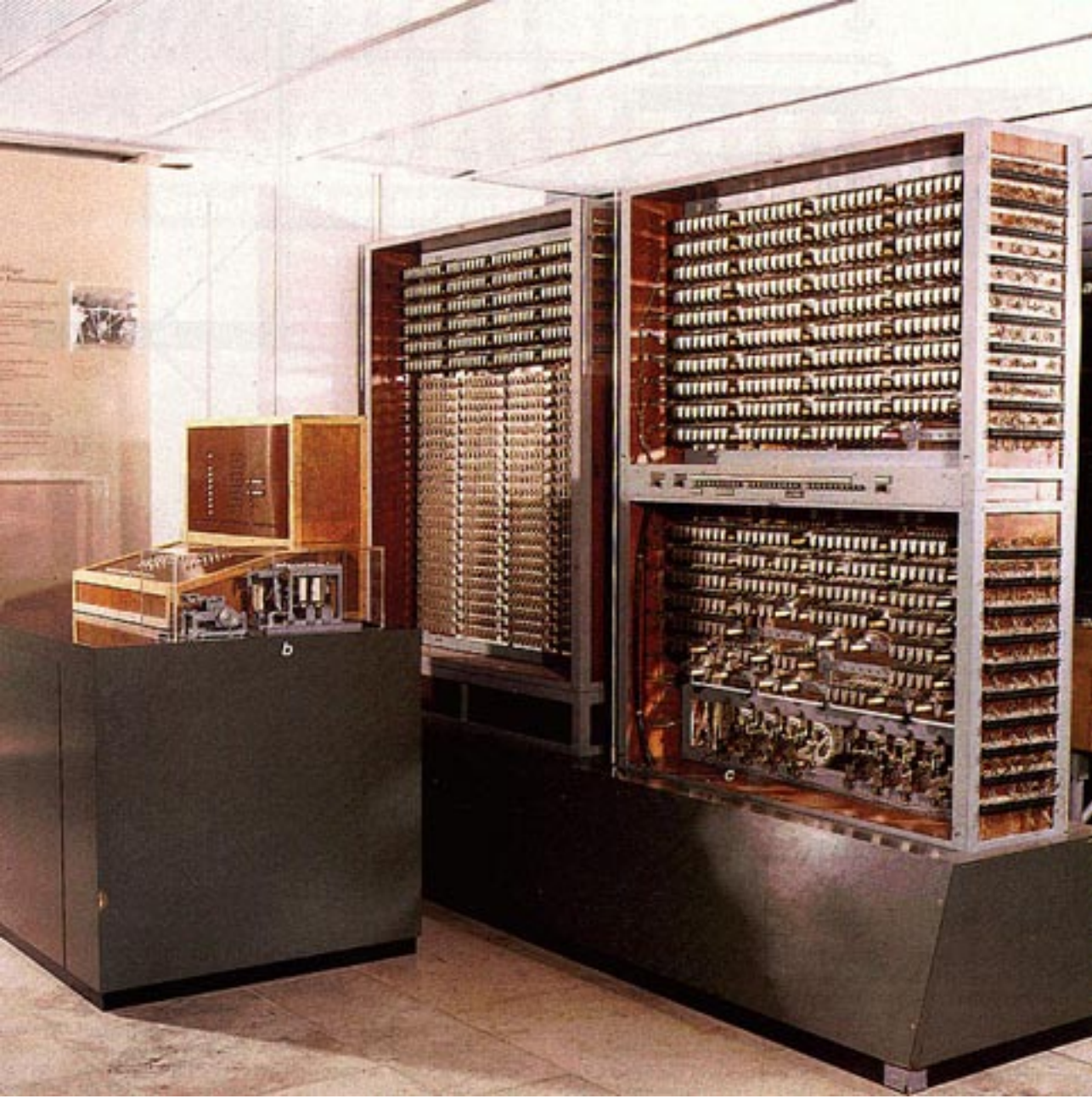
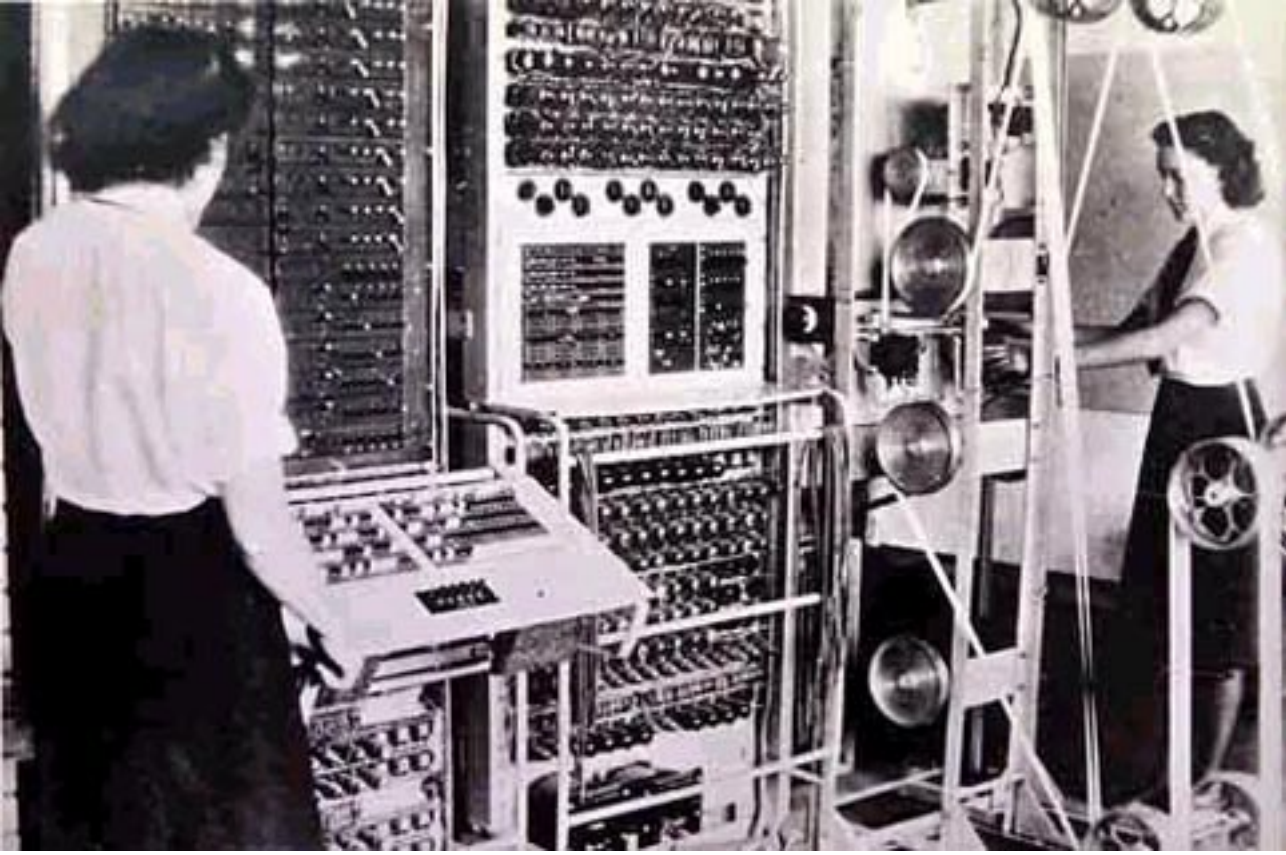
