Support de cours langage C

Equipe Pédagogique Informatique – Département PCMI

M. ASSOHOUN Egomli Stanislas

INITIATION AU LANGAGE C

Buts du cours

Les buts de ce cours sont multiples.

- ❑ Apprendre les principes généraux de programmation.
- ❑ Apprendre les principes de l'algorithmique. C'est à dire la manière dont on organise et conçoit un programme.
- ❑ Apprendre la syntaxe d'un langage de programmation (langage C).

Besoin de Rigueur

L'apprentissage de l'informatique ne nécessite pas d'être extrêmement doué mais nécessite par contre de la rigueur. C'est pourquoi il faut : Écrire avec soin ces programmes : deux instructions à priori similaires (ou qui vous semblent identiques) ne donnent parfois pas les mêmes résultats. Oublier certains signes peut rendre un programme non fonctionnable.

INITIATION AU LANGAGE C

Besoin de travail sur machine

Une nécessité de travailler sur la machine pour comprendre les phénomènes qui entrent en jeu. Cela permet de confronter la théorie à la pratique. De plus, la programmation ne s'apprend concrètement qu'avec de la pratique

Environnement

L'environnement de travail, c'est à dire l'infrastructure qui vous est offerte, consiste en un compilateur et en environnement de développement.

Le compilateur : Compilateur GNU.

L'environnement de développement CODE BLOCK

CHI- GENERALITES

CHI- LANGAGE C

I- Historique

Le C a été conçu en 1972 par Dennis Richie et Ken Thompson, chercheurs aux Bell Labs, afin de développer un système d'exploitation UNIX sur un DEC PDP-11. En 1978, Brian Kernighan et Dennis Richie publient la définition classique du C dans le livre The C Programming language . Le C devenant de plus en plus populaire dans les années 80, plusieurs groupes mirent sur le marché des compilateurs comportant des extensions particulières. En 1983, l'ANSI (American National Standards Institute) décida de normaliser le langage ; ce travail s'acheva en 1989 par la définition de la norme ANSI C. Celle-ci fut reprise telle quelle par l'ISO (International Standards Organization) en 1990. C'est ce standard, ANSI C, qui est décrit dans le présent document.

CHI- LANGAGE C

II- Justification du choix du langage C

Largement répandu : Le langage C est un langage qui a été et reste largement utilisé par nombre de programmeurs même si actuellement il tend à être supplanté. Ainsi Windows (à partir de windows 95) a été écrit en C (les versions de maintenant en C++). En tout cas il s'avère comme nécessaire de le connaitre pour bon nombre d'informaticiens.

Langage de haut niveau: Ce langage est aussi un langage évolué qui permet de rédiger les programmes sous une forme lisible humainement.

Possibilité d'évolutions: Une fois le langage C connu et les principes d'algorithmique compris, cela donne les moyens, à chacun de pouvoir apprendre (avec un certain travail néanmoins) d'autres langages tels que : – Le C++ – Le JAVA

CHI- LANGAGE C

II- Justification du choix du langage C (suite)

Un bémol : Le langage C est ardu à apprendre et n'est pas forcément le plus pédagogique car :

– C'est un langage syntaxiquement complexe.

– C'est un langage dont le compilateur est très souple. Il peut donc laisser beaucoup d'erreurs.

Aussi le cadre de travail n'est pas aussi normé qu'avec des langages tels que PASCAL.

III- Définitions et terminologie

- **Ordinateur** : Le terme ordinateur désigne un appareil électronique qui accepte les données sous un format numérique et les traite en vue d'un résultat.
- **Programme:** C'est la description d'une suite d'opérations élémentaires en vue de la réalisation au cours de son exécution d'une action donnée.

CHI- LANGAGE C

III- Définitions et terminologie

❑ **Programme:** (suite)

Ces opérations peuvent être :- une suite d'opérations élémentaires différentes - ou la répétition un grand nombre de fois d'un ensemble d'opérations. On parle dans ce cas là d'algorithme.

