DEUG MIAS et MASS S1 (première année, premier semestre) année 2001-2002 (septembre 2001)

Elément Constitutif (=module) d'informatique dans l'Unité d'Enseignement "fondamental" (MIAS) ou "complément+découverte" (MASS).
Cours-TD 30h (12*2.5) + TP 20h (10*2).

Présentation

Vérifier les horaires et les salles, donner le planning et les noms des encadrants.

Demander une disquette PC/HD3.5 pour les TP ?

Rappeler les règles de propriété des logiciels.

Les choix faits pour cette année (gpc+xemacs) et l'évolution prévisible (Delphi/Kylix). Attitude commerciale de Borland (=Inprise) : Turbo Pascal 7 est téléchargeable gratuitement ; La revue Presqu'Offert propose Delphi 1.0 à moins de 100fr ; Delphi 5 existe en pack étudiant à 450fr.

Enjeux du cours : culture personnelle, capacité techniques minimales dans un environnement faisant de plus en plus appel à ces techniques (NTIC), début de spécialisation professionnelle scientifique où la maîtrise de l'informatique est indispensable (voir l'évolution du CAPES de maths et la reconversion en informaticiens des Bac+5 divers).

DEUG MIAS et MASS S1 (première année, premier semestre) année 2001-2002 (septembre 2001)

Plan du cours

(1) presentation-generalites

(2) structure d'un programme en Pascal ; objets et instructions elementaires
(integer, real, char, string[n], readln, write, writeln...)
----- TP1 -----

(3) schemas itératifs
3.0. schema sequentiel
3.1. schemas alternatif ; objets booleens
3.1.1. objets booleens (boolean)
3.1.2. alternative simple (if-then-else)
3.1.3. alternative etendue, menus [au moins 3 choix] (case-of)
----- TP2 -----
3.2. schemas repetitif
3.2.1. boucles bornees simples (for-to|downto-do)
----- TP3 -----
3.2.2. boucles precontrolees (while-do)
3.2.3. boucles postcontrolees (repeat-until)
----- TP4 -----

(4) variables indicées (array)
4.1. a un indice, parcours de tableau (sommes, produits, maximum, minimum...)
----- TP5 -----
==========> Partiel le 24 novembre 2001
avec tableaux et menus.
Sa note ne compte que si elle est
superieure a celle de l'examen terminal.
4.2. a deux indices + boucles imbriquees
----- TP6 -----

(5) sous-programmes
5.1. procedures (programmation modulaire)
----- TP7 -----

5.2. fonctions (avec parametres par valeurs)
----- TP8 -----

(6) programmation avancee
6.1. portee des variables locales|globales
6.2. parametres par adresse
6.3. etc
----- TP9 -----
----- TP10 ----

(7) Revisions (12eme cours)

==========> examen terminal vers les 7-8 janvier 2002
avec sous-programmes, variables a 2 indices.
==========> deuxieme session d'examen terminal en juin 2002
(possibilite d'utiliser leurs connaissances nouvelles du second semestre).

N.B. : ne sont donc pas au programme ?
les unites et la compilation separee
les fichiers et pointeurs
le schema recursif

(1) Introduction

Le mot informatique a été créé en ~ par Philippe Dreyfus, alors qu'il travaillait pour la compagnie Bull. Ce mot est construit à partir du mot "information", avec un suffixe "-ique" imité de mots tels que mathématique, électronique, technique ; il doit être regardé comme un "mot-valise" formé de "**informati**on" et auto**matique**".

L'informatique regroupe donc les techniques de la collecte, du tri, de la conservation (mise en mémoire, stockage), de la transmission et de l'utilisation des informations **traitées automatiquement** à l'aide de programmes appelés logiciels. Plus rapidement l'informatique correspond aux théories et techniques de traitement automatique de l'information.

Les anglo-saxons utilisent "computing science", ce qui fait référence plus explicitement au calcul (voir "comput" qui correspond aux calculs des dates du calendrier ecclésiastique, du latin "computare" : calculer, qui a donné compter).

Les machines qui servent à traiter cette information sont appelés, en français, **ordinateurs**. Ce mot a été proposé en 1954 par le professeur Jacques Perret (éminent latiniste très averti des choses de la religion catholique) en faisant référence à la fois à l'aspect **ordonné** du travail de ces machines (latin ordo=ordre) et aux évêques catholiques qui, donnant l'**ordination** aux sous-diacres, diacres et prêtres, sont donc nommés **ordinateurs** (=ordinants, sens du mot déjà oublié du public à cette époque) !

Les anglo-saxons, ont choisi le mot **computer** (i.e. calculateur).

