Éléments d'informatique – Cours 1. Éléments d'architecture des ordinateurs

Pierre Fouilhoux et Lucas Létocart

Survol du contenu du cours (ce semestre)

Architecture des ordinateurs

Variables en programmation

Algorithme

Exemple et fin

ENT et contrôle des connaissances

Informations et matériel disponibles sur l'ENT

- Supports de cours
- Sujets de TD et de TP
- Exercices en + : à faire à la maison

Contrôle des connaissances :

- 2 partiels
- Évaluation continue :
  - QCM
  - TP notés
  - Contrôles TD

Survol du contenu du cours (ce semestre)

- Éléments d'architecture des ordinateurs
- Éléments de systèmes d'exploitation
- Programmation structurée impérative (éléments de langage C)
  - Structure d'un programme C
  - Variables : déclaration (et initialisation), affectation
  - Évaluation d'expressions
  - Instructions de contrôle : if, for, while
  - Types de données : entiers, caractères, réels, tableaux, types composés (enregistrements)
  - Fonctions d'entrées/sorties (scanf/printf)
  - Écriture et appel de fonctions
  - Débogage
- Notions de compilation
  - Analyse lexicale, analyse syntaxique, analyse sémantique
  - préprocesseur du compilateur C (include, define)
  - Édition de lien
- Algorithmes élémentaires (+ initiation à la complexité)
- Méthodologie de résolution, manipulation sous linux

« L'informatique n'est pas plus la science des ordinateurs que l'astronomie n'est celle des télescopes. »

E. W. Dijkstra

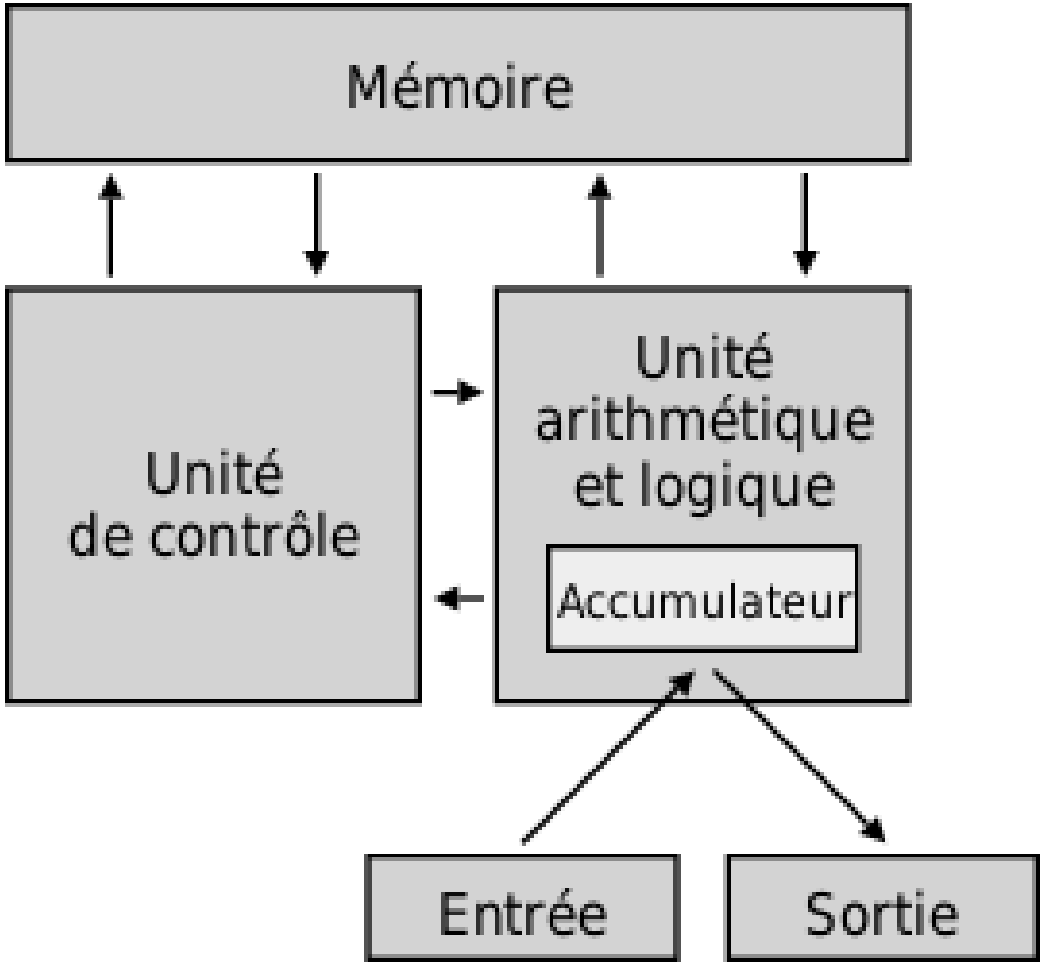
Architecture de von Neumann

(1945)

- John William Mauchly et John Eckert autant (ou plus) que vN
- Qu'est-ce que c'est ?
  - L'idée d'une machine à programme stocké
  - Une machine réalisée, l'ancêtre de nos processeurs
- De quoi cette machine est-elle faite ?
  - De mémoire (une suite de cases numérotées)
  - d'une unité de calcul, travaillant sur des registres
  - d'un bus système (adresses et données) reliant mémoire et UC
  - ~~De périphériques~~ (on oublie !)
  - La mémoire contient le programme et les données.

