Algorithmique et Programmation en langage C

Pr. Abderrahmane SADIQ
Filière: GI
Semestre: S3
Faculté Polydisciplinaire Taroudant
Année Universitaire: 2024-2025

Généralités: Questions ?

Qu'est ce qui fait que votre ordinateur est bien ou pas ?

Vitesse du processeur.

Mémoire Vive (RAM).

Disques durs.

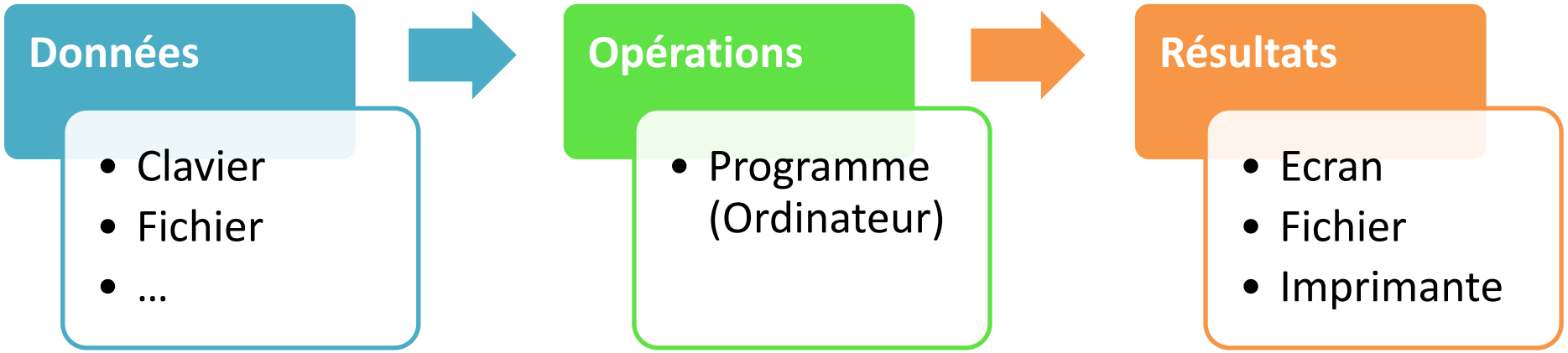
Autres

- Périphériques, cartes de gestions de périphériques.

Généralités

Que fait un ordinateur ?

- DES CALCULS (SIMPLES).



Depuis 1950 (Von Neumann), les opérations sont stockées en mémoire dans l'ordinateur : la machine est programmable.

Généralités

Pour qu'un programme « tourne » (double clic)

- Le programme est chargé « Dans la mémoire de l'ordinateur.
- Le programme s'exécute et travaille sur des données également Chargées en mémoire.
- Les sorties sont stockées en mémoire puis redistribuées (écran, fichier, imprimante).

Plus votre ordinateur a de mémoire, plus il peut faire de grandes choses (exécuter des gros programmes et travailler sur beaucoup de données)

Introduction : Algorithmique et Programmation

Programmation

- Programmer c'est expliquer à une machine comment résoudre un problème.
- La programmation comprend trois grandes étapes :
  - Modélisation: énoncer le problème en termes explicites.
  - Résolution: construire un ***algorithme*** qui spécifie, par une séquence d'actions, comment résoudre le problème.
  - Codage: traduire l'algorithme dans un langage de programmation qui peut être traité la machine.

Modélisation

- Enoncé d'un problème
- Un problème est spécifié par des données et un résultat :
  - Données :c'est l'ensemble des objets (ingrédients) que nous avons à notre disposition
  - Résultat :c'est la solution que nous cherchons à obtenir
- Instance: Une instance de problème consiste en une valeur associée à chaque donnée du problème.