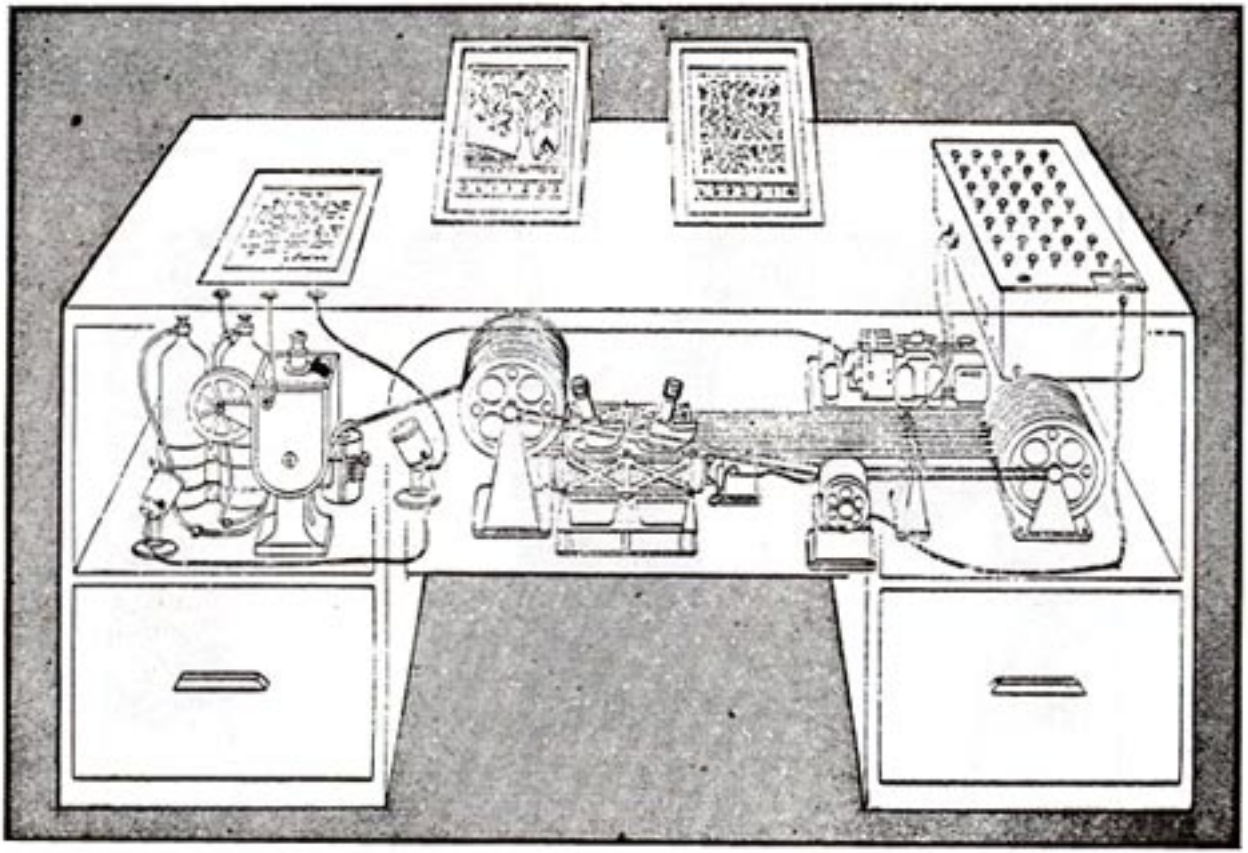
Fonctionnement d’un ordinateur.



