UNIVERSITE PIERRE ET MARIE CURIE

Licence de Chimie UE-209

INTRODUCTION A LA PROGRAMMATION FORTRAN

Plan :

1 : Introduction

2 : Structure d'un programme Fortran

3 : Aiguillages

4 : Processus itératifs

5 : Variables indicées

6 : Sous-programmes

Contrôle des connaissances :

CC N°1 (*10 min - 5pts*) pendant le deuxième cours

CC N°2 (*10 min - 5pts*) pendant le troisième cours

Examen (*1h - 30pts*) pendant le quatrième cours

Objectifs

L'objectif de cette UE est d'approfondir les connaissances des étudiants dans les objets et concepts mathématiques de base utilisés dans les différents domaines de la chimie. L'enseignement est donc axé sur les besoins des différents enseignements dispensés en L2 et L3 de la mention chimie, dont seront tirées les principales applications. Les concepts sont illustrés et complétés par des enseignements d'informatique qui vont familiariser les étudiants avec des notions de Linux et de Fortran. Ces notions d'informatique permettront ensuite aux étudiants lors de leurs études (L3 et Master) d'aborder l'informatique appliquée aux sciences chimiques.

Conventions utilisées

Le texte écrit comme ceci est un message affiché sur l'écran ou un texte à taper au clavier

Le texte écrit *«comme ceci»* est un texte que vous devez adapter selon le contexte.

la touche ENTER (entrée) est indiquée par le signe ↵

le caractère espace est indiqué par le signe ^

- COURS I –

INTRODUCTION

I.1) Historique et Définitions

Informatique: traitement des **INFOR**mations par voie auto**MATIQUE:** ensemble des techniques de traitement de l'information utilisant les ordinateurs.
Ordinateur: calculateur digital doué de mémoire. Les calculateurs sont soit *analogiques* (traitement de l'information continue - règle à calcul), soit des *digitaux* (traitement de l'information discontinue - boulier; ordinateur) L'unité la plus simple est le chiffre binaire ou **BIT BI**nary-digi**T**.

Pascal invente la première machine à additionner en 1642, Leibniz, la première machine à multiplier en 1671, Babbage les cartes perforées en 1840. En 1855 International Business Machine est fondée aux Etats-Unis.
Matériel (Hardware, Hard): tout ce qui a une existence physique: l'ordinateur et ses périphériques.
Logiciel (Software, Soft): l'information, elle même et les langages permettant de la traiter.

L'algorithmique (recherche des algorithmes les mieux adaptés aux ressources de l'ordinateur) et la programmation ne sont qu'une branche de l'informatique qui s'intéresse surtout à la conception des machines, à leurs couplages, aux recherches de langage d'exploitation permettant aux machines de fonctionner...

I.2) Les unités d'information.

Le chiffre binaire est le BIT. Le codage d'un chiffre décimal nécessite quatre bits. Le système décimal n'est pas le plus favorable puisque 4 bits permettent tout aussi bien de coder un chiffre hexadécimal 2^4=16. Le codage de chiffres va utiliser des multiples de 4 bits. Les unités utilisées sont

- l'octet (BYTE) qui vaut 8 bits,
- le kilooctet, K qui vaut 2^{10} bytes soit 1024 bytes (proche de 1000)
- le megaoctet, MK qui vaut 2^{10} K. (1K de K)
- le gigaoctet, qui vaut 2^{10} MK.

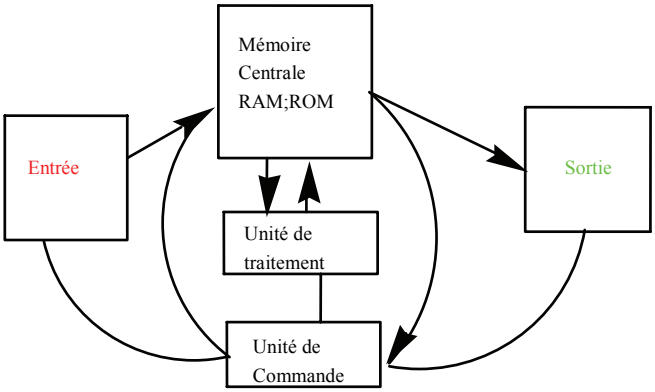
Un entier de quatre octets représenté par 32 bits (dont un pour le signe) devra être au maximum égal a 2^{31}-1= 2 147 483 647 (codé en binaire) soit un nombre de 10 chiffres décimaux;

