanical machines in the decades that followed: "They'll work on the basis of 0s and 1s. They'll have a memory and carry out logical operations." With the help of his assistant Clifford Berry, he built the first ever digital computer.*

John Atanasoff had been a human calculator, or computer. So, he called his machine a "computer" and, in late 1939, the ABC (Atanasoff Berry Computer) entered into service: it performed a calculation every 15 seconds and weighed more than 300 kilos.

(Loosely adapted from owni.fr)

Images and encryption:

What is a bit?

A **bit** (**bi**nary digi**t**) is the most basic unit in a numbering system that uses only two symbols, 0 and 1.

What is a digital image?

The term "**digital image**" means any image (picture, icon, photograph, etc.) *acquired*, *created*, *processed* and *stored* in binary form.

(Wikipedia)

Visually, a digital image appears as a table of pixels of a certain **width (w)** and **height (h)**: **w x h**

Though they are always stored in binary form, images can be **encoded in different ways** for digital storage. The methods differ in terms of the amount of space saved and the quality of the base image retained.

What is a pixel?

A **pixel** is the base unit of measurement of the definition of a digital image. Its name is a portmanteau of *pix* (from picture) and *el* (from element).

A digital image's **definition** is the number of pixels it comprises on the horizontal and vertical axes.

Its **resolution** is the number of pixels per inch, expressed in ppi or dpi (dots per inch). The higher the resolution, the less visible the pixels in the image.

What about my screen?

In computer science, a pixel is encoded on one or more bits:

In a black and white image, each pixel is either black or white, i.e. encoded on one bit (0 or 1).

Just for my own knowledge:

In a grayscale image, with just four possibilities – black, dark grey, light grey and white – one pixel is encoded on two bits.

In general, in an image with various shades of grey, each pixel is encoded on 8 bits (1 byte), so $2^8 = 256$ levels of grey.

In a colour image, each pixel comprises three sub-pixels (RGB: red-green-blue), each of which can be encoded on 8 bits, so 256^3, giving more than 16 million colours.

Sub-pixels in the RGB model
Wikimedia commons

What does a computer consist of?

- **Processor**: the computer's **brain**. It processes information and carries out instructions. The more powerful it is, the quicker it can perform **calculations**.

- **RAM** (or **live memory**): this is a storage area in the computer where **data are stored temporarily** while you work. It is the **short-term memory**. The more RAM the computer has, the better it performs. Storage capacity is expressed in bytes.

- **Hard disk**: the hard disk is a **storage area** and one of the principal components of a computer. It is the **long-term memory**, where your software, photos, videos and other types of computer data are stored. It is possible to attach an external hard drive to a computer, to use in addition to the existing internal one.

- **Motherboard**: this is the element that **connects all of the other computer elements** (processor, memories, input and output devices, etc.) and enables them to **communicate** with each other.

- **Ribbon cables**: these **link** the various **components** to the **motherboard** to ensure reliable and rapid **communication** within the computer.

- **Graphics card**: this enables all **graphical elements**, e.g. photos, videos, games, etc. to be displayed by producing an **image on a screen**.

- **Sound card**: this **relays sound** from within a computer and **receives sound** from outside of it.

- **Network card**: this is a set of components usually integrated into the motherboard that allows the computer to **connect to a network** in order to **share data**.

- **Power pack**: this supplies the **electric current** to all **parts** of the computer.

- Examples of **input devices**:

  -the **keyboard**: allows you to write.

  -the **mouse**: allows you to move a cursor around the screen.

  -the **microphone**: allows you to record an audio source.

  -the **webcam**: allows you to record live action.

- Examples of **output devices**:

  -the **screen**: allows you to visualise data from the computer.

  -the **printer**: allows you to produce text, images, etc. from the computer on various materials.

  -**loudspeakers**: allow you to broadcast sounds from the computer.

- Examples of **input/output devices**:

  -**disk drive**: allows you to **read** (and, in some cases, burn/**write**) **digital data** on an optical disk (CD, DVD). Even when the computer is switched off, it can be **opened mechanically** by inserting a **paper clip** into a nearby hole provided for this reason.

  -**floppy-disk drive**: like the disk drive, the floppy-disk drive allows you to read and **write** (save) **digital data** on disks.

  -**USB drive** (or external hard drive): allows you to **save** and **read digital data**.

- Computer **connections**:

  -**USB ports**: allow **peripheral devices** to be easily **connected** to the computer.

  -**VGA connector**: allows a **screen** to be **connected** to a computer via a graphics card.