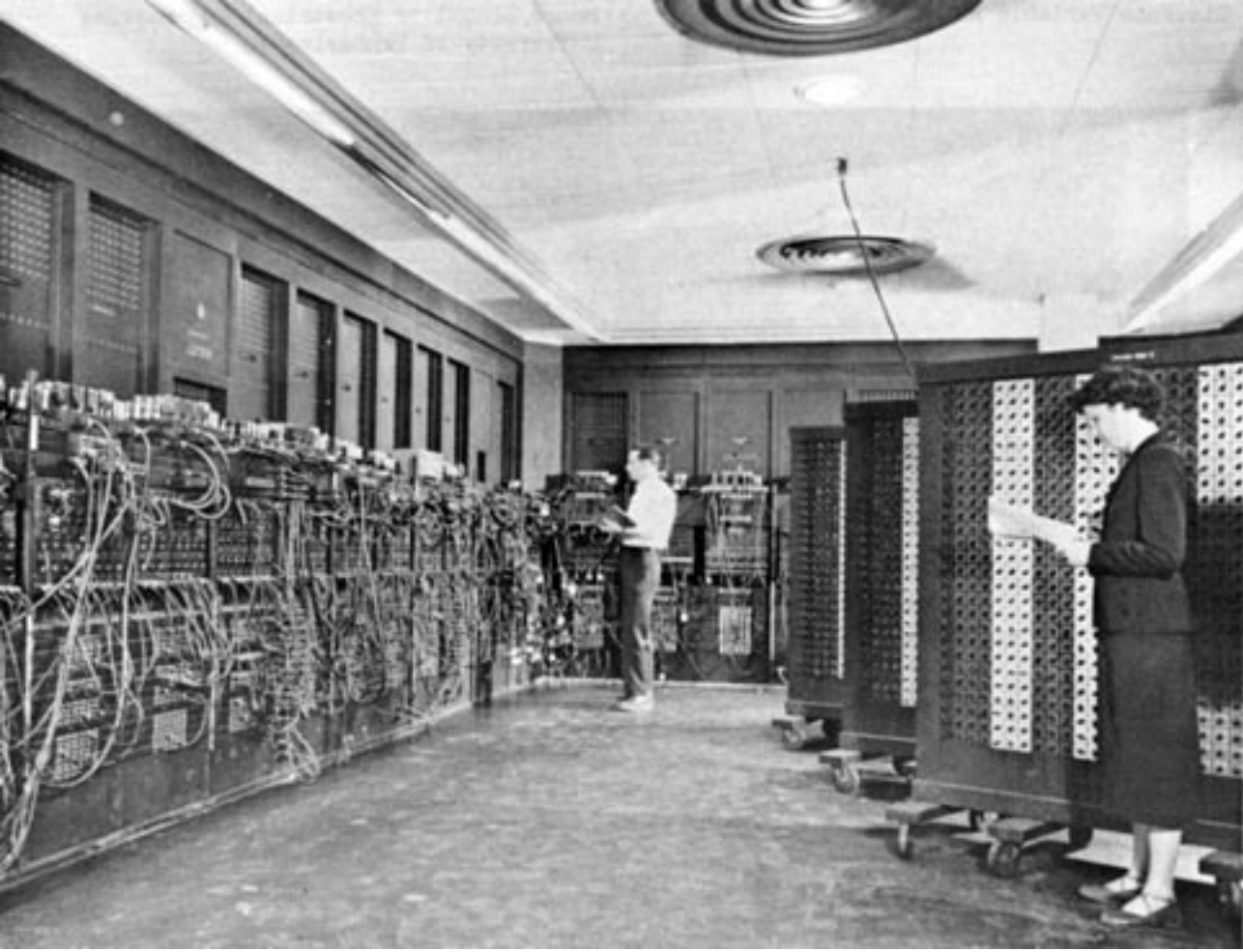
Konrad Zuse : Z3 (1941).
Premier ordinateur programmable mécaniquement
(ici une reconstruction vers 1960).
http://irb.cs.tu-berlin.de/~zuse/Konrad_Zuse/en/Rechner_Z3.html



