

PART 1 : INTRODUCTION À LA PROGRAMMATION C

- ❑ **Qu'est ce qu'un programme?**
- ❑ **Le positionnement par rapport à l'ordinateur**
- ❑ **Les logiciels de la programmation**
 - ❑ Le compilateur
 - ❑ L'IDE : Environnement de développement
 - ❑ Installation
- ❑ **Base d'un programme**
 - ❑ Des données et des instructions
 - ❑ Des « librairies » de fonctions
- ❑ **Premier programme**
 - ❑ La fonction main(): entrée du programme
 - ❑ Afficher du texte avec la fonction printf()

Introduction à la programmation C

❖ Qu'est ce qu'un programme?

- C'est une suite finie d'opérations (instructions) organisées dans un ordre précis en vue de l'accomplissement d'une ou plusieurs tâches.
- Les instructions reposent sur des données qui sont des informations stockées en mémoire.

❖ Le positionnement par rapport à l'ordinateur

- L'ordinateur est construit sur plusieurs niveaux superposés et solidaires entre eux.
- Chaque niveau s'appuie sur le niveau précédent et en réduit la complication et permet d'élaborer des opérations plus complexes.

Introduction à la programmation C

Niveau	Correspondance
6	Programmes applicatifs
5	Langage de programmation
4	Langage assembleur
3	Noyau du système d'exploitation
2	Langage machine
1	Logique numérique

Introduction à la programmation C

Haut niveau

Langage humain :

"Si x est plus grand que 10,
alors décrémenter x..."

Langage haut niveau (C, PHP, ...):

"if(x > 10) x--;"

Langage assembleur :

```
"cmp x, 10          dec: dec x  
jb dec              "
```

Langage machine :

```
"0011101001110110110110101011101  
000011000110111011011001101001"
```

Bas niveau

Base d'un programme (1)

❖ Des données et des instructions:

Pour écrire un programme, nous disposons de deux catégories d'outils:

1. Les structures de données:

- Chaque type de variable est déterminé par un mot clé.
- Exemple: char, short, int, long, float, double, struct, tableaux et pointeur.

2. Traitement appliqués aux données ou instructions:

- Les instructions natives dans un langage sont distinguées par un mot clé.
 - Exemple: if, if-else, switch, while, do-while, for
- Il existe aussi différentes variétés d'opérateurs.
 - Affectation, arithmétiques, bits à bits, comparaisons, pointeurs, tableau, structures, fonctions

Base d'un programme (2)

❖ Des bibliothèques de fonctions

- Il est possible d'utiliser des instructions sophistiquées déjà écrites pour des opérations fréquentes.
- En C, nous disposons en standard de bibliothèques de fonctions classées par thèmes.
- Exemples: entrées-sorties, traitement de chaînes de caractères, fonctions mathématiques, etc

Premier programme (1)

- ❖ **En C, tous les programmes contiennent la fonction `main()`.**
- ❖ **Une fonction est toujours désignée par un nom et des parenthèses à droite.**
- ❖ **Les commentaires `//` et `/* ... */`**
- ❖ **Affichage du texte avec la fonction `printf()`:**
 - `Printf()` fait partie de la librairie `<stdio.h>`
 - Il faut inclure la librairie `<stdio.h>` à chaque fois qu'on souhaite utiliser cette fonction:
 - `#include<stdio.h>` tout au début du programme
 - Les « > » et « > » indiquent à la machine que le fichier « `stdio.h` » se trouve dans le dossier « `include` » du compilateur, sinon on aura un message d'erreur.

Premier programme (2)

- Printf() permet d'afficher **du texte** dans une fenêtre console.
- Exemple: Afficher « il fait beau »

■ `printf("il fait beau");`

Mettre des
guillemets

Un point
virgule à la fin

Premier programme (3)

❖ Programme qui dit « bonjour! »:

```
#include <stdio.h>    // Pour avoir l'accès à la fonction printf()

int main ()           // Entrée du programme
{                     // Ouverture bloc d'instructions

    /* appel de la fonction printf() qui affiche la chaîne de
    caractères passée en paramètre */
    printf("bonjour !");

    /* valeur de retour de la fonction main qui indique un
    bon déroulement du programme */
    return 0;

}                     // Fermeture bloc d'instructions
```