Le codage d'un texte nécessite, la représentation des chiffres, des lettres de l'alphabet (y compris des caractères spéciaux + - * /). C'est le cas d'un texte fortran. 6 bits permettent un premier codage. Un chiffre (0 à 3) correspondant à deux bits (partie hors-texte) associé à un chiffre décimal (partie numérique) définissent un caractère. Ainsi en codage DCB, selon que la partie hors-texte vaut 0, 1, 2 ou 3, la valeur 1 de la partie numérique devient 1, A, J ou /. Le principe de l'EBDIC est équivalent.

S'il y a beaucoup de texte (traitement de texte), il vaut mieux s'organiser en multiple de 6 bits (mot BCD ou EBDIC); s'il y a beaucoup de chiffres, il faut prendre des multiples de 8 bits (octets). L'utilisation d'un plus grand nombre de caractères (Majuscule, minuscules, lettres grecques...) demande davantage de place. Un octet permet de définir 2^8=256 caractères. Le code ASCII définit 2^7=128 caractères que l'on peut étendre à 256 en utilisant des octets.

I.3) Le Matériel.

Il comprend des organes d'entrée, de traitement et de sortie.



Périphériques :
Entrée : **Clavier**, Unité de disque, lecteur CD-ROM ou DVD, clé USB, réseau de communication ...
Sortie : **Ecran**, Unité de disque ou graveur CD-ROM/DVD, clé USB, Réseau de communication, **Imprimante** ....

Unité Centrale :
Mémoire centrale: là où l'on stocke les informations. Une image simple serait celle d'une armoire avec des tiroirs: chaque tiroir (registre) contient un nombre en binaire.

ROM (Read Only Memory) Mémoire morte dans laquelle le constructeur a préenregistré des logiciels (le système permettant à la machine de fonctionner).

RAM (Random Access Memory) Mémoire vive dans laquelle vous copiez des logiciels (système d'exploitation que l'on peut changer: l'opération de chargement s'appelle Boost; vos programmes)

Organe de traitement :
Microprocesseur; c'est la partie active de l'ordinateur, celle où se font les opérations. Elle est toute petite (en bits). En effet prenons l'addition comme opération de base. Elle nécessite la présence de trois "tiroirs" si l'on veut conserver les valeurs de départ (une pour les deux nombres à ajouter, une pour le résultat) ou de deux tiroirs seulement (si l'on place le résultat, là où il y avait un des deux nombres à ajouter).

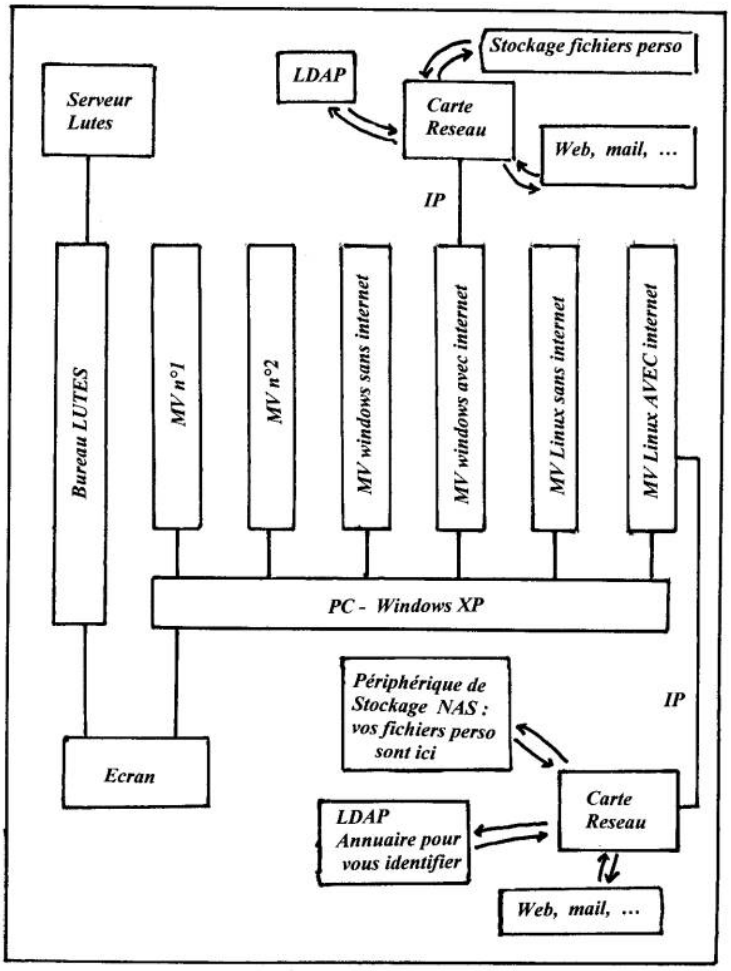
Unité de commande : organe physique permettant de charger les instructions dans un ordre donné (il y a un compteur ordinal) et de les contrôler. Horloge interne.