Ces opérations manipulent des données structurées des : structures de données.

On voit donc apparaître les notions **d'exécution**, **d'instructions** et de **données**

❑ **Remarques:** Au cours de leurs exécutions les programmes manipulent – des donnés simples – des structures de données – des adresses mémoires sur ces données. Ces dernières doivent exister en mémoire.

❑ **Logiciel:** Un logiciel est un programme ou un ensemble de programmes qui visent à offrir à un utilisateur (ou à l'ordinateur) une fonctionnalité ou un service donné.

CHI- LANGAGE C

III- Définitions et terminologie

- **Structure de données :** Une structure de données peut être vue comme une manière d'organiser des données simples (nombre, caractères,...) dans un ensemble plus complexe en vue de faciliter le maniement de cet ensemble par le programme.
- **Les fichiers:** Sont une manière de stocker les données sur un périphérique mémoire. Il existe plusieurs type de fichiers. Citons, entre autre, .doc, .htm, .xls, .dat, .txt chaque extension "qualifiant" la manière dont le fichier est structuré .... Ces fichiers sont des fichiers de données.
- **NB**: Le .exe se distingue ce n'est pas un fichier de données mais un binaire ou **programme exécutable** qui décrit ce que doit faire la machine (le processeur) interpréter les ordres, manipuler les données,

CHI- LANGAGE C

III- Définitions et terminologie

- **Fichier exécutable:** Programme directement compréhensible par le processeur, contenant une suite des "commandes" qui entraînent des actions ou des opérations de la part du système. En tant qu'adjectif, il caractérise un fichier que l'on peut exécuter.

IV- Langage de Programmation

Nous abordons maintenant la manière dont un humain peut écrire un programme. . Le fait d'écrire des programmes s'appelle la Programmation.

Le principe est simple : il s'agit de dire à l'ordinateur ce que vous voulez qu'il fasse. Pour ce faire il faut décrire à l'ordinateur, lui dire dans un langage qu'il comprend, la suite d'instructions et les données qu'il va manipuler. Dans ce but, on utilise un langage de programmation..

CHI- LANGAGE C

IV- Langage de Programmation

Nous abordons maintenant la manière dont un humain peut écrire un programme. . Le fait d'écrire des programmes s'appelle la Programmation.

Le principe est simple : il s'agit de dire à l'ordinateur ce que vous voulez qu'il fasse. Pour ce faire il faut décrire à l'ordinateur, lui dire dans un langage qu'il comprend, la suite d'instructions et les données qu'il va manipuler. Dans ce but, on utilise un langage de programmation.

- **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 qui est, le plus souvent, lisible par un humain mais pas directement par l'ordinateur.

CHI- LANGAGE C

IV- Langage de Programmation

- **Langage Machine** : C'est le "langage" de base compréhensible par un ordinateur (utilisé par le processeur), soit une suite de zéros et de uns.

NB: le langage machine n'est pas compréhensible facilement par l'humain moyen. Aussi,
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.

- **Langage de haut niveau de bas niveau** : Plus le langage est compréhensible par l'homme, en d'autres termes plus il est évolué, plus on dira qu'il est de haut niveau. Ainsi si la programme est écrit dans un langage proche de l'ordinateur on dira qu'il est écrit dans un langage de bas niveau.

CHI- LANGAGE C

IV- Langage de Programmation

❑ La compilation

On s'intéresse maintenant à la transformation du programme écrit sous la forme d'une suite d'instructions compréhensibles par l'homme en une suite d'instructions compréhensibles par la Machine.

- ✓ **Code source:** C'est l'ensemble des instructions d'un programme écrites dans un langage de programmation informatique de haut niveau. La traduction du code source peut se faire de deux manière soit par **l'interprétation**, soit par la **compilation**.
- ✓ **l'interprétation** : A l'interpretation un programme auxiliaire analyse, traduit et exécute les instructions du programme.