Si l'automatisation du calcul fut un enjeu historique fondamental (voir pour la seconde guerre mondiale, les capacités de décryptage des alliés, ainsi que la puissance de calcul nécessaire à la conception d'une bombe atomique ; voir aussi l'envoi de fusées ou satellites dans l'espace), il ne faut pas oublier les progrès réalisés à partir des besoins des machines à tisser (métiers ~Jacquard) ou des "votations".

Si à l'heure actuelle, rapidité, puissance et autonomie caractérisent l'informatique moderne, il ne faut cependant pas en oublier les limites : incapacités à prendre en charge certaines dimensions (éthique, linguistique...) et fragilités liées à la complexité des machines et des réseaux qui les relient. Une formation solide et des qualités de rigueur sont indispensables pour devenir un bon informaticien.

On fait habituellement la distinction entre l'aspect matériel (hardware) et fonctionnel (software) de l'informatique.

Par ailleurs, le fonctionnement des ordinateurs suit une logique très éloignée du fonctionnement de l'esprit humain, une bonne partie d'un projet informatique consistera donc à déterminer précisément ce qu'on veut faire domaine de l'algorithmique) pour le traduire dans un langage accessible à l'ordinateur (programmation). Même si un travail logiciel considérable a été réalisé pour rapprocher l'ordinateur des habitudes humaines, la conception et la réalisation d'un projet informatique reste encore un travail de spécialiste selon des méthodes qui sont encore perfectibles (qualité notamment).

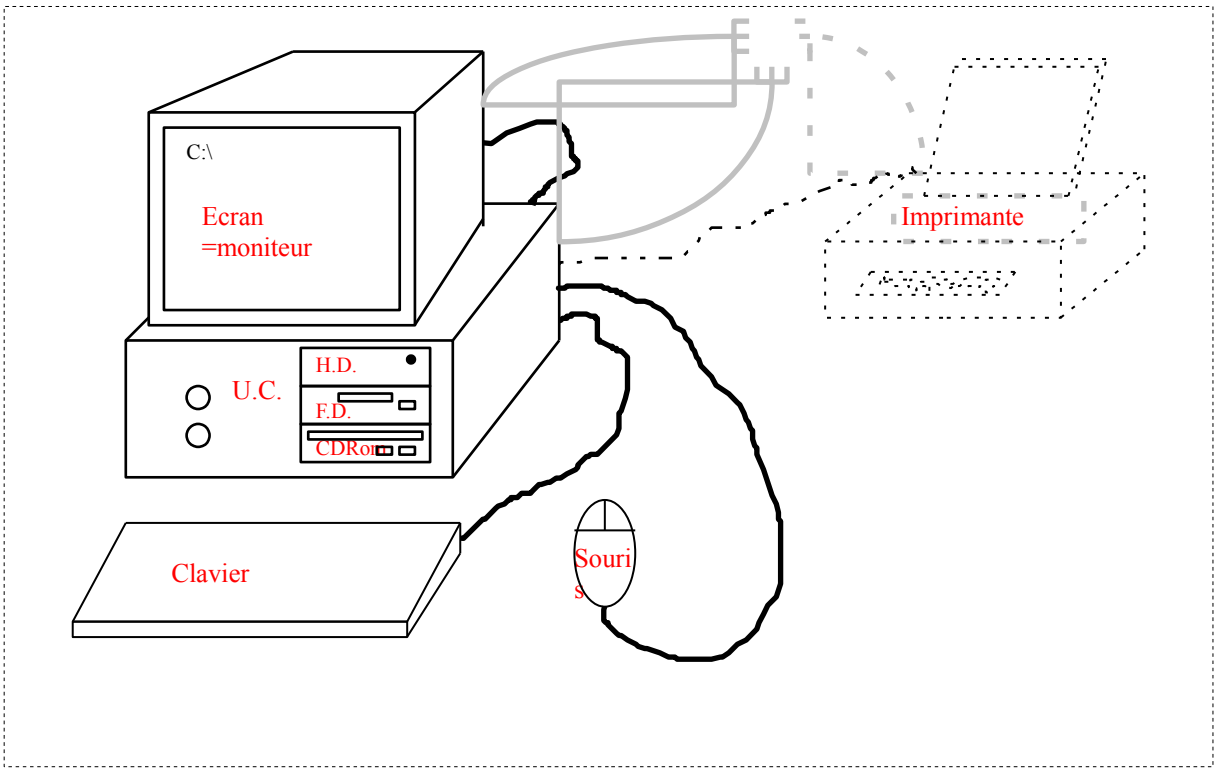
1.) qu'est-ce q'un ordinateur ?