Modélisation

- Exemple : le problème est divisible par
  - Données : deux nombres entiers *n et d*
  - Résultat : “oui” si *n est divisible par d, et “non” sinon*
- Une instance du problème
  - pour les valeurs *n = 20 et d = 10,*
  - le résultat est la valeur “vrai”

Modélisation

- Exemple : le problème du tri
  - Données : une séquence de *n nombres réels* $\langle x_1, x_2, \ldots, x_n \rangle$
  - Résultat : une permutation $\langle x'_1, x'_2, \cdots, x'_n \rangle$ de la séquence telle que $x'_1 \leq x'_2 \leq \cdots \leq x'_n$
- Une instance du problème
  - pour la séquence :

  - le résultat est :

Modélisation

- Exemple : le problème de la recherche du chemin le plus court
  - Données : une carte routière, une ville de départ et une ville d'arrivée
  - Résultat : l'itinéraire le plus court (en kilomètres) depuis la ville de départ jusqu'à la ville d'arrivée
- Une instance du problème
  - Pour la carte du maroc, la ville d'Agadir et la ville de Taroudant, le chemin le plus court est :

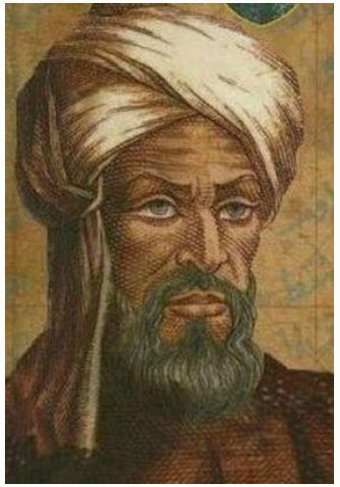
Résolution: Algorithme

Algorithme

- Un algorithme est une séquence d'actions (appelées instructions) qui permet de passer des données du problème au résultat attendu.

Ou

- Une séquence d'instructions qui décrit comment résoudre un problème particulier
- Le mot "algorithme" vient du mathématicien perse Al Khuwarizmi.
- Muhammad Ibn Musa al-Khuwarizm (780-850)
  - mathématicien, géographe, astrologue et astronome perse



Résolution: Algorithme

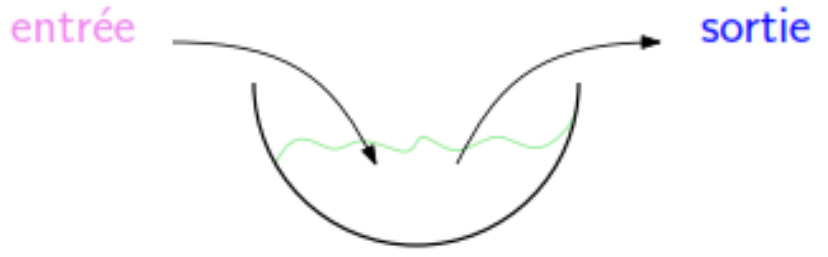
Au départ, il y a un problème

- Comment faire un gâteau au chocolat ??
- Comment trier dans l'ordre croissant une suite de nombres entiers?
- Comment trouver le chemin le plus court entre deux villes

- Comment additionner 2 nombres?

Résolution: Algorithme

- Rappel: Un algorithme est une séquence d'actions (appelées instructions) qui permet de passer des données du problème au résultat attendu.



Résolution: Algorithme

Pour 6 personnes:

200 g de chocolat noir

4 œufs

150 g de sucre

80 g de farine

200 g de beurre

entrée

sortie

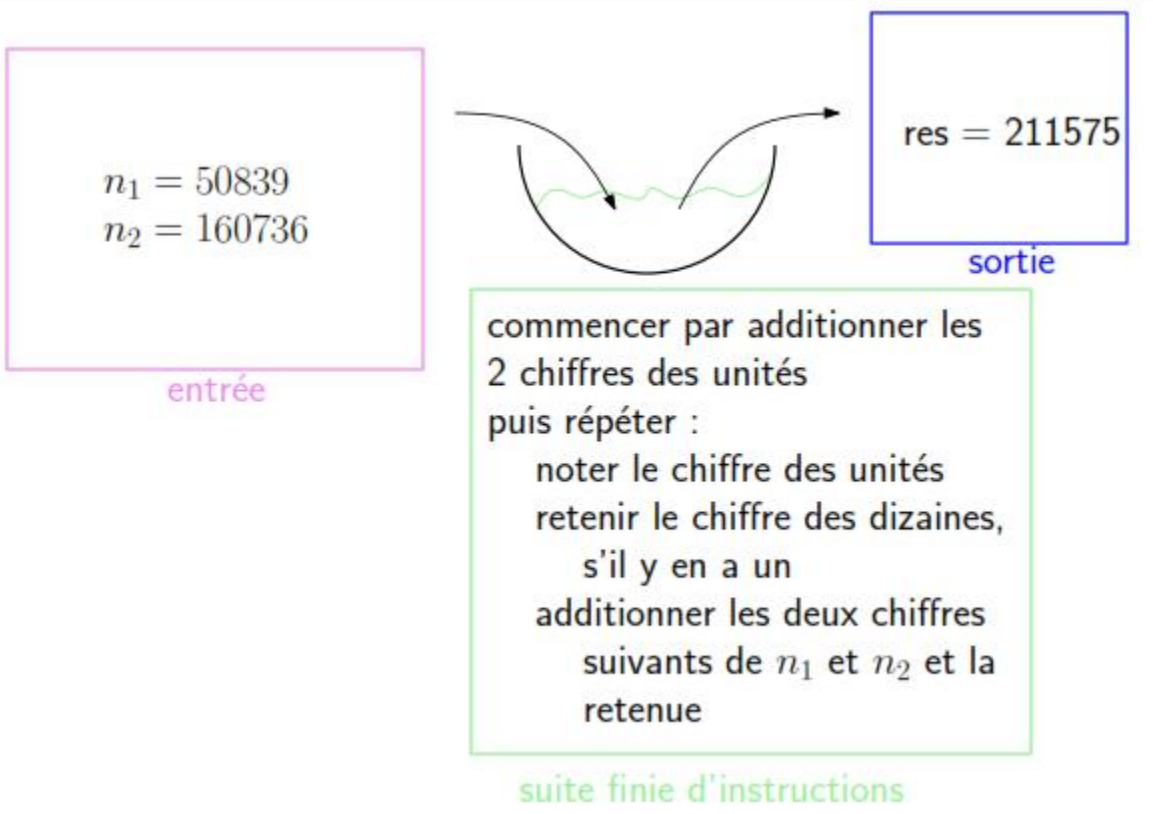
mélanger les jaunes d'œufs avec

...