Tommy Flowers : Colossus (1944).
Premier ordinateur électronique programmable.
http://en.wikipedia.org/wiki/Colossus_computer

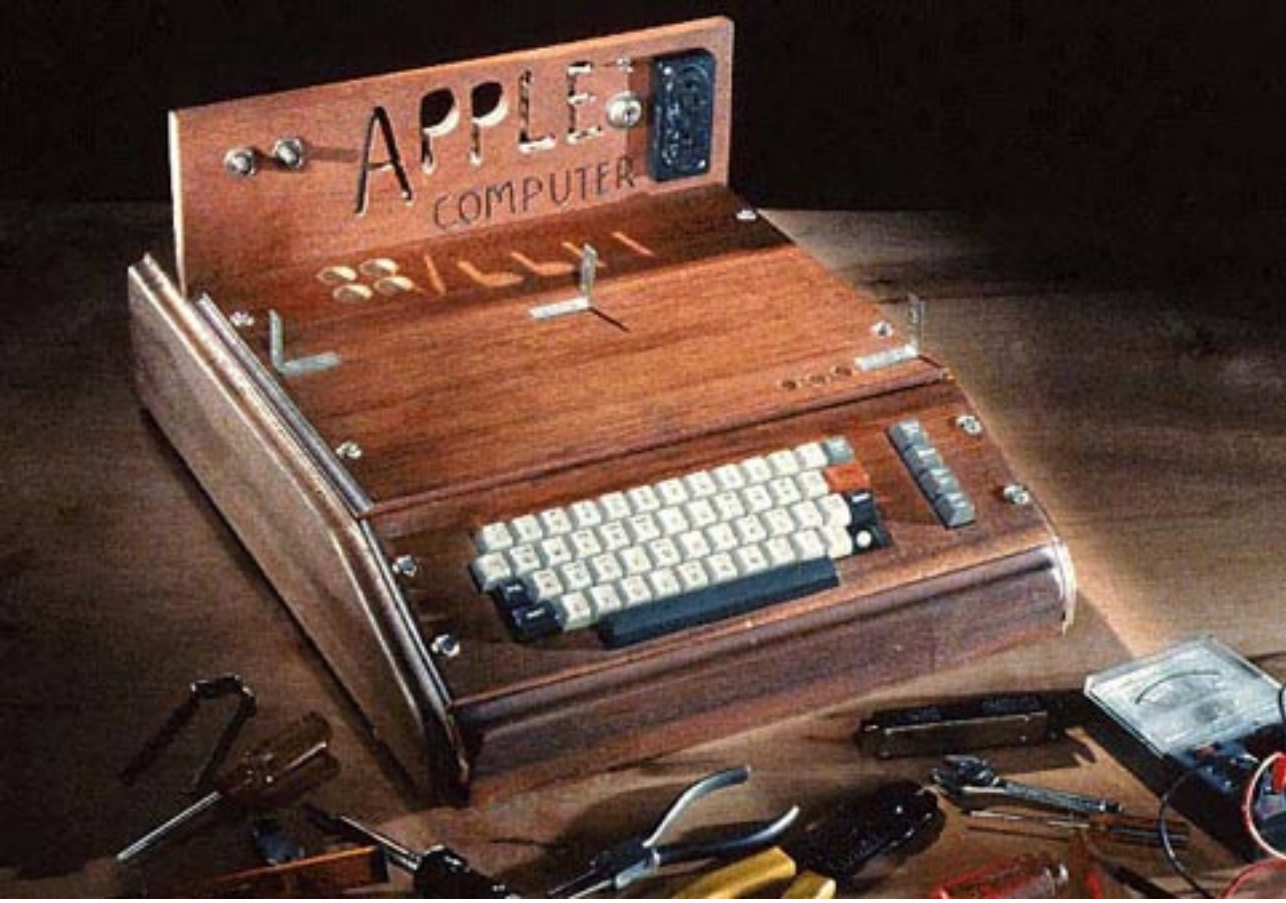


Vannevar Bush : Memex (Memory extender) (1945).
Ordinateur analogique théorique et visionnaire.
http://en.wikipedia.org/wiki/Memex



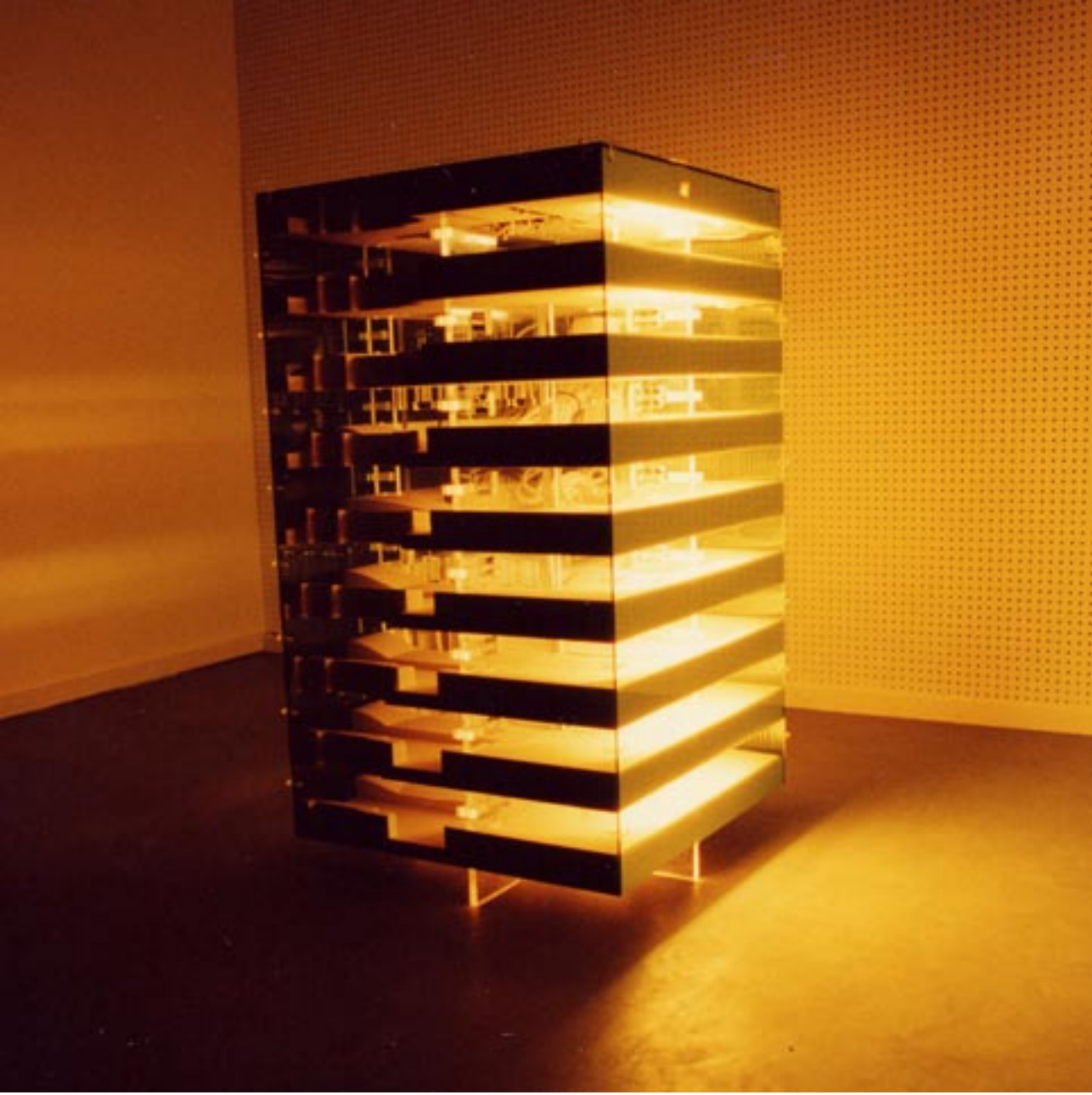
John William Mauchly & J. Presper Eckert : ENIAC
(Electronic Numerical Integrator And Computer) (1946).
Premier ordinateur entièrement électronique.
http://fr.wikipedia.org/wiki/ENIAC