1.1.) Bref historique

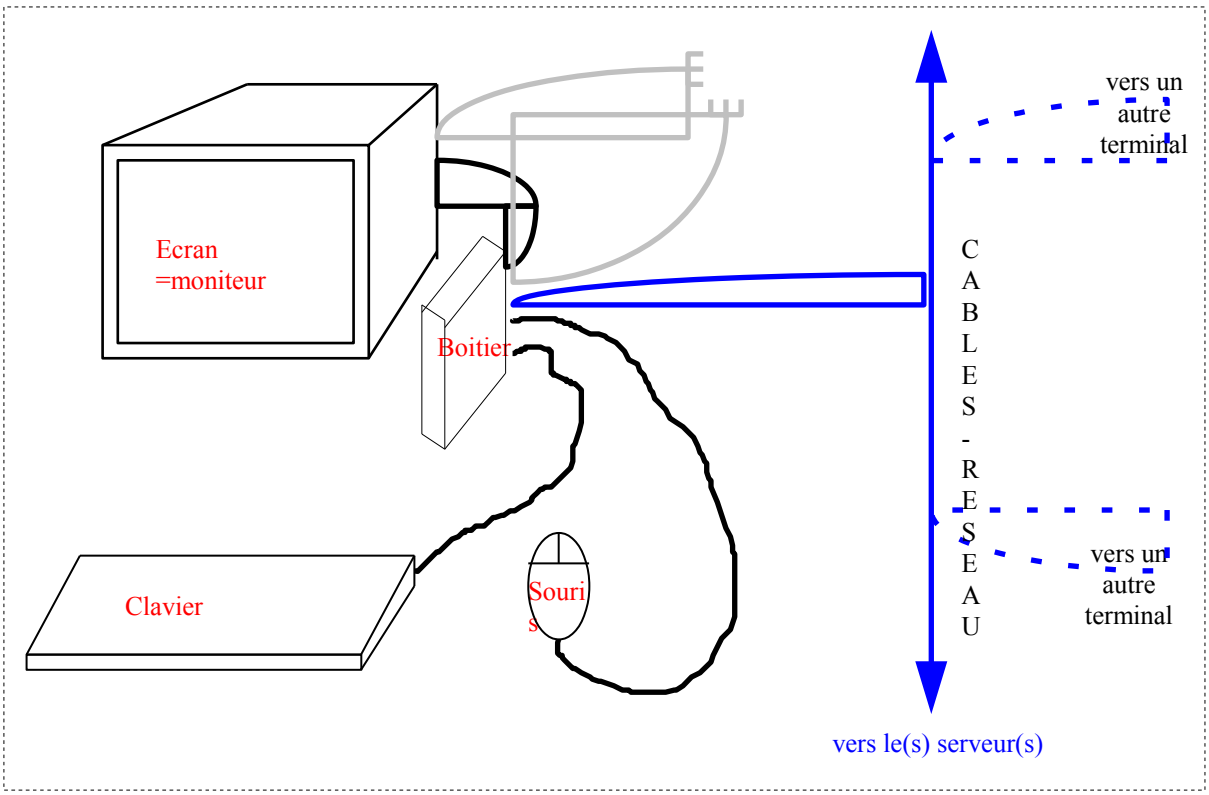
Machine de calcul de Pascal (Pascaline), machines mécaniques/hydrauliques antérieurs et ultérieures.
Premiers calculateurs électriques à base d' "ampoules triodes", taille, consommation, fragilités.
L'électronique et la miniaturisation : transistor puis circuits intégrés.
Les ordinateurs diminuent progressivement de taille (premières calculatrices 197*) jusqu'à pouvoir tenir sur un bureau.
Après divers précurseur (Bull,...), IBM lance, en août 1981, le PC (personnal computer) avec 64Ko de MEV et ? des FD5.25 360Ko... pourvu d'un système d'exploitation minimale : MSDOS, fourni par une petite entreprise : MicroSoft.
Clônes et progrès, voies parallèles de Apple et autres. Qualités et défauts comparés, succès et échecs.