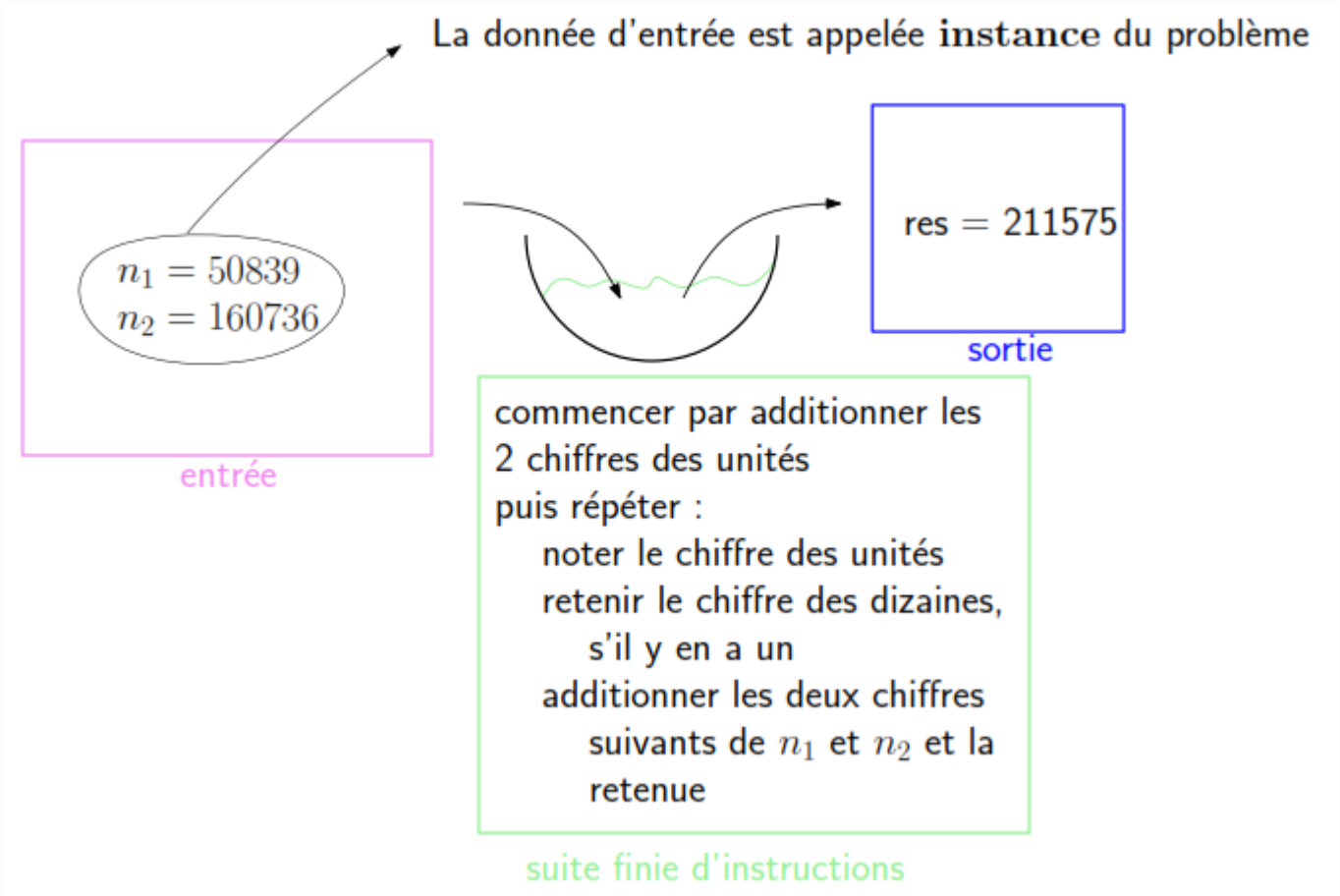
mettre 20 min au four à 180 °

suite finie d'instructions

Résolution: Algorithme



Résolution: Algorithme



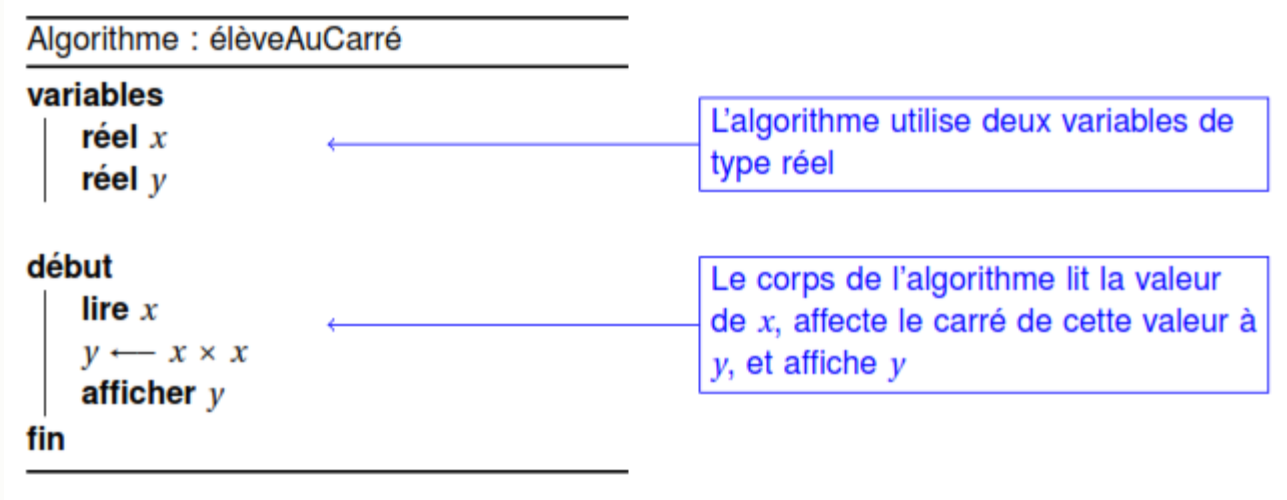
Résolution: Algorithme

- Un algorithme est dit correct si, pour toute instance du problème, l'algorithme s’arrête lorsqu'il produit la bonne donnée de sortie.

- Moyen d’atteindre un but en répétant un nombre fini de fois un nombre fini d’instructions.
- Donc, un algorithme se termine en un temps fini

Résolution : Exemple

- Un algorithme pour le problème: mettre au carré
  - Données : un nombre réel *x*
  - Résultat : le carré de *x*



Résolution: Algorithme

Construire un Algorithme

- Maîtriser l'algorithmique requiert deux qualités :
  - L'intuition: savoir décomposer le problème en une série d'instructions.
  - La rigueur : vérifier que la série d'instructions aboutit bien au résultat attendu.
- Les deux qualités s'acquièrent par l'expérience !

Ecriture Algorithmique

- Un algorithme est un manuel utilisateur pour résoudre un problème. Il ne dépend pas
  - du langage de programmation dans lequel il sera codé
  - de la machine qui exécutera le programme correspondant

Le langage algorithmique (ou pseudo-code) utilise des instructions faciles à comprendre et calculer.

Résolution: Algorithme

Analyser un Algorithme

- Correction: peut-on garantir que pour chaque instance du problème, l'algorithme retourne le résultat attendu ?
- Complexité: de combien de temps et d'espace (mémoire) l'algorithme a-t-il besoin pour résoudre le problème ?