  -**microphone socket**: allows a **microphone** to be **connected** to the computer via the **sound card**.

  -**audio socket**: allows **loudspeakers**, **headphones** or a **headset** to be **connected** to the computer via the **sound card**.

(Loosely adapted from pratique.leparisien.fr)

<u>*Just for my own knowledge:*</u>

This is what a floppy disk looks like and, yes, I can assure you, such a thing did exist!

Key:

1. *Capacity indication (in this case, large)*
2. *Hub (disk drive)*
3. *Shutter (movable protective cover)*
4. *Plastic housing*
5. *Paper/soft ring*
6. *Magnetic disk*
7. *Disk sector*

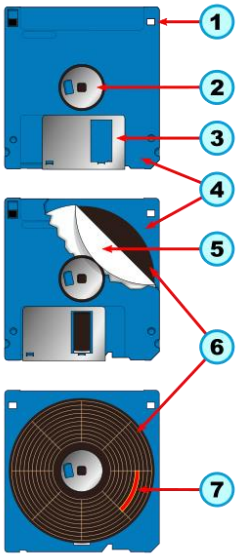


Diagram of a floppy disk
Wikimedia commons

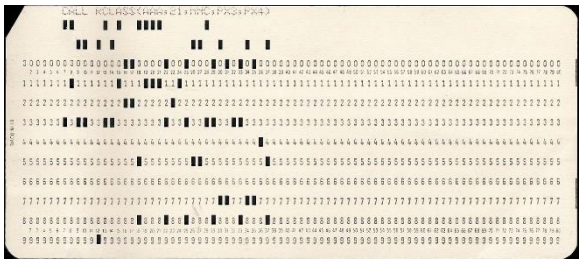
Databases:

A **database** facilitates the storage and retrieval of complete sets of raw **data** or information in relation to a particular subject or activity. There are different types and they can be connected, to a certain extent.

The earliest databases were based on the introduction of **punch cards**, which were divided into fixed-width lines and columns.

A **punch card** is piece of stiff paper that contains data represented by the presence or absence of holes in predefined positions.

The original **mechanism for reading** the punch cards was rather peculiar. Basically, it involved passing a pin over the lines and columns on the card. Underneath the card there was a tiny cup of mercury. If the pin touched the mercury, an electric current passed through the pin and closed the circuit, thus indicating the presence of a hole.



Punch card with 80 columns and 10 lines
Wikimedia commons

<u>Just for my own knowledge:</u>

The cards were perforated by specialised operators working from "entry dockets". The cards were then able to be sorted on machines called sorters and collators. The data machines used these cards up until such machines were replaced by computers, around 1970. Up until the early 1980s, computers were equipped with peripheral devices capable of reading and punching these cards.

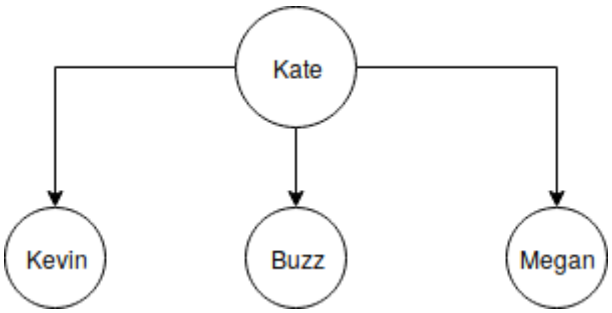
Punch cards? A simplified example!

In simplified terms, imagine a system in which **each card** represents **an element** and all of the cards are **perforated** to cover every possible **feature**. However, depending on the **value** of the element, the hole would not be the same for each **feature**. If the element has the desired feature, the hole is "**open**", i.e. connected to the edge of the card, and if it does not, the hole is "**closed**". You could then use **needles** to make a particular **request**: if you pass a needle through the hole corresponding to the test feature for the deck of cards and then lift it up, only the cards where the element has the requested feature will fall (the others will remain hanging on the needle).

Example of how this system works demonstrated by Marie Duflot-Kremer: to determine a dish to serve at a formal meal, each card is a guest and each feature is a dish that the guest in question likes (open hole) or dislikes (closed hole)

Trees

A tree is a **data structure** for **classifying** data in order of priority. For example, in the tree opposite, Kate is the mother of Kevin, Buzz and Megan.



In "Home Alone", Kate is the mother of Kevin, Buzz and Megan

***Data structures** are a way of storing and organising data to make them easier to store, use and modify.*

Tree features:

-a **node** is a basic element of a tree. In the tree opposite, "Kate", "Kevin", "Buzz" and "Megan" are nodes. Each node has a single **parent**, apart from the root ("Kate", in this case).

