République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieure et de la Recherche Scientifique

Université Abderrahmane Mira - Bejaia

Faculté de Technologie

Département de Technologie



Polycopié de cours

Première Année Technologie

Structure des Ordinateurs et Applications

Enseignant :

Dr. OUARET Ahmed

Année Universitaire : **2026/2027**

A propos de ce cours

Ce support de cours est destiné au module **Structure des ordinateurs et applications**, proposé aux étudiants de première année Licence (L1) dans le cadre du système LMD, ainsi qu'aux étudiants de première année du cycle Ingénieur. Il présente de manière progressive et structurée les fondements de la science informatique : notion de traitement automatique de l'information, évolution historique de l'informatique et des ordinateurs, architecture simplifiée d'un ordinateur, composition matérielle d'un système informatique, codage de l'information, ainsi que les expressions arithmétiques, relationnelles et logiques.

La seconde partie de ce cours est consacrée à l'initiation à la programmation à travers l'algorithmique, étape fondamentale pour tout futur programmeur, puis à la mise en œuvre pratique via le langage de programmation C. Les thèmes abordés incluent : la notion d'algorithme, les étapes de conception algorithmique, les entrées et sorties, les constantes, variables et types de données scalaires, les instructions de base, les instructions conditionnelles et les structures itératives, ainsi que leur traduction en langage C.

À l'issue de ce cours, l'étudiant sera capable d'écrire ses propres algorithmes et programmes en C pour résoudre des problèmes de complexité moyenne.

Pré-requis

Pour suivre ce cours dans de bonnes conditions, l'étudiant doit disposer de connaissances de base en :

- Culture générale en informatique
- Résolution de problèmes et raisonnement logique
- Notions élémentaires sur les langages de programmation

Les compétences visées

À l'issue de cet enseignement, l'étudiant sera capable de :

- Comprendre les concepts fondamentaux de l'architecture des ordinateurs
- Maîtriser les bases de l'algorithmique et de la conception de programmes
- Écrire et exécuter des programmes simples en langage C
- Développer une approche méthodique et rigoureuse dans la résolution de problèmes informatiques

Contenu et organisation de l'enseignement

Cours : 1h30 par semaine

TP : 1h30 par semaine

Méthode d'évaluation : 60% examen et 40% contrôle continu

Table des matières

Introduction 1

Chapitre I : Introduction à l'informatique

I.1. Objectif du chapitre 3
I.2. Définition de l'informatique 3
I.2.1. Role de l'informatique 3
I.3. Evolution de l'informatique et des ordinateurs 4
I.3.1. Evolution de l'informatique 4
I.3.2. Evolution des ordinateurs 5
I.3.2.1. Définition de l'ordinateur 5
I.3.2.2. Générations des ordinateurs 5
I.4. Architecture et fonctionnement d'un ordinateur 6
I.5. Composition d'un ordinateur (Matériel & Logiciel) 8
I.5.1. Partie matérielle (Hardware) 8
I.5.2. Partie logicielle (Software) 11
I.6. Les systèmes de codage des informations 12
I.6.1. Les systèmes de numérotation 12
I.6.2. Conversion d'un nombre d'un système à un autre 14
I.6.3. Le code D.C.B. (Decimal coded Binary – Décimal codé binaire) 18
I.6.4. La codification Alphanumériques 18
I.7. Les expressions arithmétiques, relationnelles et logiques 19
I.7.1. Expressions arithmétiques 19
I.7.2. Expressions relationnelles 20
I.7.3. Expressions logiques (booléennes) 20
I.7.4. Les fonctions 21
I.7.5. La priorité dans les expressions 22
I.7.6. Evaluation des expressions 22
Série de TP N°01 24
Corrigé de la série de TP N°01 26
Exercices supplémentaires sur la série de TP N°01 30

Chapitre II : Notion d'algorithme et de programme

II.1. Objectif du chapitre 32

II.2. Concept d'un algorithme 32
II.3. La démarche et analyse d'un problème 33
II.4. Représentation des algorithmes 33
II.4.1. Représentation par organigramme 33
II.4.1.1. Symboles des organigrammes 34
II.4.2. Représentation par pseudocode 35
II.5. Structure d'un algorithme/programme 35
II.5.1. L'Entête 36
II.5.2. La partie déclarative 37
II.5.3. Le coprs de programme 40
II.6. Types d'instructions 40
II.6.1. Instructions d'Entrées/Sorties (Lecteure/Ecriture) 40
II.6.1.1. Entrées (Lecteure) 40
II.6.1.2. Sorties (Ecriture) 41
II.6.2. Instruction d'afectation 42
II.6.3. Structures de contrôles 51
II.6.3.1. Structures de contrôle conditionnelle 51
II.6.3.2. Structure de contrôle répétitives 55
Série de TP N°02 61
Corrigé de la série de TP N°02 62
Exercices supplémentaires sur la série de TP N°02 66
Série de TP N°03 67
Corrigé de la série de TP N°03 69
Exercices supplémentaires sur la série de TP N°03 80
Série de TP N°04 81
Corrigé de la série de TP N°04 82
Exercices supplémentaires sur la série de TP N°04 93
Bibliographie 94

Introduction

L'informatique, contraction d'information et automatique, est la science du traitement automatique de l'information. Depuis son émergence au milieu du 20ème siècle, elle n'a cessé d'évoluer à un rythme soutenu, étendant progressivement son champ d'application bien au-delà du simple calcul. Elle englobe aujourd'hui des domaines aussi variés que l'automatisation des processus, le contrôle et la commande des systèmes, la communication et le partage de l'information.

Tout système informatique repose sur deux composantes fondamentales et complémentaires : le matériel (hardware), qui désigne l'ensemble des éléments physiques du système tels que l'unité centrale, la mémoire et les organes d'entrées-sorties, et le logiciel (software), qui correspond à l'ensemble des instructions et des programmes définissant le comportement de la machine.

Ce cours intitulé **« Structure des ordinateurs et application »** vise à fournir une compréhension solide de l'architecture des ordinateurs et des principes fondamentaux de la programmation. Il est structuré en plusieurs unités d'apprentissage couvrant les composants matériels d'un ordinateur, les systèmes de codage de l'information, la conception d'algorithmes ainsi que la méthodologie d'analyse et de résolution de problèmes.