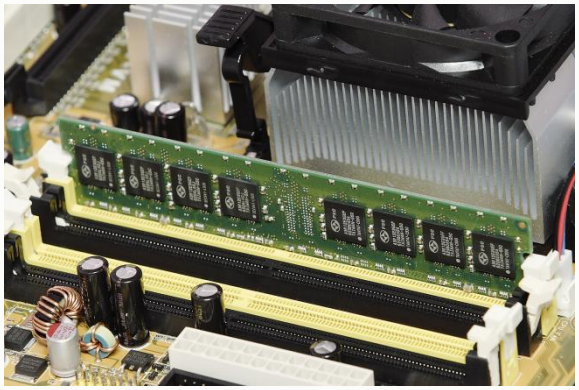
Note

- Durant ce semestre, nous étudierons la correction d'algorithmes simples

Algorithme: Données

Données

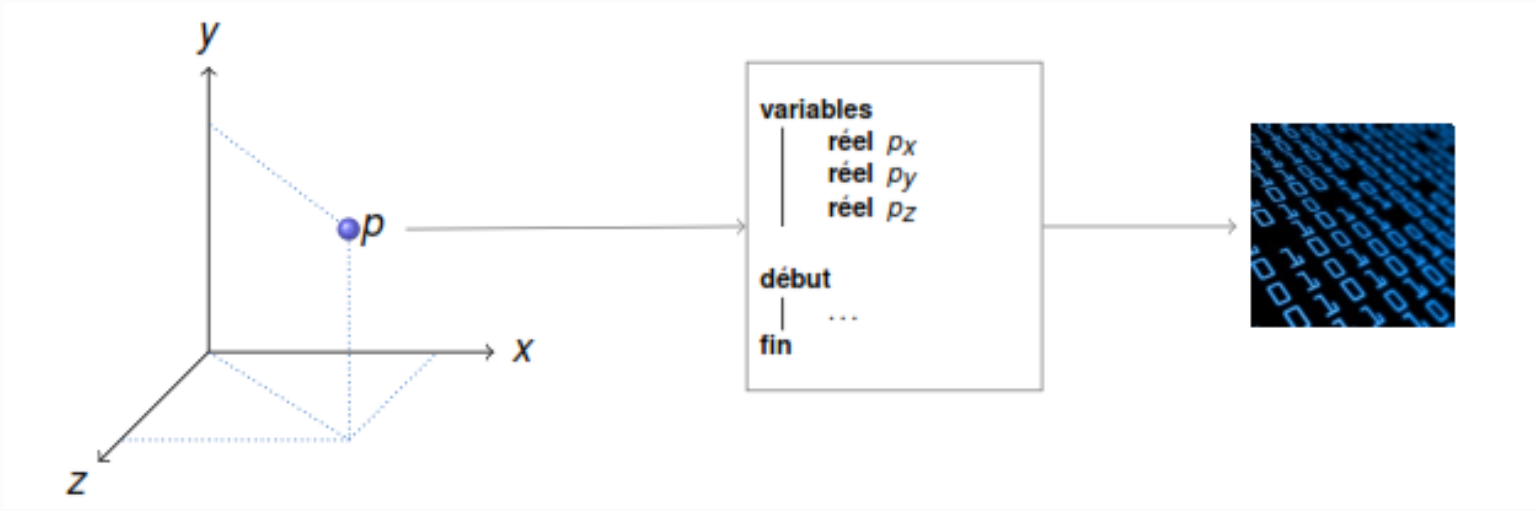
- En informatique, une donnée est la représentation d'un objet abstrait ou concret du monde. Dans une machine chaque donnée est codée et stockée dans une mémoire.



Algorithme: Données

Données

- En informatique, une donnée est la représentation d'un objet abstrait ou concret du monde. Dans une machine chaque donnée est codée et stockée dans une mémoire.



Modalités du cours

- **Pré-requis: Algorithmique et prog 1;** matière fondamentale de l'informatique, nécessaire pour aborder le reste du cours.
- 2h de cours, 2h de TD et 2h de TP par semaine.
- Il faut travailler les cours, TD et TP, et surtout bien les comprendre.

Donc n'hésitez pas à poser des questions à vos enseignants !

Modalités du cours

- Le cours sera disponible après chaque séance.
- Même si vous disposerez des supports, il est utile de prendre des notes pour bien comprendre !.

Avant de commencer !

A noter….

Travailler pour le plaisir, pour évoluer son niveau et pas uniquement pour une NOTE

Objectif …

Ce module vise à . . .

- Introduire un **langage de programmation procédurale : C**
- Appuyer les étapes fondamentales pour **résoudre un problème** via une machine, **organiser et concevoir des solutions,**
- Développer **l'esprit d'analyse et une bonne méthodologie de** programmation : les bonnes habitudes de programmation,
- Développer les aptitudes à parler dans un **langage que la** machine peut comprendre,
- Utiliser une machine pour **produire et non seulement pour** consommer.

Programmer !

OK, but what makes a good programmer ?

- Être **Passionné par les nouvelles technologies,**
- La considérer comme **un loisir plutôt qu'une corvée,**
- Enrichir ses connaissances avec une **grande variété de** technologies et méthodologies ,
- Être "Intelligent" et débrouillard,
- Posséder des "compétences" sociales : être conviviale et facile à **communiquer avec. Si vous ne pouvez pas parler avec un** programmeur, vous ne serez jamais communiquer ses besoins de manière efficace,
- Ne pas se contenter juste de comment utiliser une technologie mais surtout de savoir **ce qui est possible à avoir grâce à cette technologie** !