-the **root** is a node that does not have a parent.

-if node P is the parent of node E, we can therefore say that E is the **child** of P.

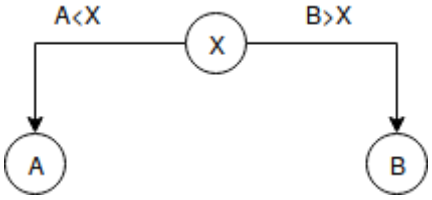
-a **leaf** is an element of the tree with no child. In our example, "Kevin", "Buzz" and "Megan" are leaves.

-a **sub-tree** is a part of a tree. Its root can be any node other than the root of the tree.

A binary search tree? What's that?

In a binary tree, **each node** has, **at most**, **two children**: a child on the left and a child on the right.

In a binary search tree, the child on the **left** of a node (and every element of a left sub-tree) is **smaller** than this node. The child on the **right** of a node (and every element of a right sub-tree) is **bigger** than this node.

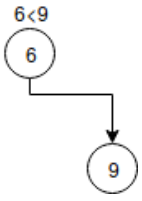


Principle of a binary search tree

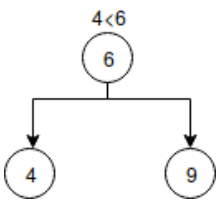
Example of how to construct a binary search tree from the ordered list of numbers 6, 9, 4, 5, 1:

Step 1:

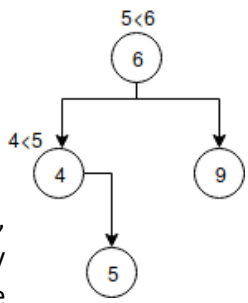
Step 2:



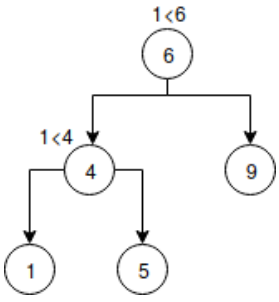
Step 3:



Step 4:



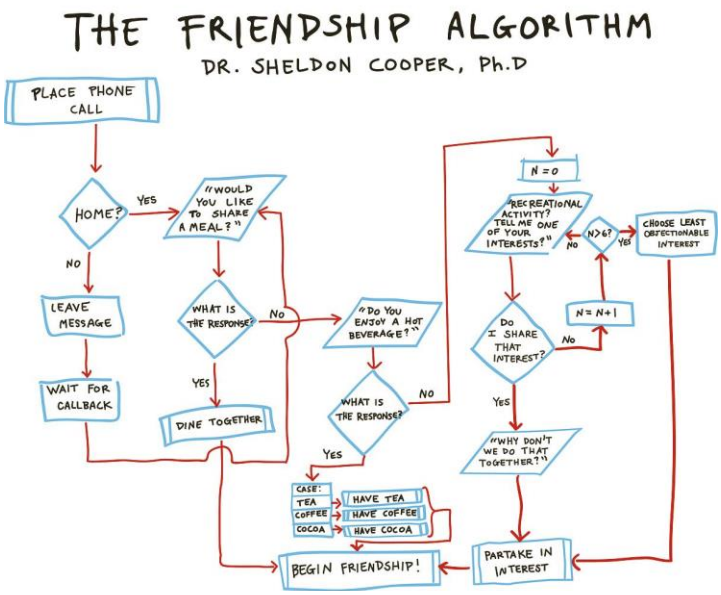
Step 5:



By way of **example**: in the final tree arrived at in step 5, nodes 1, 4 and 5 form a **sub-tree** of the tree, and every element of this sub-tree is **smaller** than the **root** of the tree (6, in this case), because the sub-tree is to the left of the root.

Algorithms

An algorithm is a **finite sequence of instructions** drafted in a **non-ambiguous** way with the aim of **solving a problem**. The processor, for example, takes instructions that have been entered and executes them – that's how the computer works. An **algorithm** developed in pseudocode can be **programmed** to form a **sequence of instructions** drafted in a language that the computer, after compilation, can **interpret** and thus execute. The word algorithm is derived from the name of the mathematician Al Khwarizmi (circa 780-850), who introduced the decimal numeral system and related arithmetic. Algorithms are studied for the purpose of **reducing** their **complexity in time and space** – in other words, so they can be **optimised**.