CHI- LANGAGE C

IV- Langage de Programmation (suite)

- ✓ **La compilation**: Le code source d'un programme écrit dans un langage dit "compilé" va être traduit une fois pour toutes par un programme annexe (le compilateur) afin de générer un fichier exécutable.

- ❑ **NB:** La compilateur dépend du langage. Un compilateur C ne va pas pouvoir compiler un programme écrit en PASCAL. En effet, le compilateur s'appuie sur la syntaxe du langage pour lequel il a été écrit. Le langage C est un langage compilé.

- ❑ **Sémantique et syntaxe d'un langage**

La syntaxe est la branche de la linguistique qui étudie la façon dont les mots se combinent pour former des propositions, tandis que la sémantique s'attache le plus souvent à la signification des proposition.

CHI- LANGAGE C

IV- Langage de Programmation (suite)

❑ Sémantique et syntaxe d'un langage (suite)

La syntaxe est la branche de la linguistique qui étudie la façon dont les mots se combinent pour former des propositions, tandis que la sémantique s'attache le plus souvent à la signification des proposition. Ces deux concepts s'appliquent aussi en informatique.

- ✓ **Syntaxe**: La syntaxe est la définition de l'ensemble de règles qui régissent l'agencement des groupes de mots (entités lexicales).
- ✓ **Sémantique**: La sémantique est l'étude de la signification (le but poursuivi) des programmes.

Ainsi la syntaxe est spécifique à un langage de programmation et consiste à l'organisation d'une instruction tandis que la sémantique n'est pas spécifique à un langage (mais plus à un type de programmation) et consiste dans l'organisation des instructions du programme.

CHI- LANGAGE C

IV- Langage de Programmation (suite)

❑ Sémantique et syntaxe d'un langage (suite)

Il existe plusieurs manières de structurer la façon dont on organise la suite des instructions et leur traitement pour réaliser une même finalité. Ainsi on a différente manière de programmer appelées : – Programmation fonctionnelle – Programmation déclarative – Programmation impérative – Programmation objet. Celle que nous allons utiliser est la programmation impérative.

- ✓ **Programmation impérative**: Les instructions qui modifient les données sont exécutées simplement les unes après les autres.

L'état du programme est défini par le contenu de la mémoire centrale à un instant et ne peut être modifié que par une instruction, La plupart des langages impératifs offrent de plus des moyens de structurer les suites d'instructions.

CHI- LANGAGE C

IV- Langage de Programmation (suite)

❑ Sémantique et syntaxe d'un langage (suite)

- ✓ **Programmation structurée**: Un langage de programmation structuré offre des structures de contrôle standardisées qui permettent d'organiser la suite des instructions (répétitions, branchements conditionnels)

L'état du programme est défini par le contenu de la mémoire centrale à un instant et ne peut être modifié que par une instruction, La plupart des langages impératifs offrent de plus des moyens de structurer les suites d'instructions.

CHI- LANGAGE C

V- Organisation générale en écrit en C

Un programme écrit en C s'écrit toujours de la même manière. Sa structure générale est la suivante.

[Directives du préprocesseur]
main(){
[Déclaration des variables]
[instructions]
Return 0;
}
	Exemple
	
	#include <stdio.h>
	
	void main(){
	printf("Mon nom est : Jean-Jacques");
	Return 0;
	}

CHI- LANGAGE C

V- Organisation générale en écrit en C (suite)

❑ Description

- ✓ **Identificateurs**: Un identificateur est un "nom" composé d'une suite de caractères alphanumériques, c'est à dire les chiffres, les lettres, auxquels s'ajoutent le caractère « _ ». Un chiffre ne peut être la première lettre d'un identificateur. Un identificateur peut être tronqué à 32 caractères. Les majuscules et les minuscules sont différenciées (on dit que c'est Case sensitive). Un identificateur sert à désigner : Une variable – Une fonction – Un type.
- ✓ **Mots réservés du langage :** Ces mots sont appelés parfois mots-Clefs ou mots réservés Il s'agit entre autre des mots : – Les types (char, float, int, void, ...) – Des instructions (do, while, else, if, ...) – divers (static, return.)