I.4) login, password : accès aux ordinateur.

Vous disposez d'un PC relié physiquement au réseau informatique d'enseignement de l'Université Pierre et Marie Curie.

- ***Le matériel informatique à LUTES***

Les machines de Lutes permettent de travailler dans plusieurs configuration en utilisant des machines virtuelles (*MV*).



Les salles d'informatique de Lutes sont équipées de PC qui permettent de travailler soit avec le système d'exploitation Windows, soit avec le système d'exploitation Linux (Mandriva).

Pour les enseignements de LC209, les étudiants se connecteront sur :
« *Mandriva 2006 avec authentification* »

<u>Pour se Connecter sur le PC</u> :
mettre son nom d'utilisateur, puis taper son mot de passe (ce dernier n'apparaît pas sur l'écran, on voit des *), puis taper sur la touche ENTER pour valider.

Lorsque l'ordinateur sera prêt, vous serez dans une configuration KDE de Linux (écran avec des icônes style Windows).

On va travailler dans une **fenêtre « terminal »** pour saisir des lignes de commandes. Le programme qui fournit cette interface est le shell (ici on aura un shell « bash ») . Dans la fenêtre, on a un prompt :

[login@localhost login]$

Ce prompt suivi par un curseur indique que l'ordinateur attend une commande.

Les répertoires sont organisés comme un arbre, la racine est le premier répertoire existant. Il est toujours indiqué par le signe /
Vous n'avez pas le droit d'écrire dedans (ou d'effacer).
Le nom d'un répertoire est un nom composé par exemple : /home/tartempion
Les différents membres du nom sont séparés par des slash (/). C'est une séquence hiérarchisée ; chaque nom est un sous-répertoire du nom qui le précède. Votre répertoire est situé dans /home/ tartempion, c'est-à-dire un répertoire situé dans / et qui s'appelle home et qui contient votre répertoire tartempion. Dans votre «Home Directory» (tartempion dans notre exemple), vous avez le droit de créer/effacer des fichiers et des répertoires.

Pour se Déconnecter du PC :

cliquer en bas à gauche sur K pour obtenir le menu :
Terminer la session
Eteindre l'ordinateur
...

sélectionner : « **Eteindre l'ordinateur** » pour revenir à l'écran de départ de LUTES.

I.5) Les logiciels.

Ensemble de programmes permettant d'utiliser l'équipement précédent.

- Système d'exploitation : **UNIX** il permet d'ordonnancer les taches exécutables par l'ordinateur et d'accéder aux programmes langages.
- Langages de programmation : Basic (**B**eginner's **A**ll **P**urpose **S**ymbolic **I**nstruction **C**ode)- 1965. **Fortran** première version 1954 Fortran IV 1962 Fortran V 1977 : il existe plusieurs versions avec en plus des possibilités d'extensions pas présentes dans les versions standards. Pascal (**P**rogram (**A**) **S**tructured **C**omputer **A**lgorithmic **L**anguage) – 1969. COBOL **Co**mmon **B**usiness **O**riented **L**anguage. ADA. C/C++, Java …

Ces langages doivent être traduits pour que la machine les exécute. En effet l'ordinateur ne comprend que les chiffres binaires; la traduction se fait au moyen de commandes UNIX: C'est la **compilation**. Le résultat obtenu en deux étapes est un module_objet puis un module_exécutable. La première étape "traduit" et crée le module_objet. La deuxième étape relie les modules_objets entre eux pour créer le module exécutable (load module).

