Lorem ispum

I Modalités de Contrôle Continu

10% : 45 minutes de contrôle en TP (sur feuille).
40% : Projet (à un, à deux ou à trois).
50% : Examen final (programmation sur feuille).

Les polycopiés de cours seront sur son site web. On aura un cours de Programmation illustré en C, et non un cours de Langage C (le professeur considérant qu'on apprend le langage en programmant).
✕ **Attention** ✕ Savoir faire des diagrammes variables à avec une case et des flèches pour représenter les adresses mémoire et le contenu (car ça serait demandé dans les examens).

✕ **Attention** ✕ Il faut indenter correctement, et pas de saut de ligne dans les if/else : les accolades sont sur la même ligne que le if/else (sinon 0 pointé).

II Ordinateur

Un a un processeur (CPU) : celui qui est chargé d'exécuter les instructions. On est environ au 10 milliards d'instructions par seconde (10 GHz).
Mémoire (RAM) : connectée au processeur, elle stocke les données et les instructions en cours d'utilisation. (de l'ordre de quelques dizaines de Go).

Définition : L'**architecture de Von Neumann** est un modèle d'ordinateur dans lequel les données et les instructions sont stockées dans la même mémoire (RAM).

Remarque : Donc les instructions sont des données comme les autres : on peut écrire un programme qui écrit un autre programme : c'est la base de la compilation.

III Langages de programmation

A Le langage machine

Les premiers ordinateurs ont servis à faire des calculs numériques pour la création de la bombe atomique. On écrivait les programmes en langage machine (langage binaire).

Définition : L'**assembleur** est un langage isomorphe au langage machine, mais plus lisible pour les humains.

Exemple : Implémenter le code z = x + y en assembleur :

```
# x, y, z sont des adresses mémoire
ld x v1 # load : charge la valeur à l'adresse x dans le registre v1
ld y v2 # load : charge la valeur à l'adresse y dans le registre v2
add v1  v2  v3 # add : additionne les valeurs des registres v1 et v2, stocke le résultat dans v3
sw v3 z # store : stocke la valeur du registre v3 à l'adresse z
```

✕ **Attention** ✕ Pour les formules mathématiques, c'est parfois compliqué... Donc on a inventé des langages de plus haut niveau pour convertir les formules en assembleur.

B FORTRAN

Définition : Le **FORTRAN** (FORmula TRANslation) est un compilateur, premier langage de programmation.

On écrit un programme dans hello.f et FORTRAN le compile en .out.

C Interpréteur LISP

Définition : Le **LISP** (LISt Processing) est un langage de programmation interprété, inventé pour la manipulation de listes.

Fonctionnement d'un interpréteur : Il y a un programme (l'interpréteur) qui lit le code source ligne par ligne, le traduit en langage machine et l'exécute immédiatement. Il n'y a pas de traduction, et pas de fichier exécutable.
Remarque : C'est plus facile à implémenter, et plus simple pour débugger (car le code écrit est réellement le code exécuté).

Attention Par contre, c'est beaucoup plus lent : c'est pour ça qu'on n'utilise plus d'interpréteurs.

D Interpréteur Bytecode

Principe : on compile le code source en un code intermédiaire (bytecode), qui est ensuite interprété par une machine virtuelle (VM). C'est un compromis entre compilation et interprétation. hello.p –> hello.bytecode –> VM (interpréteur) –> exécution

Exemple : Java, Python

Attention Java est environ 4x plus lent qu'un programme compilé en langage machine. Python est environ 100x plus lent. Car les développeurs de Python "ne savent pas créer un bon langage intermédiaire".

IV Langage C : 1978

Problème : on a des ordinateurs partout ! (portable, montre, badges, carte d'identités, ...)
On a besoin de sécurité. On a inventé : MULTICS, mais ça a fait faillite. Deux gars, Thompson et Ritchie, d'anciens employés, ont créé UNICS : un système d'exploitation (B).
Leur direction leur a dit "cool", et UNICS est devenu "UNIX". Pour écrire UNIX, ils ont inventé le langage pour succéder à B : le langage C (pas très original comme nom).

Définition : Le **langage C** est un langage de programmation compilé, bas niveau, qui a ces caractéristiques :

- Petit (car basé sur un vieux PDP-7, moins puissant qu'une carte SIM).
- Suffisant pour un OS.
- Bas niveau (proche du langage machine).

Le C est particulièrement utilisé dans les systèmes embarqués (voitures, badges, etc.).
On va travailler en C99 (norme de 1999).

Attention Désavantage : de bas niveau (plus de travail), pas de sécurité (unsafe : on peut écrire des programmes qui plantent le système).

Remarque : On a donc un objectif : programmer sans se couper des doigts.

Tout ce qui suit est rédigé dans le polycopié. (Détails techniques)

Définition : La magie est consiste à faire ce qu'on s'est répété de génération en génération sans comprendre comment faire, et qu'il faut reproduire à l'identique pour que ça fonctionne.

Exemple : Un peu d'exemple de magie pour l'instant :

```c
#include<stdio.h> // magie
int main() { // magie
    // le code, pas magie
    return 0; // magie
}
```

Compilation :

Commande : gcc -Wall hello.c

gcc : GNU C Compiler (le compilateur)
-Wall : active tous les avertissements (warnings)
hello.c : le fichier source

Tant qu'on a des warnings, on corrige le code.
Ensuite, on exécute le programme compilé : ./a.out

Remarque : Fun fact : le premier programme écrit dans un nouveau langage est toujours "Hello, World!". Cela vient d'un livre de Brian Kernighan (un des créateurs du C) en 1978 qui a commencé son livre par "Hello, World!".

variable = adresse mémoire nommée + type (= taille + interprétation) Le programmeur voit : le nom, le type et la valeur. Le nom et le type sont des caractéristiques statiques (connues par le compilateur mais pas lors de l'exécution). La valeur est dynamique (connue lors de l'exécution).