1.2.) organisation matérielle d'un PC

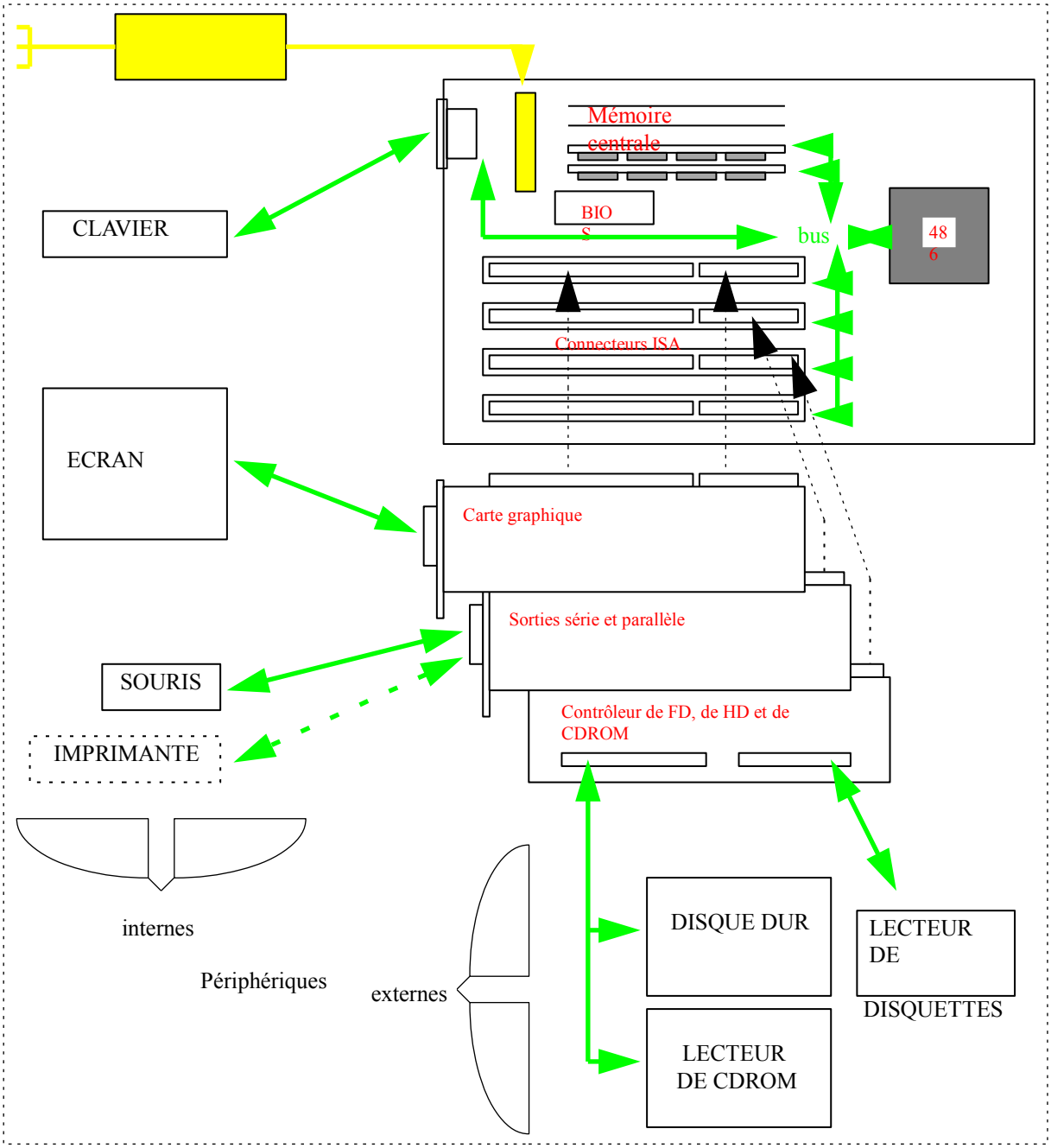
1.2.1.) schéma d'ensemble



1.2.1.) un modèle légèrement différent : le terminal



1.2.3.) architecture d'un PC 486



Le BIOS contient de la mémoire morte (=ROM), c'est-à-dire des informations qui ne peuvent pas être modifiées.