❑ **Caractères réservés:** Caractères qui ne peuvent être utilisé comme identificateur. – =, +, *, &

CHI- LANGAGE C

V- Organisation générale en écrit en C (suite)

❑ Description

✓ **Instructions** : Une instructions est un ordre de traitement que l'on donne à l'ordinateur.

Syntaxe : instruction ;

✓ **Bloc d'instructions :** Il est aussi parfois important de grouper un ensemble d'instructions

Syntaxe :

```
{
instruction1 ;
instruction2 ;
instruction3 ;
}
```

CHI- LANGAGE C

V- Organisation générale en écrit en C (suite)

❑ Description

✓ **Variables** : Une variable est un emplacement de la mémoire dans lequel est stockée une valeur. Chaque variable porte une nom et c'est ce nom qui sert `a identifier l'emplacement de la mémoire représente par cette variable.

Syntaxe : type <var_1>, <var_2>, . . . , <var_n >;

Exemple: int variable1; int autrevariable1 , autrevariable2 ;

✓ **Les types** : Une variable est une donnée en mémoire caractérisée par deux choses : Son adresse, - Sa valeur. Cette dernière définit ce que cette variable représente comme type d'information et qualifie le type de valeur de la variable.

Cette qualification s'appelle le typage et la classe de valeur le type. Nous verrons au début trois types même s'il en existe beaucoup plus. Les types (char, int, float, ...)

CHI- LANGAGE C

V- Organisation générale en écrit en C (suite)

❑ Description

✓ **Constante** : Une constante est une valeur non modifiable qui apparaît littéralement dans le code source

Exemple: const A=1.0 définit une constante de type float.

❑ Instructions de base

✓ **Affectation:** Si on souhaite affecter à la variable v une valeur, on utilise l'operateur =.

Exemple: int v ; v = 5 ; ou int v = 5 ;

✓ **Saisie:** Pour récupérer la saisie d'un utilisateur et la placer dans une variable <variable>, on utilise l'instruction suivante :

Syntaxe : scanf ("%X" , &<variable >);

CHI- LANGAGE C

V- Organisation générale en écrit en C (suite)

❑ Instructions de base

✓ Saisie (suite)

x	Description
d ou i	Entier
f	Réel
a	Caractère
s	Chaine de caractère

Exemple: int nb1; scanf(”%d”,&nb1); ou int nb1; float nb2 float ; scanf(”%d%f”,&nb1,&nb2);

CHI- LANGAGE C

V- Organisation générale en écrit en C (suite)

❑ Instructions de base

✓ **Affichage** : Cette instruction permet d'afficher la valeur d'une variable.

Syntaxe : printf ("%X" , <variable >);

Exemple1 : int v =10; printf (" l a v a l e u r de l a v a r i a b l e v e s t %d" , v) ;	int a , b , c ; float c ; a = 1 ; b = 2 ; c = 3 printf ("La valeur de a est %d , celle de b est %d , et celle de c est %f . " , a , b , c) ;

CHI- LANGAGE C

V- Organisation générale en écrit en C (suite)

❑ Instructions de base

- ✓ **Opérateurs arithmétiques** : les opérateurs arithmétiques sont : (- ; + ; *; / ; % reste de la division entière)
- ✓ **Opérateurs de comparaisons**: les opérateurs de comparaison sont : (< ; > ; <=; >= ; ==; !=)
- ✓ **Opérateurs logiques booléens**: les opérateurs logiques sont : (&& le et ; || le ou, ; ! la négation)
- ✓ **Commentaires**

// ceci est un commentaire sur une ligne

/* ceci est un commentaire sur plusieurs lignes*/

CHI- LANGAGE C

V- Organisation générale en écrit en C (suite)

❑ Instructions de base

✓ Formes contractées

a =a +1 < - > a++	a =a -1 < - > a--
a = a + b < - > a += b	a = a – b < - > a -= b
a = a * b< - > a *= b	a = a & b < - > a &= b
a = a % b < - > a %= b	a = a * b< - > a *= b
a = a ^ b < - > a ^= b	a = a | b < - > a |= b

✓ **Le cast** : On caste en plaçant entre parenthèse le type dans lequel on veut convertir juste avant l'operande que l'on veut convertir.