John William Mauchly & J. Presper Eckert : UNIVAC I
(UNIVersal Automatic Computer I) (1951).
Premier ordinateur commercial.
http://fr.wikipedia.org/wiki/UNIVAC_I



Steve Jobs et Steve Wozniak : Apple I (1976).
Premier ordinateur individuel à être conçu
pour être combiné à un clavier et à un moniteur.
http://fr.wikipedia.org/wiki/Apple_I

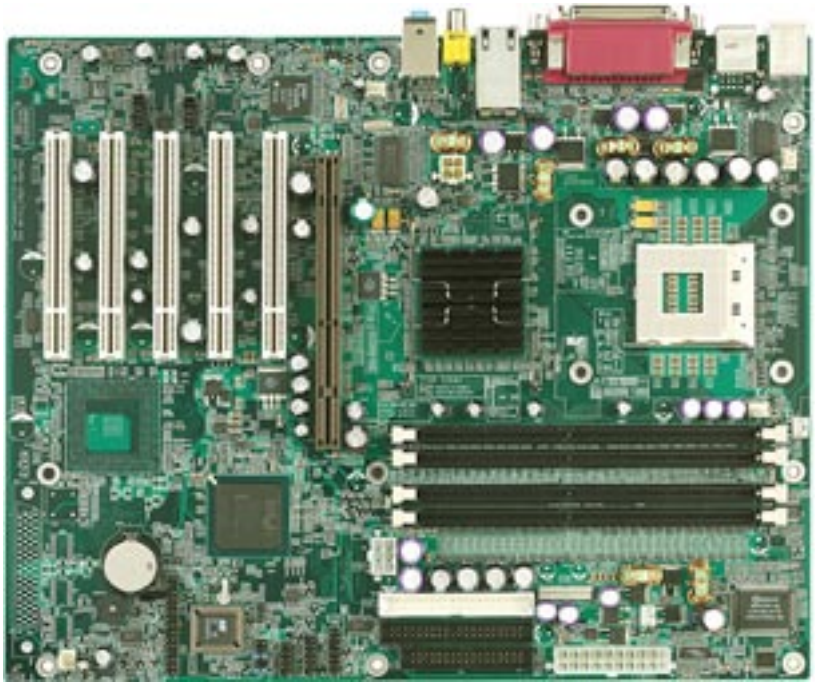
Steve Jobs et Steve Wozniak : Apple II (1977).
http://fr.wikipedia.org/wiki/Apple_Computer



Markus Popp : Oval Process Terminal (2004),
container, écran LCD, Mac G4, logiciel.
http://www.medienkunstnetz.de/works/oval-process/

Schématiquement, un ordinateur est composé de trois parties distinctes:

1- la **Mémoire Centrale**.
2- l'**Unité Centrale**.
3- les **Périphériques**.



← Processeur (unité centrale)

← Mémoire

La carte-mère permet la connexion des éléments essentiels.

Pour en savoir plus :
http://www.commentcamarche.net/pc/carte-mere.php3

1- la **Mémoire Centrale** permet de mémoriser les programmes pendant le temps nécessaire à leur exécution. On y trouve également les informations temporaires manipulées par ces programmes: données après leur introduction, résultats avant leur communication à l'extérieur, informations intermédiaires apparaissant pendant le déroulement d'un programme.

La mémoire centrale correspond à ce que l'on appelle la **Mémoire Vive** ou **RAM** (Random Access Memory, mémoire à accès direct).

Contrairement au stockage de données sur une mémoire de masse telle que le disque dur, la mémoire vive est volatile, c'est-à-dire qu'elle permet uniquement de stocker des données tant qu'elle est alimentée électriquement. Ainsi, à chaque fois que l'ordinateur est éteint, toutes les données présentes en mémoire sont irrémédiablement effacées.

Plus précisémment, on peut distinguer :

- les **Registres** : petites mémoires rapides de 8, 16, 32 ou 64 bits. Suivant le type de processeur, le nombre global de registres peut varier d'une dizaine à plusieurs centaines.

- la **Mémoire Cache** (mémoire tampon) : mémoire rapide permettant de réduire les délais d'attente des informations stockées en mémoire vive.

Pour en savoir plus :
http://www.commentcamarche.net/pc/memoire.php3

2- L'**Unité Centrale** (ou processeur, ou CPU - Central Processing Unit, ou UCT - Unité Centrale de Traitement) est la partie active de l'ordinateur. Elle est chargée de prélever une à une chaque instruction de programme située en mémoire centrale et de l'exécuter.

Plus précisémment, on peut distinguer deux sortes d'instructions:

- celles qui agissent sur des informations situées en mémoire centrale. Ce sont elles qui permettent de réaliser le traitement escompté.

- celles qui assurent la communication ou l'archivage d'information. Elles réalisent en fait un échange d'information entre la mémoire centrale et d'autres appareils nommés **Périphériques**.