Architecture de von Neumann

(1945)



Représentation en binaire des informations

Definition (bit)

- Le chiffre binaire (en base 2), ou *bit*, est l'équivalent binaire de nos chiffres décimaux (en base 10). Il peut valoir soit 0 soit 1. Un bit est une quantité élémentaire d'information (oui ou non, ouvert ou fermé, etc.).

- Dans une base donnée, un nombre entier positif est représenté de manière unique par une suite de chiffres de la base :
  - En base 10, on écrit le nombre 109 comme la suite des chiffres 1, 0, 9 car :

$$109 = 1 \times 10^2 + 0 \times 10^1 + 9 \times 10^0$$

  - Et en base 2, le nombre 25 s'écrit comme la suite des bits 1, 1, 0, 0, 1 car :

$$\underline{11001} = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

Représentation en binaire des informations

- Plus généralement, la correspondance entre la représentation et le nombre est donnée par :

$$\underline{b_k \dots b_0} = \sum_{i=0}^{k} b_i \times 2^i$$

- L'information manipulée par un ordinateur est constituée de bits.
- Les cases mémoires et les registres contiennent des mots mémoire : des suite de n bits, où n est fixé une fois pour toute par l'architecture matérielle.
  Pour $n = 8$ (1 octet), la représentation de 5 est `00000101`
- les instructions du langage machine sont écrites en binaire.
- le langage assembleur est une notation du langage machine plus pratique pour les humains.

Nous en verrons un peu plus sur les codages en binaire des données dans un autre cours.

Cycle d'exécution

- Le registre compteur de programme (CP) contient l'adresse du mot mémoire représentant la prochaine instruction
- le contenu de ce mot est transféré de la mémoire centrale dans le registre d'instruction (RI)
- CP est *incrémenté* (c'est à dire que sa valeur augmente de 1)
- le contenu de RI est décodé afin de déterminer l'opération à exécuter
- l'opération est exécutée (le contenu d'un ou plusieurs registres est modifié, ou bien celui d'une case mémoire)
- Fin du cycle d'exécution et démarrage d'un nouveau cycle

Instructions

Une instruction type comporte un code d'opération et, si nécessaire, une ou deux *opérandes* (ou *arguments* de l'opération).

Vocabulaire

Dans l'expression arithmétique usuelle $3 + 5$, le signe $+$ est l'opérateur et les nombres 3 et 5 sont les opérandes.

Variables

Variable : un symbole (habituellement un nom) qui renvoie à une position de mémoire dont le contenu peut prendre successivement différentes valeurs pendant l'exécution d'un programme
NB :

- Une variable doit être déclarée et initialisée pour pouvoir être utilisée dans un programme
- Nom des variables : suite de
  - lettres majuscules et minuscules (non accentuées),
  - de chiffres
  - et du caractère _ (*underscore*/souligné),

  ne commençant pas par un chiffre.

Variables

(exemple d'utilisation)

Programme C

```
1  int main()
2  {
3  /* Declaration de la
4     variable x */
5      int x = 5;
6
7   /* Instructions */
8      x = x * 3;
9      x = 2 + x;
10 }
```

Trace

Instructions	Ligne	Cycles	x
Initialisation	5	0	5
`x = x * 3;`	8	1	15
`x = 2 + x;`	9	2	17

Résolution d'un problème informatique

Pour résoudre un problème sous une forme informatique, il faut suivre une méthodologie précise :

1. Définir les données en entrée et le résultat en sortie (*spécifications*)
2. Résoudre le problème *indépendamment du langage de programmation* (*algorithme*)
3. Traduire l'algorithme dans le langage de programmation (*programme*)
4. Effectuer la trace du programme (*vérification*)
5. Calculer le nombre d'instructions et l'occupation mémoire du programme (*complexité en temps et en espace*)

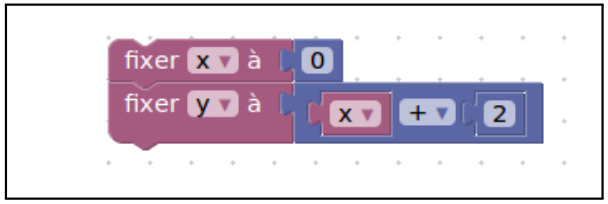
Exemple et fin

Problème : une variable doit recevoir la valeur d'une autre variable (initialisée à 0) incrémentée de deux

Algorithme

```
Nom: incrementationDeux
Entree: vide
Sortie: vide
Algorithme:
  Soit 2 variables entieres x et y
  x recoit la valeur 0
  y recoit la somme de la valeur de x et de 2
  Retour: OK
```

Programme Blockly



Programme C

```c
int main()
{
/* Declaration et initialisation des variables */
     int x;
     int y;

 /* Instructions */
     x = 0;
     y = x + 2;
}
```