Langage de programmation C

Plan

- Introduction générale
- **Terminologie**
- Environnement de développement C : Dev-C++
- Premiers programmes en langage C
- Variables : type, portée et durée de vie
- Les fonctions d'entrées-sorties classiques
- Expression et ordre des opérations
- Constante
- Structure de contrôle : si ...alors ...sinon
- Trace d'exécution d'un programme
- Exemple d'utilisation de fonctions existantes
- Tableaux
- Chaîne de caractères et tableaux
- Bloc d'instruction
- Structures de contrôle imbriquées
- Structures de contrôles itératives

Langage machine

Le langage machine est le "langage" de base compréhensible par un ordinateur (utilisé par le processeur), soit une suite de zéros et de uns.

Il faut noter que :

- Le langage machine n'est pas compréhensible facilement par l'humain
- **Il est plus pratique de trouver un langage intermédiaire, compréhensible par l'Homme, qui sera ensuite transformé en langage machine pour être exploitable par le processeur**
- Plus le langage est compréhensible par l'Homme (plus évolué), plus on dira qu'il est de **haut niveau. Ainsi, si le programme est** écrit dans un langage proche de l'ordinateur on dira qu'il est écrit dans un langage de **bas niveau.**

Langage de Programmation

Un langage de programmation est un ensemble de symboles et de règles destiné à décrire l'ensemble des actions qu'un ordinateur doit exécuter. C'est une manière de donner des instructions à un ordinateur.

Il faut noter que :

- Un langage de programmation permet de décrire à l'ordinateur, dans un langage qu'il comprend, la suite d'instructions et les données qu'il va manipuler,

- L'utilisation d'un langage de programmation se justifie par la difficulté d'exprimer la suite d'instructions dans le langage machine

La compilation et l'interprétation

Lien entre l'Homme et la Machine :

Question : Y a-t-il un moyen pour transformer un programme écrit sous la forme d'une suite d'instructions compréhensibles par l'ordinateur

Lien entre l'Homme et la Machine :

La traduction/transformation d'un programme peut se faire de deux manières : **l'interprétation et la compilation.**

Faciliter le Passage :

Du **Code Source vers un Fichier/Programme Exécutable**

La compilation et l'interprétation

Définition

Le **Code source est l'ensemble des instructions d'un programme** écrites dans un langage de programmation informatique de haut niveau.

Définition

Le **Fichier/Programme exécutable est un programme** directement compréhensible par le processeur de la machine, contenant une suite des "commandes" qui entraînent des actions ou des opérations de la part du système.

Problème

Le **Fichier/Programme exécutable Un programme écrit** dans un langage machine dépend étroitement du type de processeur utilisé : chaque type de processeur peut avoir son propre langage machine.
Changer de machine –> Réécrire entièrement le programme

La compilation et l'interprétation

Définition

Un programme écrit dans un langage **interprété a besoin d'un** programme auxiliaire appelé **interpréteur qui analyse, traduit et** exécute les instructions du programme.
Le cycle d'un interpréteur est le suivant :

1. lire et analyser une instruction,
2. si l'instruction est syntaxiquement correcte, l'exécuter,
3. passer à l'instruction suivante.