Pour en savoir plus :
http://www.commentcamarche.net/pc/processeur.php3

3- Les **Périphériques** désignent tous les appareils susceptibles d'échanger des informations avec la mémoire centrale.

Ils sont de deux sortes :

- ceux qui assurent la communication entre l'homme et l'ordinateur (claviers, écrans, souris...)

- ceux qui assurent l'archivage d'information (disques durs, sticks USB...). Ces derniers ont un rôle de mémorisation d'information au même titre que la mémoire centrale, dont ils constituent ainsi une sorte de prolongement.

Pour en savoir plus :
http://www.commentcamarche.net/pc/peripherique.php3

Lorsqu'un ordinateur **exécute** un programme, son travail consiste en grande partie à **gérer la mémoire** :
- soit pour y **lire** une instruction.
- soit pour y **stocker** une information.

En ce sens, l'ordinateur peut être perçu comme un **robot** qui agit en fonction d'**ordres** qui lui sont fournis.

Ces actions sont en nombre limité :

- **Déposer** ou **lire** une information dans une case mémoire.
- **Exécuter** des opérations simples telles que l'addition ou la soustraction.
- **Comparer** des valeurs.
- **Communiquer** une information élémentaire.
- **Coder** l'information.

Déposer ou lire une information dans une case mémoire.

La mémoire est formée d'éléments, ou **cases-mémoire,** qui possèdent chacune un numéro (une **adresse**).

Chaque case-mémoire est en quelque sorte une **boîte aux lettres** pouvant contenir une information (une lettre). Pour y déposer cette information, l'ordinateur (le facteur) doit connaître l'adresse de la boîte.

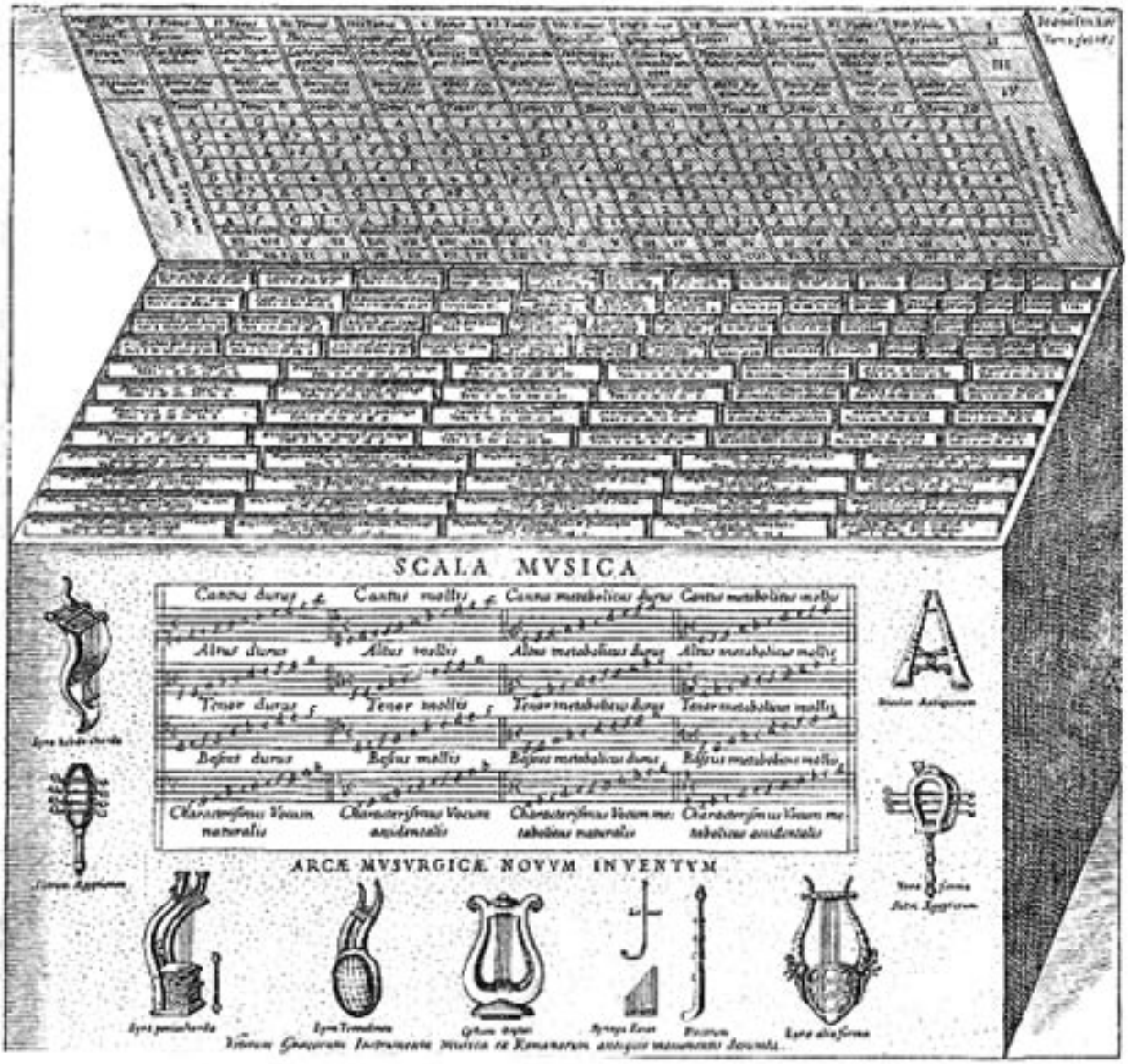
Lorsque le robot place une information dans une case mémoire, il mémorise l'adresse où se situe celle-ci afin de retrouver l'information en temps nécessaire. Le robot sait déposer une information dans une case, mais il ne sait pas la retirer (au sens de prendre un courrier déposé dans une boîte aux lettres).