Algorithm using conditions and loops
The Big Bang Theory

A loop is a **logical structure** whereby an instruction or set of instructions is **executed several times**. There are **two types** of loop: the "**for**" loop, which enables an instruction to be repeated **a certain number of times**, and the "**while**" loop, which enables an instruction to be repeated as long as a certain **condition** is **met**, e.g. "while X<10". A complete passage through a loop is called an **iteration**.

A condition is a **control construct** that **executes** an instruction or set of instructions **only** if a certain **condition** is **met**.

Just for my own knowledge, regarding some concepts relating to algorithms:

***Pseudocode** is an informal way of writing an algorithm. It uses the structural conventions of algorithms and a language that is similar to everyday language to express the essence of the instructions required to solve the problem.*

***Compilation** is the transformation of a programme written in programming language that can be read by a human into a binary code that can be interpreted and executed by a machine.*

***Recursive algorithm**: an algorithm is said to be recursive if it invokes itself. The principle of recursion describes the steps involved in solving a problem using the solutions to smaller instances of the same problem.*

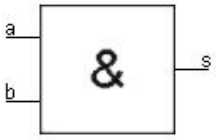
***The divide-and-conquer paradigm** works mainly by breaking the problem down into a number of sub-problems, then solving each of these sub-problems individually before finally combining them, i.e. working out a solution to the initial problem. This technique produces very effective algorithms. It is used to solve sorting problems, in binary search and, in general, to solve problems that are highly complex in terms of time and space.*

(Source: https://www.lamsade.dauphine.fr/~mayag/Chapitre_1_Introduction_Algorithmique.pdf)

Logic gates

"**Logic gates**" are one of the key devices in digital electronics. They are **basic electronic elements** involving one or more inputs that perform a calculation based on these inputs. The result is sent via the single output. These calculations are based on a type of **logic known as Boolean**, named after its creator, George Boole.

Diagrammatically, logic gates look like this:



- The letters **a and b denote the input variables**, but there may be just one or several inputs. The value of the inputs will be 0 (no electrical signal) or 1 (electrical current flowing).
- The **square** represents the **electronic component**. The **symbol** in the centre of the square corresponds to the calculation made by the component. Here, that calculation is "AND": s has the value 1 only if a=1 AND b=1.
- The letter **s is the output result**, with a value of 0 or 1.
- On the output line, there may be a **small circle attached to the square**: this indicates an inverted output value (called NOT + the name of the calculation). If, for an AND, you have a=1 and b=1, then s will be equal to 1; for NAND, s is inverted and has a value of 0.

Truth tables are generally used to show how each set of input values relates to its corresponding output values.

Just for my own knowledge:

*Let's take a **practical example**: a light switch. If the current is flowing, the light will go on – if not, it will remain off. Let's imagine that the activation of the light is controlled by an electronic system and that there are two buttons that have to be activated at the same time in order to switch on the light. These two buttons are connected to an "AND" logic gate and this gate only allows current to flow (value of 1) to the lamp if both buttons are pressed at the same time. If only one button is pressed, the "AND" logic gate will retain an output value of 0, i.e. there will be no current, so the light will remain off.*

IP addresses and routing

On a local network (e.g. my college's network) or on the **internet**, my computer (as well as other digital devices, such as my phone) is **recognised** by other devices by a **so-called IP address** (internet protocol), just like where I live has its own postal address. This IP address **consists of 4 numbers in the range 0 to 255**, separated by full stops: "X.Y.Z.W". (For example: 190.170.10.254).

When I access the internet, my computer has to **connect to servers**, which host my favourite website, for example, or store my emails. These, too, have IP addresses. For the connection to be made, **"packets" of data will be sent to a destination** (the site server). To reach the destination, t**he packets are transferred progressively from one computing unit to the next, by what are known as routers**, which also have an IP address.

Each router has a **routing table**, which **identifies**, for each address it knows, the **address of the next router** to which the message is to be transferred. So, the packet doesn't always go via the shortest or quickest route but is redirected as necessary, according to what each routing table indicates.

Just for my own knowledge:

*A **server** is a device that provides services, such as the use of an email inbox or access to a website. A computer has to connect to it in order to access its services.*
*When a piece of data is sent via a network, it is divided up into several **packets**, which are then reassembled once they've arrived at their destination.*
*A **router** is an item of computer hardware that facilitates the transmission of data so that it arrives at its destination address. It receives the packet of data and sends it to another address – for example, to another router – which, in turn, sends the data to another address, and so on, until the destination is reached.*
*It uses what is known as a **routing table** to identify the next address to which it must send the data. The routing table indicates, for each known destination address, the address of the next element to which the data must be sent in order for it to reach its intended destination.*