Exemple: int i = 4 , j= 5 ;

printf ("Le quotient de %d et %d est %f \n" , i , j , (float) i/j) ;

CHI- LANGAGE C

VI- Traitements conditionnels

On appelle traitement conditionnel une portion de code dont l'exécution est conditionnée par le succès d'un test.

❑ Si ... Alors

Si *condition* **alors**
| instructions
fin si

```
if (<condition>)
{
        <instructions>
}
```

NB: La formulation d'une condition se fait souvent `a l'aide des operateurs de comparaison.

❑ Si ... Alors ... Sinon

Si *condition* **alors**
| instructions
sinon
| autresinstructions
fin si

```
if (<condition>)
{
        <instructions1>
}
else
{
        <instructions2>
}
```

CHI- LANGAGE C

VI- Traitements conditionnels

On appelle traitement conditionnel une portion de code dont l'exécution est conditionnée par le succès d'un test.

❑ Switch

```
switch(<nomvariable>)
{
   case <valeur_1> : <instructions_1> ; break ;
   case <valeur_2> : <instructions_2> ; break ;
   /* ... */
   case <valeur_n> : <instructions_n> ; break ;
   default : /* instructions */
}
```

NB: N'oubliez surtout pas les break !.

CHI- LANGAGE C

VII- Boucles

Une boucle permet exécuter plusieurs fois de suite une même séquence d'instructions. Cette ensemble d'instructions s'appelle le corps de la boucle. Chaque exécution du corps d'une boucle s'appelle une itération, ou plus informellement un passage dans la boucle. Lorsque l'on s'apprete a exécuter la première itération, on dit que l'on rentre dans la boucle, lorsque la dernière itération est terminée, on dit qu'on sort de la boucle. Il existe trois types de boucle : while – do ... While – for

Chacune de ces boucles a ses avantages et ses inconvénients. Nous les passerons en revue ultérieurement.

CHI- LANGAGE C

VII- Boucles

❑ while

En C, la boucle tant que se code de la fa¸con suivante :

```
while(<condition>)
{
  <instructions>
}
```

Les instructions du corps de la boucle sont délimitées par des accolades. La condition est évaluée avant chaque passage dans la boucle, a chaque fois qu'elle est vérifiée, on exécute les instructions de la boucle. Une fois que la condition n'est plus vérifiée, l'execution se poursuit après l'accolade fermante.

CHI- LANGAGE C

VII- Boucles

❑ do ... while

Voici la syntaxe de cette boucle :

```
do
{
   <instructions>
}
while(<condition>);
```

La fonctionnement est analogue a celui de la boucle tant que `a quelques détails près :
– la condition est évaluée après chaque passage dans la boucle.
– On exécute le corps de la boucle tant que la condition est vérifiée.
En C, la boucle répéter ... jusqu'`a est en fait une boucle répéter ... tant que, c'est-`a-dire une boucle tant que dans laquelle la condition est évaluée a la fin. Une boucle do ... while est donc exécutée donc au moins une fois.

CHI- LANGAGE C

VII- Boucles

❑ For

Cette boucle est quelque peu délicate. Commençons par donner sa syntaxe :

```
for(<initialisation> ; <condition> ; <pas>)
{

  <instructions>
}
```

L'<initialisation> est une instruction exécutée avant le premier passage dans la boucle. La <condition> est évaluée avant chaque passage dans la boucle, si elle n'est pas vérifiée, on ne passe pas dans la boucle et l'execution de la boucle pour est terminée. La <pas> est une instruction exécutée après chaque passage dans la boucle.

Merci de votre attention