Lorsque le robot prend l'information déposée dans une case mémoire, il ne fait que la lire. En aucun cas il ne la retire ni ne l'efface. L'information lue reste toujours dans la case mémoire.

Remarque : pour effacer une information d'une case mémoire, il est nécessaire de placer une nouvelle information dans cette même case. Ainsi, la nouvelle donnée remplace l'ancienne, et l'information précédente est détruite.

Les **cases-mémoires** (aussi appelées **Registres**) de la mémoire centrale peuvent être comparées aux tiroirs d'une armoire de bureau ou à des casiers à courrier.

Jacques Tati : Playtime (1967).
Une unité centrale de calcul ?
http://www.tativille.com/



Athanasius Kircher (1601-1680) a inventé un système destiné à engendrer des partitions musicales, ce qui fait de lui le père de la musique algorithmique générative. Il est également l'auteur de concepts d'instruments de musique automatisés.

http://en.wikipedia.org/wiki/Athanasius_Kircher
http://kircher.stanford.edu/

Exécuter des opérations simples
telles que l'addition ou la soustraction.

La machine **lit** et **exécute** les opérations dans l'ordre où elles lui sont fournies.

Pour faire une addition, il va chercher les valeurs à additionner dans les **cases-mémoire** appropriées (stockées, par exemple, aux adresses **a** et **b**) et réalise ensuite l'opération demandée. Il enregistre alors le résultat de cette opération dans une case d'adresse **c**.

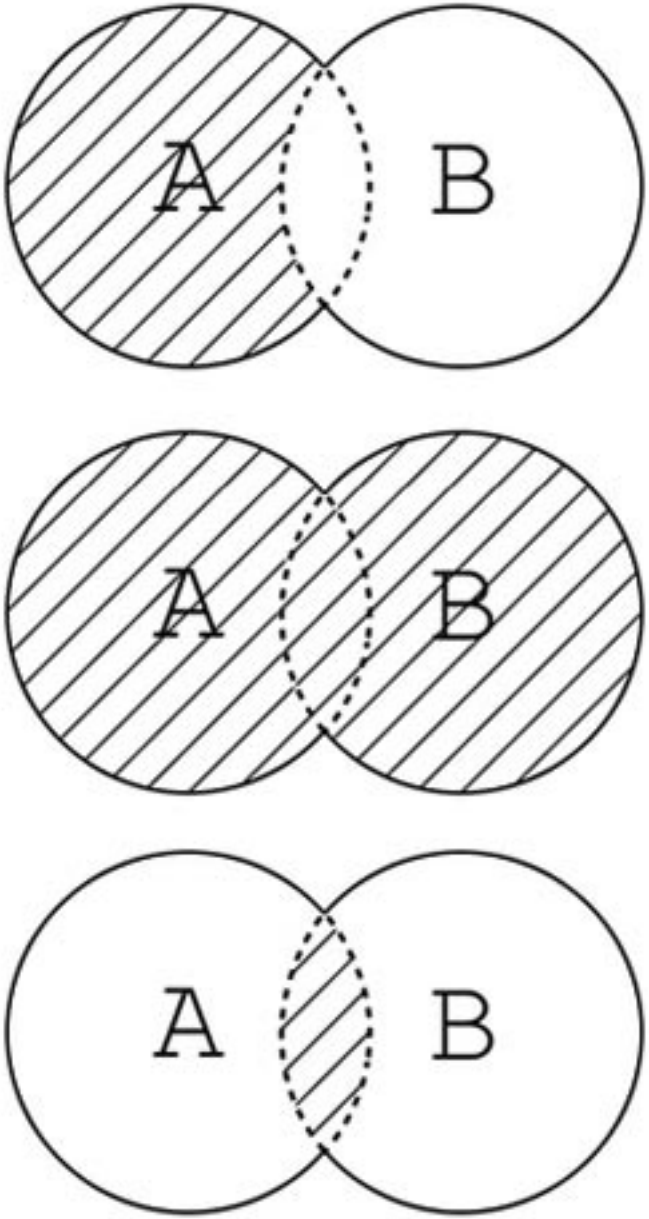
De telles opérations sont décrites à l'aide d'ordres, appelés aussi **instructions**.

Comparer des valeurs.

La machine est capable de **comparer** deux valeurs entre elles pour déterminer si l'une d'entre elles est plus grande, plus petite, égale ou différente de l'autre valeur. Grâce à la comparaison, la machine peut tester une condition et exécuter un ordre plutôt qu'un autre, en fonction du résultat du test.

La réalisation d'une comparaison ou d'un test fait que le robot ne peut plus exécuter les instructions dans leur ordre d'apparition. En effet, suivant le résultat du test, il doit rompre l'ordre de la marche à suivre, en sautant une ou plusieurs instructions. C'est pourquoi il existe des **instructions** particulières dites **de branchement**. Grâce à ce type d'instructions, le robot est à même non seulement de sauter des ordres mais aussi de revenir à un ensemble d'opérations afin de les répéter.

Pour en savoir plus :
http://www.commentcamarche.net/logic/intro.php3



Algèbre de Boole : fonctions logiques OU, ET, NON.
http://fr.wikipedia.org/wiki/Algèbre_de_Boole_%28logique%29

Communiquer une information élémentaire.

Un programme est essentiellement un outil qui traite l'information.

L'utilisateur **transmet** cette information à l'ordinateur par l'intermédiaire du clavier ou de la souris.