Le langage de programmation C est utilisé comme support pratique tout au long de ce cours. Reconnu pour sa proximité avec le matériel, sa performance et sa portabilité, il représente un outil particulièrement adapté pour illustrer concrètement les concepts abordés et initier l'étudiant à une programmation rigoureuse et structurée.

L'ensemble de ces composantes est illustré par la figure 1, qui place les algorithmes au cœur du processus informatique, en interaction avec les données, les machines, les langages de programmation et les interfaces avec le monde extérieur.

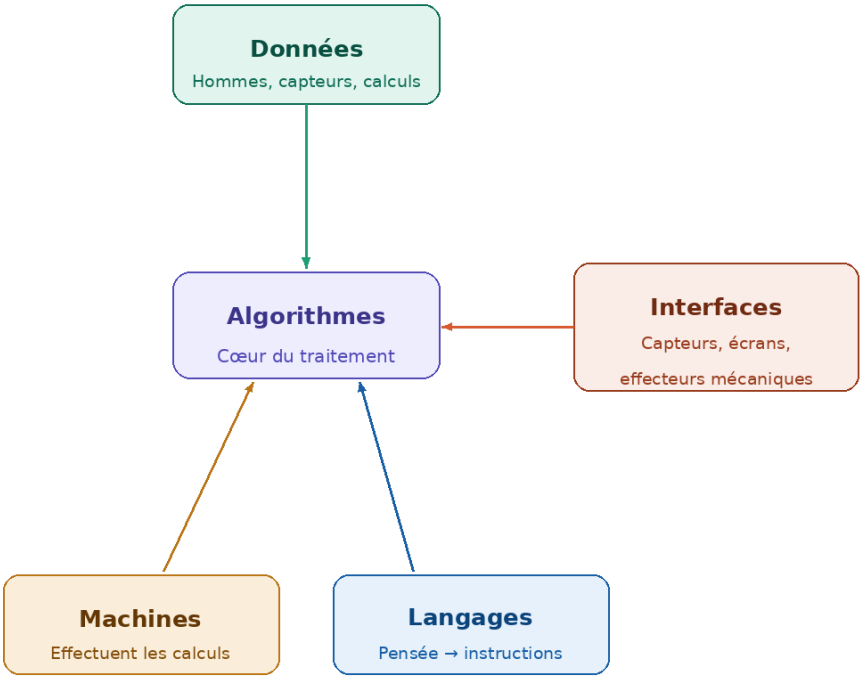


Figure 1 : Les composantes de l'informatique

CHAPITRE I

Introduction à l'informatique

Chapitre I : Introduction à l'informatique

I.1. Objectif du chapitre

À l'issue de ce chapitre, l'étudiant sera capable de :

- Se familiariser avec l'environnement informatique et ses concepts fondamentaux
- Identifier et décrire les principaux composants matériels et logiciels d'un ordinateur
- Comprendre et appliquer les systèmes de codage de l'information
- Évaluer et manipuler les expressions arithmétiques, relationnelles et logiques

I.2. Définition de l'Informatique

L'informatique est la science qui s'occupe du traitement automatique de l'information à l'aide de systèmes électroniques programmables. Le terme « informatique » est formé par la contraction des mots « information » et « automatique », introduit dans les années 1960 pour désigner cette nouvelle discipline.

Elle englobe un ensemble de théories, de méthodes et de techniques liées à la collecte, au stockage, au traitement et à la transmission de l'information, et trouve des applications dans de nombreux domaines tels que la médecine, l'industrie, l'éducation, la communication et la recherche scientifique.

I.2.1. Rôle de l'informatique

L'informatique joue un rôle fondamental dans le monde moderne. Elle intervient principalement à trois niveaux :

- La conception et la construction des ordinateurs : elle couvre le développement des composants matériels et logiciels qui constituent les systèmes informatiques.
- Le fonctionnement et la maintenance des ordinateurs : elle assure le bon fonctionnement des systèmes, leur fiabilité, leur sécurité et leur mise à jour continue.
- L'exploitation des ordinateurs : elle englobe l'utilisation des outils informatiques dans les différents domaines d'activités tels que la gestion, l'industrie, la médecine, l'enseignement et la recherche scientifique.

I.3. Evolution de l'informatique et des ordinateurs

I.3.1. Evolution de l'informatique

L'histoire de l'informatique est marquée par la volonté constante des hommes d'automatiser certaines tâches autrefois réalisées manuellement, en particulier les opérations de calcul. Cette quête d'automatisation a conduit, au fil des siècles, à des avancées technologiques majeures qui ont profondément transformé notre façon de traiter et d'échanger l'information.

Le tableau I.1 ci-dessous présente les principales étapes de cette évolution.

Tableau I.1 : Evolution de l'informatique

Date	Inventeur	Machine	Fonction
1642	Blaise Pascal	Pascaline	Première machine à calculer mécanique, capable d'effectuer des additions et des soustractions sur des nombres de six chiffres.
1822-1832	Charles Babbage	Machine à Différences *(Difference Engine)*	Conçue pour effectuer des calculs numériques complexes et générer automatiquement des tables numériques.
1887	Hermann Hollerith	Machine mécanographique (Calculateur à cartes perforées)	Développée pour accélérer le traitement des données lors du recensement américain de 1890, marquant l'invention de la carte perforée.
1945 (inauguré en 1946)	John P. Eckert & John W. Mauchly	**ENIAC** (Electronic Numerical Integrator and Computer)	Premier calculateur numérique électronique programmable. Composé de 18 000 tubes électroniques et pesant 30 tonnes, il était capable de multiplier deux nombres de 10 chiffres en moins d'une milliseconde.

1945	John Von Neumann	**EDVAC** (Electronic Discrete Variable Automatic Computer)	Premier ordinateur conçu sur le principe du programme enregistré en mémoire, posant les bases de l'architecture moderne des ordinateurs.

I.3.2. Evolution des ordinateurs

I.3.2.1. Définition de l'ordinateur

Un ordinateur est une machine automatique programmable, utilisée pour le traitement de l'information.