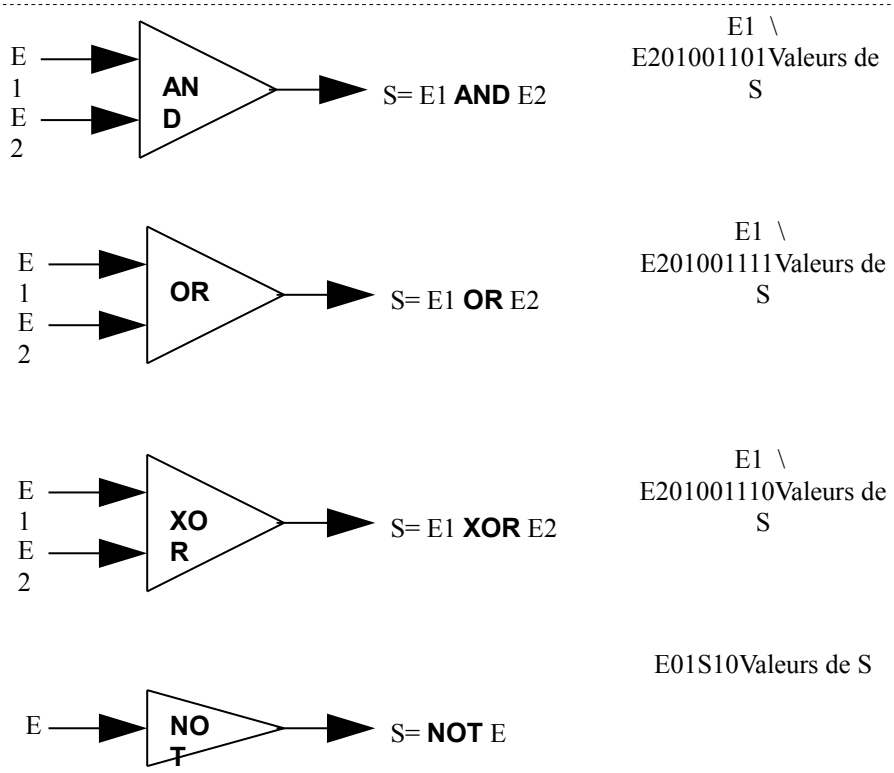
La mémoire centrale composée de barrettes de mémoire vive (=RAM) contient des informations modifiables utilisées quand l'ordinateur est en marche. Elles s'effacent quand l'ordinateur est éteint.

Les mémoires auxiliaires, appelées aussi mémoires de masse (correspondant ici aux périphériques internes) peuvent conserver leurs informations de manière rémanente. Disques durs et disquettes peuvent être réécris et effacés, alors que les données d'un CD ne peuvent plus être modifiées (ce n'est pas le cas des CD-R ni des CD-RW).

On distingue souvent les périphérique d'entrée (clavier,...) des périphériques de sortie (écran,...).

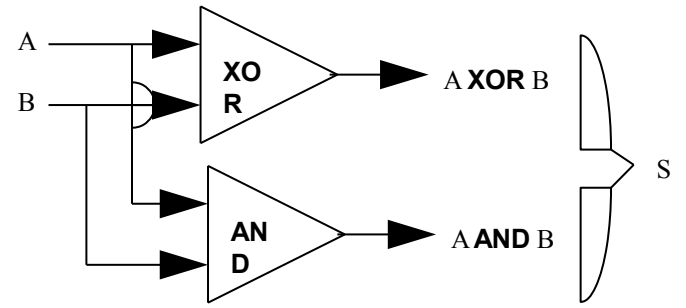
1.3.) le microprocesseur

Le microprocesseur peut être vu comme le "cerveau" de la machine, cerveau relié à la mémoire centrale et aux les périphériques. Il est formé sur une surface de quelques cm2 (mm2) d'un circuit intégré composés de millions d'unités électroniques élémentaires (équivalent des transistors) relié par un réseau de fils microscopiques (taille actuelle de gravure : 0,18µm).

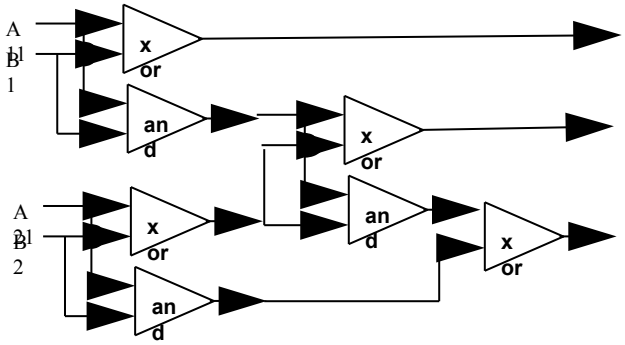


Quelques portes logiques gravables sur des circuits intégrés

Comment le microprocesseur peut-il réaliser des calculs ?



Additionneur 1 bit avec retenue



Additionneur 2 bits avec retenue

Les premiers PC regroupaient 8 bits (=1 octet) en parallèle pour le calcul des entiers naturels et pouvaient donc calculer la somme de deux nombres compris entre 0 et 255 (2^8 valeurs). Par la suite, la capacité des processeurs a été portée à 16 bits (0 à 65535) puis 32 bits (0 à 2^32–1#3.E6).
Quand un processeur "8 bits" a une fréquence de 40MHz, il peut effectuer 40E6 opérations sur des nombres à 8 bits par secondes.