Un programme est une suite ordonnée d'instructions. Quelque soit le langage, il comprend trois parties: une entrée (lecture de données), une partie traitement et une sortie (restitution des résultats).

I.5.a) Les commandes

Une action sur l'ordinateur est déclenchée par une commande. Une commande peut être obtenue en frappant son nom, puis en tapant sur la touche retour ↵ qui envoie l'exécution.

man

Des informations peuvent être consultées pour chaque commande par man cmd1 (man renvoie aux pages d'un manuel dans une bibliothèque de l'ordinateur).

Une commande peut être obtenue en frappant une touche préréglée du clavier; ce sont les clefs de commandes. Les clefs sont des touches qui exécutent directement une action sur l'ordinateur. Par exemple CNRTL-Z est défini et son action suspend une commande.

Le mode historique des commandes déjà tapées est souvent utile pour relancer une commande deux fois de suite. Vous activez cette option en lançant l'interpréteur de commande `bash` ↵. L'historique de vos commandes apparaîtra à l'aide des flèches hautes et basses.

I.5.b) Les commandes pour les répertoires.

pwd

La commande pwd affiche le nom absolu (à partir de /) du répertoire courant.

mkdir

créer un sous-répertoire par mkdir␣rep1

rmdir

détruire par rmdir␣rep1

ou bien par rm␣-r␣rep1 (catalogue ayant un contenu)

mv

changer son nom par mv␣rep1␣rep2

cp

copier par cp␣-r␣rep1␣rep2

Mise en commun de fichiers pour plusieurs répertoires :

I.5.c) Les commandes pour les fichiers.

ls

Taper ls ↵, vous verrez apparaître la liste des fichiers/répertoires qui sont déjà installés dans votre répertoire s'il y en a. Les fichiers contiennent des écritures. Ils peuvent contenir du texte à consulter dans un langage quelconque (fichier dans lequel on peut lire ou écrire), des listes de commandes, un module objet ou exécutable (dans ces 2 derniers cas le fichier n'est ni lisible ni imprimable. Ne jamais lancer l'impression d'un module exécutable!).

La plupart des commandes sont accompagnées de paramètres (obligatoires ou optionnels) et d'arguments. Par exemple ls␣-ail vous donne plus d'informations: un numéro (ou inode) associé à chaque fichier, des renseignements sur les gens autorisés à lire, écrire ou utiliser vos fichiers... Les paramètres a (all*) i (inode) et l (long) sont abrégés en ail et placés derrière un tiret. Les arguments concernent les objets sur lesquels agissent les commandes, par exemple un nom de fichier ou de répertoire.

more

La commande more vous montre le contenu d'un fichier. Par exemple more␣fic1 ↵
Si un fichier est trop long pour être affiché sur un seul écran, il est préférable de l'afficher page par page. Pour cela faire more␣-l␣*«nom du fichier»*. On passe alors de page en page en actionnant la barre d'espace (ou en arrière grâce à CNTRL et B).

grep

On peut rechercher une expression avec la commande grep. Par exemple, grep␣mot␣fic1 va rechercher l'expression mot dans le fichier fic1 et affichera à l'écran toutes les lignes contenant l'expression recherchée.

rm
mv
cp
lpr
wc
diff

détruire par rm^fic1 (sans confirmation)
rm^–i^fic1 (avec confirmation)
changer son nom par mv^fic1^fic2
copier par cp^fic1^fic2
imprimer un fichier lpr^-Pibm7^fic1
compter les lignes d'un fichier par wc^-l^fic1
les caractères et mots wc^-lcw^fic1
comparer 2 fichiers (et afficher la différence) diff^fic1^fic2 donne toutes les lignes qui diffèrent

Si une commande est incomplète, il peut y avoir affichage de ? au lieu du prompt. Cela vous indique que l'ordinateur attend un paramètre.