I.3.2.2. Générations des ordinateurs

- **1ère Génération (1945 - 1957) :** C'est les ordinateurs construits sur la base des "*tubes électroniques*" (aussi appelés "*tubes à vide*". **Exemple** : ENIAC (1945) et IBM 701 (1952).
- **2ème Génération (1958 - 1963) :** Elle est définie par l'invention du "*Transistor"* en 1947 (le transistor est un composant électronique capable de réguler le courant). Ces ordinateurs sont 100 fois plus rapides que ceux de la 1ère génération et consomment moins d'énergie électrique et sont moins volumineux (occupent moins d'espace). **Exemples**: PDP I (1er ordinateur de 2ème génération (1960)), IBM 7030 (1961).
- **3ème Génération (1964 - 1971) :** Elle est définie après l'apparition du "*Circuit Intégré*" en 1958, les ordinateurs de cette génération utilisent les transistors et les circuits intégrés. **Exemple**: IBM 360 (1964), DEC PDP8 (1964).
- **4ème Génération (1971 - 1980) :** Elle survient après l'invention du "*microprocesseur*". Le premier microprocesseur est fabriqué par la société INTEL en 1971. On parle, désormais, de microordinateurs. **Exemple**: MICRAL 8008 (1973), ALTAIR 8800 (1975).
- **5ème Génération (1980 - 2000) :** Les ordinateurs entrent dans les foyers, on parle alors d'informatique familiale (c'est la naissance des ordinateurs personnels (PC)). **Exemple** : IBM PC (1981), Intel Pentium (1993).
  Aussi, les systèmes d'exploitation MS-DOS et Mac OS font leur apparition.

- **6ème Génération (2000 – aujourd'hui) :** On assiste, dans cette génération, à l'apparition des ordinateurs portables et des réseaux sans fil.
  Actuellement l'évolution des ordinateurs tend à exploiter le laser, les fibres optiques, la biochimie et même l'informatique moléculaire.

I.4. Architecture et fonctionnement d'un ordinateur

L'architecture et le fonctionnement d'un ordinateur constituent des notions fondamentales pour la compréhension d'un système informatique.

L'architecture d'un ordinateur désigne l'organisation générale ainsi que la structure interne de ses différents composants matériels, tels que le processeur, la mémoire centrale, les unités de stockage et les périphériques d'entrée/sortie. Cette organisation permet d'assurer le traitement, le stockage et l'échange des informations nécessaires à l'exécution des différentes tâches informatiques.

Le fonctionnement d'un ordinateur repose sur l'interaction entre ces composants, qui coopèrent afin d'exécuter les instructions des programmes et de traiter les données de manière automatique et rapide.

La Figure I.1 illustre l'architecture générale d'un ordinateur.

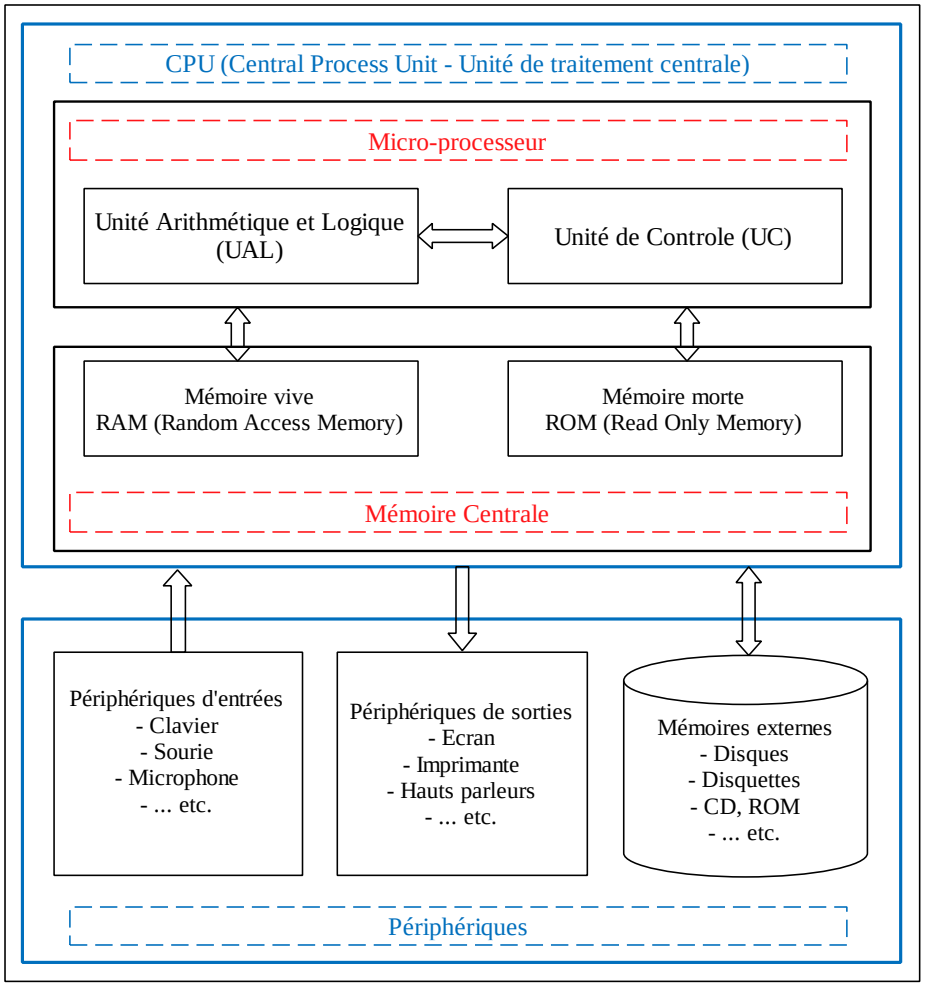


Figure I.1 : Architecture générale d'un ordinateur

CPU : Central Process Unit (Unité de traitement centrale)
UAL : Unité Arithmétique et Logique

UC : Unité de Contrôle ou de commande
MC : Mémoire Centrale

Un ordinateur est constitué, d'une manière générale, d'une Unité Centrale de Traitement, d'une mémoire centrale et d'un ensemble de périphériques.

a) L'Unité Centrale de Traitement / Processeur

Le processeur, également appelé microprocesseur ou CPU *(Central Processing Unit)*, constitue le cœur de l'ordinateur. Il est chargé d'interpréter et d'exécuter les instructions des programmes. Il comprend deux unités fondamentales :

- **L'Unité de Contrôle (UC)** : responsable de la lecture et du décodage des instructions stockées en mémoire centrale, elle coordonne et synchronise l'ensemble des opérations du processeur.
- **L'Unité Arithmétique et Logique (UAL)** : également appelée unité de traitement, elle exécute toutes les opérations arithmétiques, logiques et relationnelles. Elle assure également les échanges de données (réception et transmission) avec la mémoire vive.