Un processeur ne fait pas seulement des calculs (unité de calcul), il communique avec la mémoire centrale et les périphériques (unité de traitement). Cependant la vitesse de la mémoire centrale (100 ou 133 MHz) et des périphériques (HD 66MHz) est beaucoup plus lente que celle des processeurs actuels (n*100MHz en 32 bits), les ordinateurs ont donc diverses mémoires tampons permettant d'accumuler temporairement les informations pour laisser le processeur travailler le plus vite possible. Conséquences concrètes : la sauvegarde effective d'un fichier sur la disquette peut être fait quelques secondes après que l'ordinateur ait considéré cette action comme réalisée, l'ordinateur est disponible avant la fin d'une impression.

Les diverses mémoires peuvent conserver une quantité très importante d'information, mesurée en octets (1 octet = 8 bits). Valeurs à retenir : 1 Kb=2^10 b=1024 b, 1Mb=2^100 b#1000000 b, etc. Même principe pour les octets. Ordres de grandeurs actuels : FD à 1,44MB (standard PC-HD-3.5), HD à 10GB, RAM à 64MB (à comparer aux premiers PC : FD à 360KB, RAM à 64KB).

En conclusion, puissance et rapidité particularisent l'activité de l'ordinateur.

1.4.) l'émergence du sens (software)

On peut se demander comment ce qui précède permet à l'ordinateur de calculer sur des nombres réels, traiter des textes et des dessins élaborés, ... Ceci est réalisé par une organisation concentrique de programmes de plus en plus élaborés.

Niveau 0 : les circuits gravés dans le microprocesseur lui donne des capacités de calculs et de communication élémentaires. Exemple : additionner, soustraire deux entiers à n bits, ou deux entiers signés (1 bit pour le signe) ; placer le résultat dans la mémoire vive.

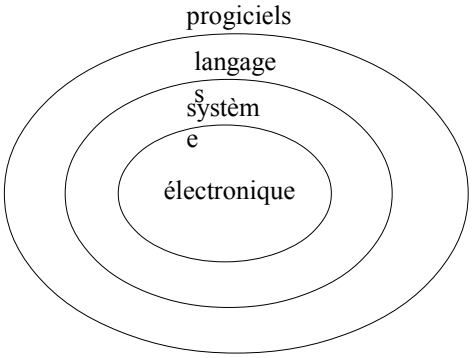
Niveau 1 : le BIOS contient un mini-programme (l'amorce) chargé, au démarrage de l'ordinateur de chercher sur une des mémoires de masse le système d'exploitation et de le charger en mémoire.

Niveau 1bis : le système d'exploitation = operating system (MSDOS, Unix, MacOS,...) gère les différentes parties de l'ordinateur (affichage écran, saisie clavier, lecture-écriture des disques,...) et présente à l'utilisateur une organisation cohérente (système de répertoire, "drivers" divers,...).

Niveau 1ter : un programme supplémentaire peut venir améliorer l'apparence du S.E. : Windows (MSDOS), Xwindow (Unix)...

Niveau 2 : l'utilisateur peut donc lancer des logiciels, dont certains, tel TurboPascal, lui permettront de fabriquer d'autres programmes, grâce à un langage de programmation. Ces logiciels sont appelés "langages".

Niveau 3 : certains logiciels, appelés progiciels, sont spécialisés dans une utilisation de l'ordinateur à une activité particulière : traitement et édition de texte (Word,), tableur (Excel,), gestions de données structurées (Dbase, Access,)...



2.) Le traitement automatique

2.1.) description par un algorithme

Le mot algorithme est vient du nom (déformé par l'influence du grec arithmos : nombre) d'un mathématicien arabe, AlKhowarismi, dont l'ouvrage ("...AlDjabr...") a été traduit en latin au XIIème siècle. Il sert actuellement à désigner un ensemble ordonné d'actions agissant sur des objets donnés (une donnée) et aboutissant à un résultat devant résoudre un problème. L'algorithmique, science des algorithmes est un domaine d'étude très vaste de l'informatique théorique.

Exemple 1 : ma recette de tarte aux pommes
Donnée : farine, beurre, eau, sucre, pommes.
Résultat : une tarte au pomme délicieuse.
Recette : Mettre le four en marche
Former avec la farine, le beurre et l'eau une pâte
Etaler la pâte sur un moule
Eplucher les pommes
Couper les pommes en tranches fines étalées sur la pâte
Mettre au four
Attendre pendant 30 mn
Eteindre le four
Retirer la tarte du four et laisser tiédir.