Définition

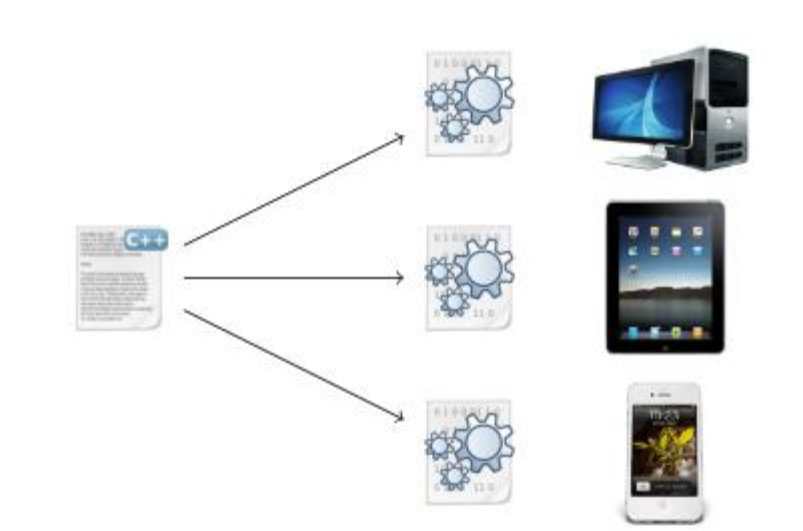
Le code source d'un programme écrit dans un langage de programmation **compilé est traduit une fois pour toutes par un** programme annexe appelé **le compilateur afin de générer un** fichier exécutable.

Le langage de programmation C est un langage compilé.

La compilation

Un compilateur est un programme informatique qui transforme:

- un **code source écrit dans un langage de programmation compréhensible par l'humain**
- en un **code assembleur ou code machine compréhensible seulement par la machine**



Avant de commencer

En linguistique, on distingue deux notions de base pour l'étude d'un langage naturel :

- **La syntaxe : qui est une branche de la linguistique qui** étudie la façon dont les mots(morphèmes) se combinent pour former des propositions et par la suite des énoncés. Elle s'intéresse à la forme.
- **La sémantique : qui s'occupe le plus souvent de la** signification de ces propositions/énoncés : étude des signifiés, concepts. Elle s'interesse donc au fond.

Nous gardons les deux concepts en langage de programmation : distinguer entre la façon d'écrire un programme dans un langage de programmation et le but poursuivit qui détermine une suite d'actions à faire faire.

En d'autres mots…

Il faut distinguer ente **la signification (la sémantique) des opérations/programme et la manière dont on l'écrit (la syntaxe,** l'ensemble de règles qui régissent l'agencement des groupes de mots).

Présentation du langage C: Un peu d'histoire

- Le langage C a été créé par **Dennis Ritchie, Ken Thompson(langage B) et Brian Kernighan(aide à le** populariser) au début des années 70 (Labo. Bell, New Jersey)

- Le but : éviter autant que possible l'utilisation de l'assembleur dans l'écriture du système UNIX : portabilité

- Avantage du C (par rapport à l'assembleur) : il permet aux programmes d'être plus concis, plus simples à écrire et facilement portables sur d'autres architectures

- Le C permet de programmer sur différentes plateformes comme Microsoft Windows, GNU/Linux ou Mac OS

Présentation du langage C: Quelques dates

- **1969 : UNIX par Ken Thomson (assembleur pour DEC** PDP-7(Programmed Data Processor) de marque DEC) sur mini-ordinateur
- **1972 : invention du C par Denis Ritchie**
- **1973 : UNIX en C par Denis Ritchie et Ken Thomson (PDP-11)**
- **1978 : Livre : The C Programming Language : C K&R (Kernighan** and Ritchie C)
- **1989 : 1ère normalisation par ANSI (American National** Standards Institute) : **ANSI C (C89)**
- **1990 : 2ème normalisation par ISO (International Standards** Organization) : ISO (C90)
- **1999 : 3ème normalisation par ISO (C99)**

Présentation du langage C: Généralités

- Le langage C a une **grammaire qu'il est nécessaire de respecter** pour que le compilateur soit capable de faire son travail,
- Le compilateur C n'est pas assez "intelligent" pour deviner ce qu'on a voulu dire et donc corriger les fautes,
- Il est cependant capable de dire s'il ne comprend pas, ou s'il y a des incohérences,
- Toutefois, il arrive souvent que le compilateur réussisse à traduire le programme, et que ce dernier ne fonctionne pas comme prévu : il s'agit alors d'**erreurs de conception, Bug !**

Faire attention . . . À ce qu'on demande au compilateur d'analyser !