Ces deux unités communiquent non seulement avec la mémoire centrale, mais également avec les différents périphériques du système.

b) La Mémoire Centrale (ou Principale)

La mémoire centrale est constituée de deux types de mémoires complémentaires :

- **La RAM** *(Random Access Memory — Mémoire Vive)* : elle assure le stockage temporaire des données et des programmes en cours d'exécution. Elle est dite **volatile**, car son contenu est perdu en cas de coupure d'alimentation électrique.
- **La ROM** *(Read Only Memory — Mémoire Morte)* : contrairement à la RAM, elle est **non volatile** et conserve son contenu même en l'absence d'alimentation. Elle ne peut être que lue (ou très rarement modifiée) et contient les programmes indispensables au démarrage du système, avant le chargement du système d'exploitation en RAM.

La capacité de la mémoire centrale se mesure actuellement en milliers de mégaoctets. Les principales unités de mesure sont les suivantes :

Unité	**Équivalence**
1 bit	Unité de base (valeur : 0 ou 1)
1 octet	8 bits
1 Kilo-octet (Ko)	2^{10} octets = 1 024 octets
1 Méga-octet (Mo)	2^{20} octets = 1 024 Ko
1 Giga-octet (Go)	2^{30} octets = 1 024 Mo

Remarque : Le **bit** est l'abréviation de *"binary digit"* (chiffre binaire). C'est l'unité élémentaire d'information dans tout système numérique, pouvant prendre deux valeurs : **0** ou **1**.

c) Les Périphériques

Les périphériques désignent l'ensemble des équipements connectés à l'unité centrale permettant les échanges d'informations entre l'ordinateur et le monde extérieur. Ils se classent en trois catégories :

- **Périphériques d'entrée** : ils permettent de transmettre des données vers l'ordinateur. Exemples : le clavier, la souris, le microphone, la webcam, le scanner, etc.
- **Périphériques de sortie** : ils permettent à l'ordinateur de restituer des données vers l'extérieur. Exemples : l'écran, l'imprimante, les haut-parleurs, le vidéoprojecteur, etc.
- **Périphériques d'entrée/sortie** : ils permettent à la fois de transmettre et de recevoir des données. Également appelés **périphériques de stockage** ou **mémoires auxiliaires**, ils assurent la conservation permanente des données. Exemples : la clé USB, le disque dur externe, le CD-ROM, la carte mémoire, etc.

I.5. Composition d'un ordinateur (Matériel et Logiciel)

Un ordinateur est constitué de deux parties essentielles et complémentaires : le **matériel** *(hardware)* et le **logiciel** *(software)*. Le matériel désigne l'ensemble des composants physiques de la machine, tandis que le logiciel regroupe l'ensemble des programmes et instructions qui permettent de les exploiter.

I.5.1. Partie matérielle (Hardware)

Le matériel d'un ordinateur désigne l'ensemble des composants physiques qui constituent la machine, incluant les éléments électroniques, mécaniques et électriques nécessaires à son fonctionnement. On distingue généralement les principales parties suivantes :

- Carte Mère :

Carte principale sur laquelle on retrouve tous les composants nécessaires au fonctionnement d'un ordinateur, dont le processeur, la mémoire centrale, les bus et les connecteurs d'extension destinés à recevoir des cartes d'extension (carte graphique externe, carte son, carte réseau, etc.).	

- Processeur :

Processeur dont tous les éléments sont miniaturisés et rassemblés sur une puce en un seul ou en plusieurs circuits intégrés. Le processeur remplit les fonctions d'unité centrale dans un micro-ordinateur. Un ventilateur est utilisé pour réduire sa température souvent élevée et réduire les risques de surchauffe.	

- Mémoire Vive :

Mémoire du travail du processeur, tout programme exécuté doit être chargé sur la mémoire vive, dans laquelle les instructions peuvent lire ou écrire des données.	

- Disque Dur :

Support de mémoire de masse d'accès rapide qui est composé d'un ou de plusieurs disques magnétiques généralement solidaires d'une mécanique, et ayant une très grande capacité de stockage.	

- Carte graphique :

Carte d'extension servant à stocker et à interpréter les images reçues par l'ordinateur et à mettre à jour continuellement l'écran du moniteur. Elle n'est pas obligatoire sur certains ordinateurs qui en possèdent une incorporée dans la carte mère. Mais pour de meilleurs performances il est conseillé d'en installer une.	

- Connecteurs et Ports :

Ces ports permettent de relier des périphériques externes (clavier, souris, etc.) au boîtier de l'unité centrale. Il y a plusieurs type de ports : Port USB, port parallèle, port série, etc.	

- Écran :

Périphérique de sortie, il permet d'afficher les toute sorte de données. Il y a deux mode d'affichage : mode texte et mode graphique. Il y a des écran tactile permettant d'émettre des commandes via de simples touches de doigts, dans ce cas, l'écran représente aussi un périphérique d'entrée.	

- Clavier :

Périphérique permettant d'entrer des données sous forme textuelle vers le CPU. Donc il représente un périphérique de sortie.	

- **Sourie :**

Permet de lancer des commandes à travers de simples cliques : clique gauche pour la sélection, double clique gauche pour exécuter un programme ou ouvrir un fichier, clique droit affiche le menu contextuel, etc.	

- **Imprimante :**

Périphérique de sortie permettant d'afficher des données (textuelles, images ou graphes) sur du papier.	