Exemple 2 : Un algorithme pour allumer un ordinateur
Donnée : un ordinateur PC486 éteint.
Résultat : cet ordinateur allumé sous MSDOS.
Procédure : **si** l'écran a un cable d'alimentation séparé **alors** allumer l'écran.
Allumer l'ordinateur.
Attendre que s'affiche l'invite (=prompt) du dos : X:∴ ou Δ:∴ .

Remarques :

1) Il faut bien connaître les objets que l'on manipule afin de ne pas faire d'erreur. Par exemple, ne pas mettre l'écran au four !

2) La description peut être plus ou moins détaillée selon l'exécutant auquel il s'adresse. Par exemple, ma recette de tarte aux pommes s'adresse aux cuisinier(e)s expérimentés, qui sauront quels proportions choisir. Selon le destinataire de l'algorithme, il faut plus ou moins de précision, voire même un changement de langage.

3) L'ordre de réalisation des instructions est important. Par exemple, allumer le moniteur d'un ordinateur en marche peut provoquer des perturbations électriques ; personne n'aurait non plus l'idée d'allumer le four après en avoir retiré la tarte...

4) Certaines instructions peuvent être inutiles soit parce qu'elles sont redondantes (laisser tiédir), soit parce que leur réalisation est soumise à une condition non réalisée (ordinateur avec écran alimenté par l'intermédiaire de l'U.C.).

5) Enfin si l'algorithme est sensé résoudre un problème donné (je veux allumer l'ordinateur, je veux une bonne tarte aux pommes), rien ne prouve, a priori, qu'il le fasse de manière optimale ni même qu'il le fasse réellement. Par exemple, en 30mn, la tarte peut commencer à brûler ou rester insuffisamment cuite! On pourrait remplacer l'instruction "Attendre pendant 30 mn" par "Attendre jusqu'à ce que la tarte soit cuite à point". Cela suppose de savoir reconnaître une tarte cuite à point, sinon la boucle d'attente risque d'être infinie...ou...

La mise au point d'un algorithme résolvant un problème de manière satisfaisante est appelée ANALYSE [fonctionnelle ; conception ; design]. L'informaticien chargé de ce travail est appelé "analyste".

Définition : Un algorithme est une description méthodique des traitements à effectuer sur des données...
(Cf Wirth). Un programme fonctionnera sur le même principe, mais le langage de description des traitements et des données devra suivre des règles assez strictes pour êtres compréhensibles par un ordinateur.

2.2.) le travail de l'ordinateur

L'ordinateur travaille sur certains types de données (essentiellement des entiers en nombre borné) et peut leur appliquer un nombre fini d'instructions. Le langage correspondant est appelé LANGAGE MACHINE, il est très généralement différent d'un processeur à l'autre (Intel486, Motorola68000,...). Le S.E. met à la disposition de l'utilisateur, avec une forme utilisable (="assembleur") de ce langage, un certain nombre de commandes de base : cd md, rd, del, rd, dir, ... (sous DOS).