On peut enchaîner deux commandes (en les séparant par des points virgules s'il y a confusion possible entre nouvelle commande et argument de la précédente). La seconde prend en entrée le résultat de la première :

`ls^-l | more` la commande ls^-l fait défiler trop d'informations sur l'écran; more enchaîné va permettre de faire défiler ces informations page par page. le signe "pipe" (|) est obtenu en tapant: CNTRL+ALT+F6

Pour créer un fichier nouveau il suffit d'utiliser un éditeur en nommant un fichier nouveau. Ce fichier sera crée s'il n'existe pas, ouvert s'il existe déjà. Faire par exemple `ex^fic1, vi^fic1` ou `nedit^fic1`.

cat

Pour une première fois, nous allons faire différemment en utilisant la commande cat. Ce sera exceptionnel; par la suite vous utiliserez directement vi fic1. La commande cat affiche à l'écran un fichier : cat fic1 La flèche permet de réorienter la sortie: au lieu de l'écran
cat >fic1 crée le fichier fic1. Tout ce qui est entré au clavier est placé dans le fichier fic1. Pour sortir de l'édition du fichier, il faut faire CNTRL-D. (ou par CNTRL-Z si l'on a fait une fausse manoeuvre et si l'on veut annuler ce qui vient d'être édité). Le processus cat >fic1 remplace le fichier fic1 antérieur qui porte ce nom si un fichier existe déjà avec ce nom. Si l'on veut conserver le fic1 et écrire à sa suite, faire cat >>fic1.
La commande cat <fic1 >fic2 prend le contenu de fic1 et le place dans fic2 en remplaçant fic2. La commande cat <fic1 >>fic2 prend le contenu de fic1 et le place dans fic2 à la suite de son contenu.

I.5.d) Les commandes pour les processus.

ps
kill

Lorsqu'une commande est lancée, lorsqu'un travail est soumis, un processus apparaît.
ps donne les informations sur les processus en cours
kill provoque l'arrêt d'un processus spécifié kill -9 23896
Le numéro d'identification (dans l'exemple 23896) est dans PID de ps^u (travaux de l'utilisateur)

Un module exécutable est mis en œuvre par son nom de fichier (voir plus loin). Il est normalement exécuté en interactif (foreground) ce qui implique d'attendre que le processus soit terminé pour retrouver la main. Un processus interactif peut être suspendu par CNTRL-Z

Un processus peut être exécuté en arrière plan (background) ce qui vous laisse la possibilité de continuer à travailler. Cela se fait en accompagnant la commande du symbole &. Les commandes bg et fg font passer de foreground à background et réciproquement.

I.6) Compilation et Exécution.

ATTENTION : Les fichiers sources fortran doivent avoir un nom se terminant par .f

La source fortran doit être traduite par un compilateur pour créer un module objet. Le module objet est ensuite relié à d'autres modules objet par l'éditeur de liens pour engendrer le module exécutable. Il y a donc trois versions d'un même programme: les fichiers .f .o et le module exécutable. Seul le fichier .f est lisible.

1ere étape : compilateur fortran, g95, vérifie l'orthographe et la grammaire de votre programme

`g95 ^ -c ^` ***«nom du fichier fortran»***

La commande `g95 -c perim.f` vérifie l'orthographe et la grammaire de votre programme contenu dans perim.f. Si tout se passe bien, il apparaît dans votre répertoire un fichier .o (faire ls) et le message `Compilation successfull`.

2eme étape : le module objet est relié à d'autres modules objet par le compilateur pour engendrer le module exécutable

`g95 ^ -o ^` ***«nom du fichier exécutable»*** `^` ***«liste des modules objets»***

La commande `g95 ^ -o ^ perim ^ perim.o` crée le programme `perim`. En l'absence de nom le fichier exécutable va s'appeler `a.out` Si l'on doit réunir plusieurs modules objet faire
`g95^-o^exe1^fic1.o^fic2.o^fic3.o` si l'on doit réunir tous les modules objet faire
`g95^-o^exe1^*.o` attention, il ne faut pas avoir alors plusieurs sous-programmes avec le même nom. Il est recommandé de travailler dans un sous-répertoire dans lequel vous réunissez tous les fichiers dont vous avez besoin.

3eme étape : l'exécution du programme est lancée par `perim` ↵ (ou `./perim` ↵)

Exercices:

I-1) Créer et compiler le programme suivant :

```
^^^^^^PROGRAM^ESSAI
^^^^^^WRITE(*,*) 'bonjour^!'
^^^^^^END
```