La figure I.2 résume la partie hardware de l'ordinateur

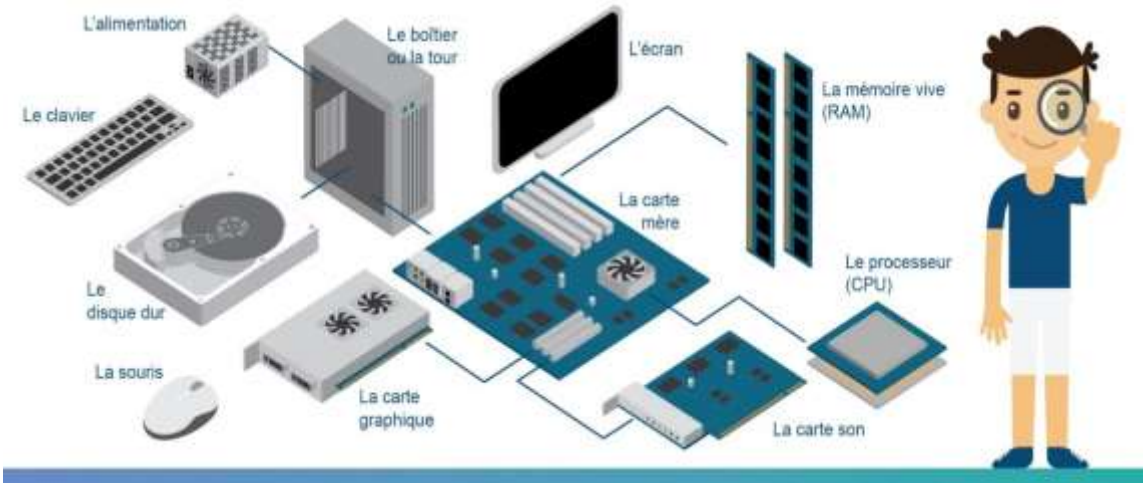


Figure I.2 : Partie hardware de l'ordinateur

I.5.2. Partie logicielle (Software)

Le logiciel désigne l'ensemble des programmes, instructions et données qui permettent à l'ordinateur de fonctionner et d'accomplir des tâches spécifiques. Il constitue la composante immatérielle du système informatique, par opposition au matériel. On distingue trois grandes catégories de logiciels :

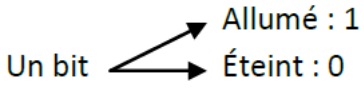
- **Logiciels de base (Système d'exploitation)** : Le système d'exploitation est le logiciel fondamental qui assure la gestion des ressources matérielles de l'ordinateur et sert d'interface entre l'utilisateur et la machine. Sans lui, aucun autre logiciel ne peut fonctionner. Exemples : Windows, Linux, macOS, Android, etc.
- **Langages de programmation** : Ce sont des outils formels utilisés par les développeurs pour concevoir et écrire des programmes informatiques. Chaque langage possède sa propre syntaxe et est adapté à des domaines d'application spécifiques. Exemples : C/C++, Java, Python, Pascal, Fortran, MATLAB, etc.
- **Logiciels d'application** : Ce sont des programmes conçus pour répondre à des besoins spécifiques de l'utilisateur final. Ils s'exécutent au-dessus du système d'exploitation et couvrent un large éventail de fonctionnalités : traitement de texte, tableur, retouche d'image, conception assistée par ordinateur, navigation sur internet, lecture multimédia, jeux, etc. Exemples : Microsoft Office, Adobe Photoshop, AutoCAD, VLC Media Player, etc.

I.6. Les systèmes de codage des informations

Toute information traitée par un ordinateur, qu'elle soit numérique, textuelle, sonore, visuelle ou vidéo, est représentée sous forme de séquences de deux chiffres : **0** et **1**. Ces chiffres sont appelés **bits** *(BInary digiT)* et constituent la base de toute représentation numérique de l'information.

En effet, un ordinateur fonctionne sur le principe de deux états électriques distincts : la présence de courant (état 1) et l'absence de courant (état 0). C'est la seule forme d'information qu'un ordinateur est capable d'interpréter directement. Le bit représente ainsi l'unité élémentaire d'information en informatique, c'est-à-dire la plus petite unité de mesure existante dans un système numérique.

Par conséquent :



I.6.1. Les systèmes de numérotation

Définition : Un système de numérotation est un ensemble de règles et de symboles permettant de représenter tout entier naturel N dans une base b donnée. Tout système de numérotation est défini par deux éléments fondamentaux :

1. **La base (b) :** le nombre de symboles distincts utilisés dans le système.
2. **Les symboles :** l'ensemble des chiffres et/ou des lettres qui composent le système et permettent de représenter toute valeur numérique.

En informatique, quatre systèmes de numérotation sont principalement utilisés, présentés dans le tableau ci-dessous :

Système	Base	Symboles	Nombre de symboles du système
Décimal	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9	10
Binaire	2	0, 1	2
Octal	8	0, 1, 2, 3, 4, 5, 6, 7	8
Hexadécimal	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F	16

❖ **Remarque 1 :** Dans le système hexadécimal, des lettres sont utilisées pour représenter les valeurs supérieures à 9 : $(A)_{16} = (10)_{10}$, $(B)_{16} = (11)_{10}$, $(C)_{16} = (12)_{10}$, $(D)_{16} = (13)_{10}$, $(E)_{16} = (14)_{10}$, $(F)_{16} = (15)_{10}$.

❖ **Remarque 2 :** Le nombre de symboles d'un système de numérotation est toujours égal à sa base : Nombre de symboles = b.

- **Notation**

Un nombre quelconque Nb exprimé dans une base b sera noté comme suit :

$$Nb = (a_{n-1}\ a_{n-2} \dots\ a_1\ a_0)_b$$

avec :

b : la base du système de numérotation

a_i : symbole du système de numérotation, *i=0,…,n-1 et* $a_i < b$

Exemples :

$Nb_1 = (1995)_{10}\ avec\ a_3 = 1, a_2 = 9, a_1 = 9, a_0 = 5$

$Nb_2 = (243)_8\ avec\ \ a_2 = 2, a_1 = 4, a_0 = 3$

$Nb_3 = (1011010)_2\ avec\ a_6 = 1, a_5 = 0, a_4 = 1, a_3 = 1, a_2 = 0, a_1 = 1, a_0 = 0$

$Nb_4 = (BAC)_8\ avec\ \ a_2 = B, a_1 = A, a_0 = C$ Cette notation est erronée, car le nombre contient un symbole supérieur ou égale à la base

$Nb_5 = (248)_8$ Cette notation est erronée, car le nombre contient un symbole supérieur ou égale à la base

❖ **Remarques :**

- Quand la base n'est pas mentionnée, on considère qu'on est en base 10.
- Dans une base b, nous avons b chiffres : 0, 1, 2, …., (b-1).