Cette transmission de données à l'ordinateur est appelée **entrée**, **input**, **saisie** ou encore **lecture de données**.

Après **traitement**, le programme fournit un **résultat** à l'utilisateur, soit par l'intermédiaire de l'écran, soit sous forme de fichiers.

Ce retour de données (de l'ordinateur vers l'utilisateur) est appelé **sortie**, **output**, **affichage** ou encore **écriture de données**.

Coder l'information.

De par la nature de ses composants électroniques, le robot ne perçoit que deux états: composant **allumé** et composant **éteint**. De cette perception découle le **langage binaire**, qui utilise par convention les deux symboles **0** (éteint) et **1** (allumé). Ne connaissant que le 0 et le 1, l'ordinateur utilise un code pour représenter une information aussi simple qu'un nombre entier ou un caractère. Ce code est un programme, qui différencie chaque type d'information et transforme une information (donnée numérique ou alphabétique) en valeurs binaires. À l'inverse, ce programme sait aussi transformer un nombre binaire en valeur numérique ou alphabétique. Il existe autant de codes que de types d'informations. Cette différenciation du codage (en fonction de ce qui doit être représenté) introduit le concept de **type de données**.

Toute information fournie à l'ordinateur est, au bout du compte, codée en binaire. l'information peut être un simple nombre ou une instruction de programme.

Exemple : Pour additionner deux nombres, l'ordinateur fait appel aux trois éléments qui lui sont nécessaires pour réaliser cette opération.
Ces éléments sont les suivants:
- le code binaire représentant l'opération d'addition (par exemple 0101) ;
- l'adresse de la case mémoire où est stocké le premier nombre (par exemple 011101) ;
- l'adresse de la case mémoire où se trouve la deuxième valeur (par exemple 010101).

Pour finir, l'instruction d'addition de ces deux nombres s'écrit en assemblant les trois codes binaires (soit, dans notre exemple, 0101011101010101).

Remarque :
Le code binaire associé à chaque code d'opération (addition, test, etc.) n'est pas nécessairement identique d'un ordinateur à un autre. Ce code binaire est déterminé par le constructeur de l'ordinateur. De ce fait, une instruction telle que l'addition de deux nombres n'a pas le même code binaire d'une machine à une autre. Il existe donc, pour un même programme, un code binaire qui diffère suivant le type d'ordinateur utilisé.

Pour en savoir plus :
http://www.commentcamarche.net/base/binaire.php3

La notion d'INSTRUCTION.

Les instructions exécutées par l'ordinateur sont des traitements élémentaires simples (comparaison de valeurs, addition, soustraction, multiplication ou division) opérant sur des données que l'ordinateur puise dans sa mémoire.

Ces données, préalablement transmises par l'utilisateur ou par le programme lui-même, sont des assemblages de bits (valeurs de 0 ou de 1).

Ces assemblages sont de deux types:
- l'**octet**, qui correspond à huit bits;
- le **mot**, terme plus général qui comprend un nombre de bits variant selon les ordinateurs (8, 16, 32, 60...)

Les instructions sont également représentées comme des suites de 0 et de 1 dans la mémoire de l'ordinateur.

Pour en savoir plus :
http://www.commentcamarche.net/langages/langages.php3

La notion de VARIABLE.

Une instruction machine effectue des opérations sur des **valeurs** repérées par leur **adresse**. Une telle adresse est représentée par un **nom**.

Une **variable** est précisémment ce nom qui sert à désigner un **emplacement** donné de la mémoire centrale.

Cette notion, simple en apparence, contribue considérablement à faciliter la réalisation de programmes. Elle vous permet, en effet, de manipuler des valeurs sans avoir à vous préoccuper de l'emplacement qu'elles occuperont effectivement en mémoire: il vous suffit simplement de leur choisir un nom.

Bien entendu, la chose n'est possible que parce qu'il existe un programme de traduction (compilateur ou interprète) de votre programme vers le langage machine : c'est lui qui attribuera une adresse à chaque variable.

Pour en savoir plus :
http://www.commentcamarche.net/langages/structure.php3

La notion de TYPE.

Les informations conservées en mémoire sont toujours codées en langage binaire. Cela est vrai en particulier pour le contenu des variables. Comme il est nécessaire de pouvoir conserver des informations de nature différente (par exemple des **nombres** et des **lettres**), il faut employer plusieurs codes différents. Dès lors, la connaissance du contenu binaire d'une variable ne suffit pas pour déterminer l'information correspondante. Il est nécessaire de savoir, en outre, comment la valeur qui s'y trouve a été codée.
Cette distinction correspond à la **notion de type**.

Ainsi, une variable destinée à recevoir des valeurs telles que **123** ou **12,45** est de type **numérique**. Une variable destinée à contenir des **lettres** sera dite de type **caractère**.

Le type limite les opérations : les opérations arithmétiques (addition, soustraction, multiplication, division), possibles avec des variables numériques, n'ont aucun sens pour des variables de type caractère.
Par contre, les comparaisons seront possibles pour les deux types.

En résumé, le type d'une variable définit:
- la **nature des informations** qui seront représentées dans la variable (numériques, caractères).
- le **codage** utilisé.
- les **limitations** concernant les valeurs qui peuvent être représentées.
- les **opérations** réalisables avec les variables correspondantes.

Pour en savoir plus :
http://www.commentcamarche.net/langages/structure.php3

Dans la vie courante, les êtres humains n’ont pas besoin de préciser le type des informations qu’ils échangent, car il est explicite : nous comprenons que 23 représente un nombre tandis que DUPONT est une suite de caractères. Ce n’est pas le cas pour l’ordinateur : il faut lui expliquer la différence.