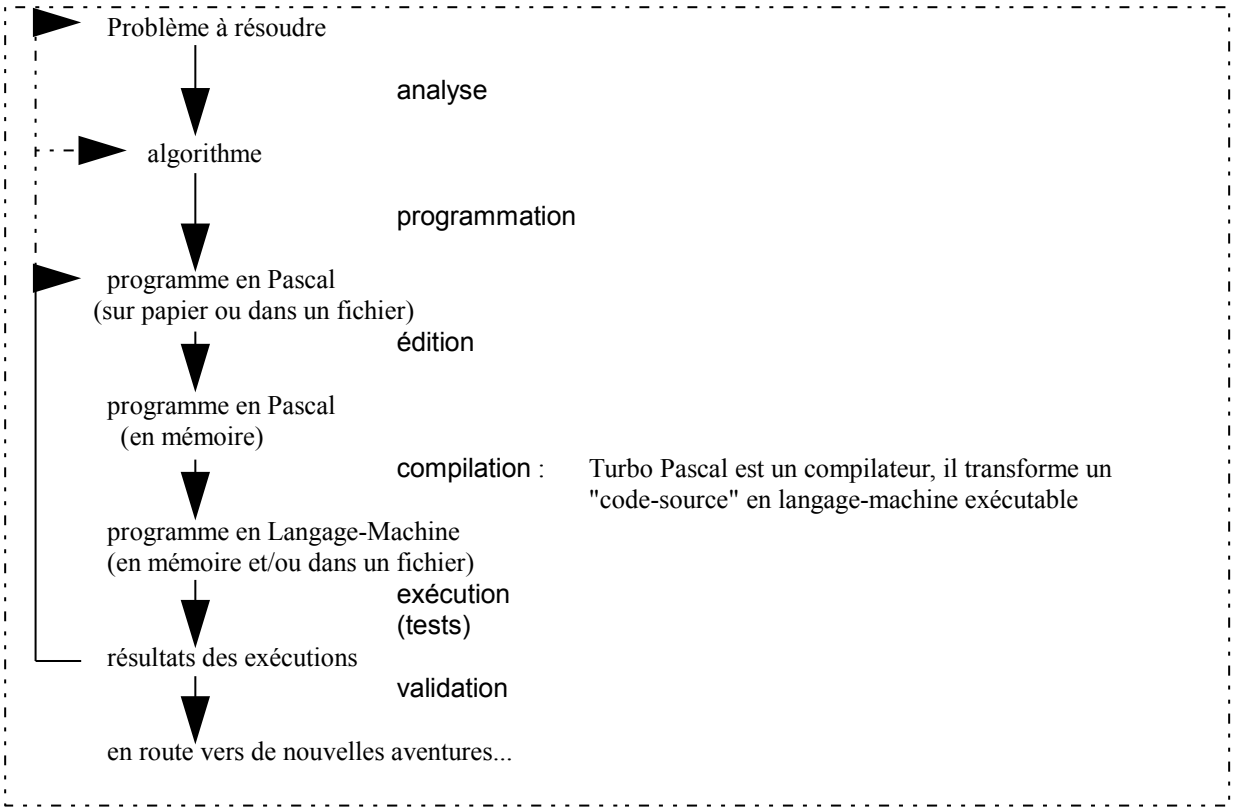
Exemple de programme :

En assembleur	en langage machine (hypothétique)	algorithme
PUSH R1	10001111.00000001	
PUSH R2	10001111.00000010	
MOV R1 Adr(x)	11000010.10110001	
MOV R2 Adr(y)	11000010.11000010	
ADD R1 R2	10100001.00010010	
MOV Adr(x) R1	11000001.00010011	x <– x + y
POP R1	11110000.00000001	
POP R2	11110000.00000010	

Pour se rapprocher de la compréhension humaine et de ses mécanismes linguistiques, il a été créé (et il se crée encore) de nombreux langages :
Fortran (Formula translator),
Basic (Beginner's Allpurpose ...),
Cobol (Common Business Oriented Language),
APL (A programming language),
PL1,
Forth,
Prolog
Lisp ("language including silly parenthesies"),
Pascal,
C (suite du B), C++,
Java (café argotique américain),
...

Ces langages permettent, en créant des données structurées et leurs opérateurs, mais aussi en se présentant sous une forme linguistique plus habituelle, de traduire rapidement un algorithme, en leur équivalent pour le langage considéré : un programme. La variété des langages peut s'expliquer, outre la fantaisie créatrice des humains, par l'adaptation à différents domaines (Cobol versus Fortran, Pascal versus C) ou à différents problèmes (Lisp, Prolog, ...). Le rôle du programmeur est d'écrire de tels programmes et de corriger les erreurs (=bogues) jusqu'à ce qu'ils soient acceptés par la machine.

2.3.) De l'énoncé du problème à sa résolution



L'algorithme de déroulement d'un TP sur PC

Donnée : un ordinateur éteint.
Résultat : un ordinateur éteint, le travail fait et mémorisé sur disquette.
Procédure : Allumer l'ordinateur (* algorithme connu *)
Si l'invite est : Δ:∴ alors taper : X:∴ <Entrée>
taper : XΔ TΠ6 <Entrée>
taper : TYPBO <Entrée>
Réaliser le TP
Eteindre l'ordinateur (* inverser l'ordre dans l'algorithme d'allumage *).

L'algorithme de déroulement d'un TP sur réseau

Donnée : un terminal disponible.
Résultat : un terminal libéré, le travail du TP2 fait et mémorisé sur le compte.
Procédure : choisir le serveur sur lequel se connecter (scisrv)
Se connecter (donner son nom de connection et son mot de passe)
taper : mkdir TP2 <Entrée>
taper : cd TP2 <Entrée>
Réaliser le TP
Se déconnecter (taper : logout | Faire ...)
(* ne jamais eteindre le terminal !!! *)
N.B. Pour réaliser le TP
taper : xemacs tp2exc1.pas & (* édition du source de l'exercice 1 *)
créer le programme
enregistrer le fichier
répéter
taper : gpc tp2exc1.pas
modifier les erreurs dans le code source
jusqu'à ce qu'il ne reste plus d'erreurs
exécuter ??a.out??