I.6.2. Conversion d'un nombre d'un système à un autre

a) Conversion de la base b (base 2, 8, 16,…etc) vers la base 10

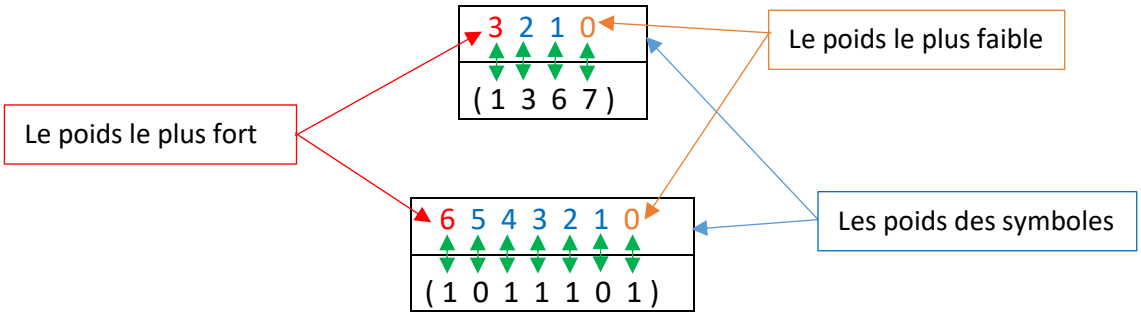
Pour convertir un nombre $Nb = (a_{n-1}\ a_{n-2} \dots\ a_1\ a_0)_b$ de la base b vers la base 10, on effectue le calcul suivant :

$$(Nb)_b = (a_{n-1} * b^{n-1} + a_{n-2} * b^{n-2} \dots + a_1 * b^1 + a_0 * b^0)_{10}$$

$$(Nb)_b = \sum_{i=0}^{n-1} a_i * b^i$$

Elle correspond à la somme des produits de chaque symbole du nombre par le poids correspondant.

Le poids des symboles : il s'obtient en numérotant les symboles à partir de la droite vers la gauche, en commençant du 0.



Exemple 1 :

$(\mathbf{1367})_{\mathbf{8}} = (?)_{\mathbf{10}}$

$(1367)_8 = (1 * 8^3 + 3 * 8^2 + 6 * 8^1 + 7 * 8^0)_{10}$

$(1367)_8 = (759)_{10}$

Exemple 2 :

$(\mathbf{1011101})_{\mathbf{2}} = (?)_{\mathbf{10}}$

$(1011101)_2 = (1 * 2^6 + 0 * 2^5 + 1 * 2^4 + 1 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0)_{10}$

$(1011101)_2 = (93)_{10}$

Exemple 3 :

$(\boldsymbol{2B3})_{\boldsymbol{16}} = (?)_{\boldsymbol{10}}$

$(2B3)_{16} = (2 * 16^2 + B * 16^1 + 3 * 16^0)_{10}$

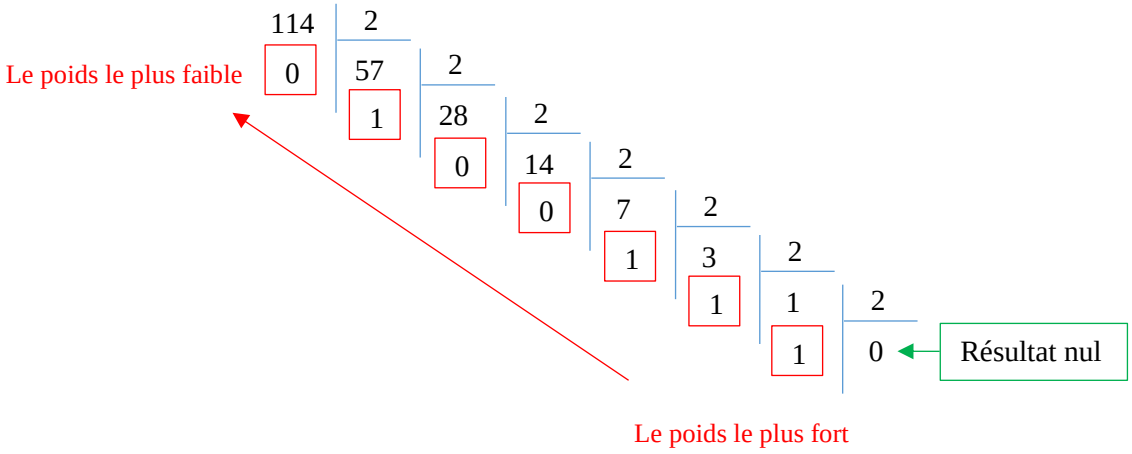
$(2B3)_{16} = (691)_{10}$

b) Conversion de la base 10 vers la base b (base 2, 8, 16,…etc)

Soit Nb un nombre exprimé dans la base 10, pour trouver son équivalent en base b, on applique la méthode des **divisions successives sur b**, jusqu'à l'obtention d'un résultat nul. Puis, on récupère les restes des divisions dans le sens inverse, i.e. le dernier reste trouvé représentera le poids le plus fort et le premier reste trouvé sera le poids le plus faible.

Exemple 1 :

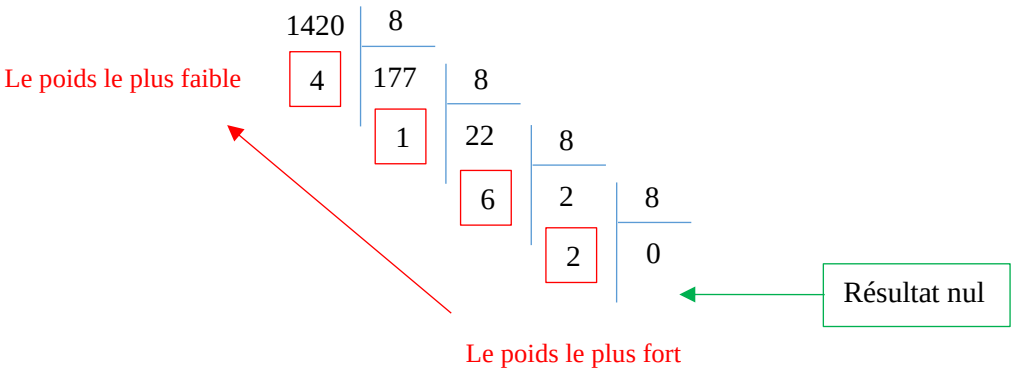
- $(\boldsymbol{114})_{\boldsymbol{10}} = (?)_{\boldsymbol{2}}$



$(114)_{10} = (1110010)_2$

Exemple 2 :

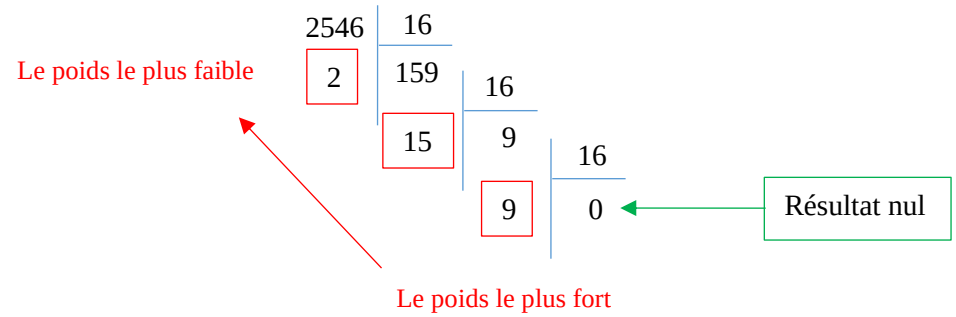
- $(\boldsymbol{1420})_{\boldsymbol{10}} = (?)_{\boldsymbol{8}}$



$(1420)_{10} = (2614)_8$

Exemple 3 :

- $(\mathbf{2546})_{\mathbf{10}} = (?)_{\mathbf{16}}$



$(2546)_{10} = (9F2)_{16}$

c) Conversion de la base 8 vers la base 2

Pour convertir un nombre Nb exprimé en base 8 vers la base 2, nous procédons comme suit:

$$8 = 2^3$$

Il faut donc utiliser **3** bits pour exprimer un seul chiffre octal en binaire.

La représentation des chiffres de la base 8 vers le binaire est comme suit :

- $(7)_8 = (1 * 2^2 + 1 * 2^1 + 1 * 2^0)_2 = (111)_2$
- $(4)_8 = (1 * 2^2 + 0 * 2^1 + 0 * 2^0)_2 = (100)_2$
- $(3)_8 = (0 * 2^2 + 1 * 2^1 + 1 * 2^0)_2 = (011)_2$

Chiffre en octal	Chiffre équivalent en binaire (2^2 2^1 2^0)
0	0 0 0
1	0 0 1
2	0 1 0
3	0 1 1
4	1 0 0
5	1 0 1
6	1 1 0
7	1 1 1

Exemple :

- $(743)_8 = (111\ \ 100\ \ 011)_2$

d) Conversion de la base 16 vers la base 2

Pour convertir un nombre Nb exprimé en base 16 vers la base 2, nous procédons comme suit:

$$16 = 2^4$$

Il faut donc utiliser **4** bits pour exprimer un seul chiffre hexadécimal en binaire.

La représentation des chiffres de la base 16 vers le binaire est comme suit :

Chiffre en hexadécimal	Chiffre équivalent en binaire (2^3 2^2 2^1 2^0)
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1

Chiffre en hexadécimal	Chiffre équivalent en binaire (2^3 2^2 2^1 2^0)
8	1 0 0 0
9	1 0 0 1
A	1 0 1 0
B	1 0 1 1
C	1 1 0 0
D	1 1 0 1
E	1 1 1 0
F	1 1 1 1

- $(7)_{16} = (0 * 2^3 + 1 * 2^2 + 1 * 2^1 + 1 * 2^0)_2 = (0111)_2$
- $(B)_{16} = (1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0)_2 = (1011)_2$
- $(3)_{16} = (0 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0)_2 = (0011)_2$
- $(A)_{16} = (1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 0 * 2^0)_2 = (1010)_2$

Exemple :

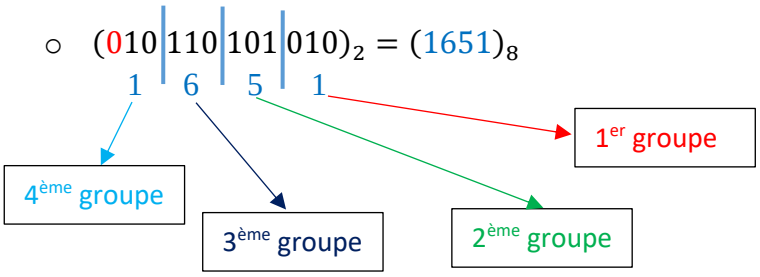
- $(7B3A)_{16} = (0111\ 1011\ 0011\ 1010)_2$

e) Conversion de la base 2 vers la base 8

Pour trouver l'équivalent d'un nombre binaire en octal, il suffit de former des **groupes de 3 bits** chacun (Puisque **8 = 2^3**), en commençant du poids le plus faible (à partir de la droite), si le dernier groupe formé possède moins de 3 bits, il suffit de rajouter des 0, puis calculer l'équivalent en octal de chaque groupe.

Exemple :

- $(10110101010)_2 = (?)_8$
- $(010|110|101|010)_2 = (1651)_8$



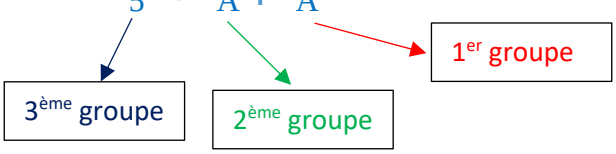
f) Conversion de la base 2 vers la base 16

Pour trouver l'équivalent d'un nombre binaire en Hexadécimal, il suffit de former des **groupes de 4 bits** chacun (Puisque **16 = 2^4**), en commençant du poids le plus faible (à partir

de la droite), si le dernier groupe formé possède moins de 4 bits, il suffit de rajouter des 0, puis calculer l'équivalent en Hexadécimal de chaque groupe.

Exemple :

- $(10110101010)_2 = (?)_{16}$
- $(0101\,1010\,1010)_2 = (5AA)_{16}$



g) Conversion de la base 16 vers la base 8

Pour convertir un nombre Nb exprimé en base 16 vers la base 8 ou vice versa, nous devons **passer par une base intermédiaire** tel que le **décimal** ou le **binaire**, mais le passage par le binaire est beaucoup plus simple.

Exemple :

- $(C9F)_{16} = (1100\ 1001\ 1111)_2$
- $(110\,|\,010\,|\,011\,|\,111)_2 = (6237)_8$
  6 2 3 7

I.6.3. Le code D.C.B. (Decimal Coded Binary – Décimal codé binaire)

Le code DCB consiste à convertir chaque chiffre décimal en un nombre binaire sur 4 positions.

Exemple :

Le nombre décimal 378 est codé en DCB comme suit : 0101 0111 1000

Dans ce code, chaque chiffre décimal est remplacé par 4 chiffre binaires (comme dans le cas de hexadécimal).

I.6.4. La codification Alphanumériques

Les chiffres, lettres, signes de ponctuation, les symboles mathématiques, etc., sont représentées généralement en utilisant le code normalisé à 8 positions binaires. On utilise pour cela le code EBDIC (Extended Binary Coded Decimal International Code) ou le code ASCII (American Standard Code Information Interchange).

Exemples :

Tableau I.2 : Exemples de codification Alphanumériques

Lettre / Signe / Symbole	En EDCDIC	En ASCII
A (Majuscule)	11000001	10100001
D (miniscule)	10000010	11100100
=	01111110	01011101

Exercice : *(Systèmes de numérotation)*

Effectuer les conversions suivantes :

$2022 = (?)_2$;

$(1011001101)_2 = (?)_{10}$;

$(1011001101)_2 = (?)_8 = (?)_{16}$;

$(32103)_4 = (?)_2$;

$(37163)_8 = (?)_2$;

$(379)_{10} = (?)_{16}$;

$(3A2D)_{16} = (?)_{10}$;

$(4D5B)_{16} = (?)_8$

Corrigé de l'exercice : *(Systèmes de numérotation)*

Effectuer les conversions suivantes :

$2022 = (11111100110)_2$;

$(1011001101)_2 = (717)_{10}$;

$(1011001101)_2 = (1315)_8 = (2CD)_{16}$;

$(32103)_4 = (1110010011)_2$;

$(37163)_8 = (011111001110011)_2$ oubien $(11111001110011)_2$;

$(379)_{10} = (17B)_{16}$;

$(3A2D)_{16} = (14893)_{10}$;

$(4D5B)_{16} = (46533)_8$

I.7. Les expressions arithmétiques, relationnelles et logiques

I.7.1. Expressions arithmétiques

C'est une combinaison d'opérande(s) (valeur, constante, variable) et d'opération(s) arithmétiques. Le Tableau I.3 résume ces opérations.

Tableau I.3 : Les opérations arithmétiques

Opération	**Opérateur**
Addition	+
Soustraction	-
Multiplication	*
Division	/
Modulo (reste de la division entière)	%

I.7.2. Expressions relationnelles

En langage C, les opérateurs relationnels permettent de comparer deux valeurs. Le résultat d'une comparaison est 1 si elle est vraie et 0 si elle est fausse. Le tableau I.4 résume les principaux opérateurs relationnels utilisés en programmation.

Tableau I.4 : Les opérateurs relationnels

Signification	**Opérateur**
Egal à	==
Strictement inférieur à	<
Inférieur ou égal à	<=
Strictement supérieur à	>
Supérieur ou égal à	>=
Différent de	!=

❖ **Remarque importante :**

- = est l'opérateur d'affectation : a = 5;
- == est l'opérateur de comparaison : a == 5

I.7.3. Expressions logiques (booléennes)

En langage C, les expressions logiques (booléennes) permettent de combiner ou de modifier des conditions. Elles renvoient 1 si le résultat est vrai et 0 s'il est faux.

Tableau I.5 : L'opérateur **ET logique (&&)**.

A	B	A **&&** B
False	False	False
False	True	False
True	False	False
True	True	True

Tableau I.6 : L'opérateur **OU logique (||)**

A	B	A || B
False	False	False
False	True	True
True	False	True
True	True	True

Tableau I.7 : L'opérateur **NON logique (!)**

A	! A
False	True
True	False

I.7.4. Les fonctions

Le Tableau I.8 présente une liste non-exhaustive de fonctions standards (ou prédéfinies) applicables sur des entiers ou des réels.

Tableau I.8 : Quelques fonctions standards en langage C

Fonction	**Description**	**Exemple**
abs(x)	Valeur absolue d'un entier	abs(-5) → 5
sqrt(x)	Racine carrée	sqrt(25) → 5
pow(x,y)	Puissance x^y	pow(2,3) → 8
sin(x)	Sinus de x (en radians)	sin(x)
cos(x)	Cosinus de x (en radians)	cos(x)
tan(x)	Tangente de x (en radians)	tan(x)
exp(x)	Exponentielle e^x	exp(1)
log(x)	Logarithme népérien	log(x)
log10(x)	Logarithme décimal	log10(100) → 2
ceil(x)	Plus petit entier supérieur ou égal à x	ceil(3.2) → 4
floor(x)	Plus grand entier inférieur ou égal à x	floor(3.8) → 3
fabs(x)	Valeur absolue d'un réel	fabs(-3.5) → 3.5
fmod(x,y)	Reste de la division de deux réels	fmod(7.5,2) → 1.5

❖ **Remarque :** les fonctions de la bibliothèque <math.h> nécessitent généralement l'inclusion de cette bibliothèque au début du programme :

I.7.5. La priorité dans les expressions en langage C

Lorsqu'une expression contient plusieurs opérateurs, le langage C respecte un **ordre de priorité** pour déterminer dans quel ordre les opérations doivent être effectuées.

Tableau I.9 : Priorité des opérateurs

Priorité	Opérateurs	Description	Associativité
1	`(); fonctions`	Parenthèses, fonctions	Gauche → droite
2	`!`	NON logique	Droite → gauche
3	`* / %`	Multiplication, division, modulo	Gauche → droite
4	`+ -`	Addition, soustraction	Gauche → droite
5	`< <= > >=`	Opérateurs relationnels	Gauche → droite
6	`== !=`	Égalité, différence	Gauche → droite
7	`&&`	ET logique	Gauche → droite
8	`||`	OU logique	Gauche → droite
9	`= += -= *= /= %=`	Affectation	Droite → gauche

I.7.6. Évaluation des expressions

L'évaluation d'une expression consiste à calculer, au fur et à mesure, les résultats des calculs jusqu'à obtenir un résultat finale. Cela se fait en plusieurs étapes :

- Écrire l'expression sous forme linéaire (Il faut noter qu'en algorithmique, les expressions s'écrivent sous forme linéaire: $\frac{(x+z)}{(y*2)} \rightarrow (x+z)/(y*2)$;
- Remplacer les identifiants (c'est à dire les noms) des variables et des constantes par leurs valeurs ;
- Évaluer (Calculer) étape par étape chacune des sous-expressions en commençant par les sous-expressions qui sont dans les parenthèses les plus internes.
- Indiquer à chaque calcul, le rang d'évaluation**.**

❖ **Remarque :** Si les opérateurs ont le même rang de priorité, l'évaluation se fait de gauche à droite.