Institut National des sciences Appliquées et des Technologies

Algorithmique et Structures de Données 1
Niveau MPI

Année universitaire
2019-2020

Dr. Aymen SELLAOUTI
Dr. Majdi JRIBI

Plan du cours

1. Concepts de base des algorithmes
2. Les structures conditionnelles
3. Les structures itératives
4. Les procédures et les fonctions
5. Les tableaux
6. Les chaînes de caractères
7. Les enregistrements
8. Les fichiers

Chapitre 1
Concepts de base des algorithmes

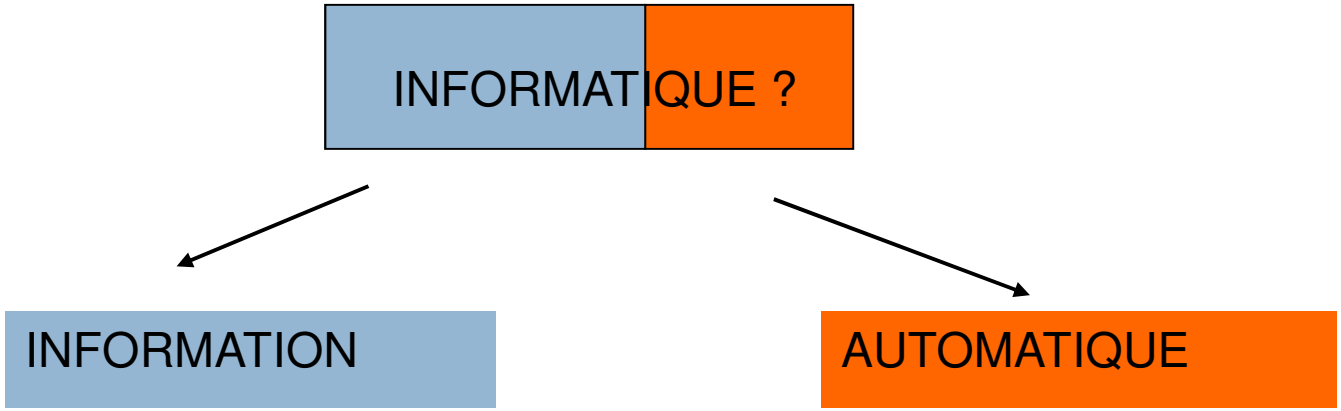
Plan

- Partie 1: Définitions et structure d'un algorithme
- Partie 2: Notion de variable et type
- Partie 3: Affectation, lecture et affichage
- Partie 4: De l'algorithmique au langage C

Plan

- Partie 1: Définitions et structure d'un algorithme
- Partie 2: Notion de variable et type
- Partie 3: Affectation, lecture et affichage
- Partie 4: De l'algorithmique au langage C

Informatique



L'informatique (Information + Automatique) est la science du traitement automatique de l'information par une machine capable de traiter ou de manipuler les informations ou les données.

Ordinateur

Machine qui saisit (périphériques d'entrée), stocke (mémoire), traite (programmes) et restitue (périphériques de sortie) des informations.

Algorithme

Définition 1

Un algorithme est une procédure de calcul bien définie qui prend en entrée une valeur, ou un ensemble de valeurs, et qui donne en sortie une valeur, ou un ensemble de valeurs. Un algorithme est donc une séquence d'étapes de calcul qui transforme l'entrée en sortie.

Définition 2

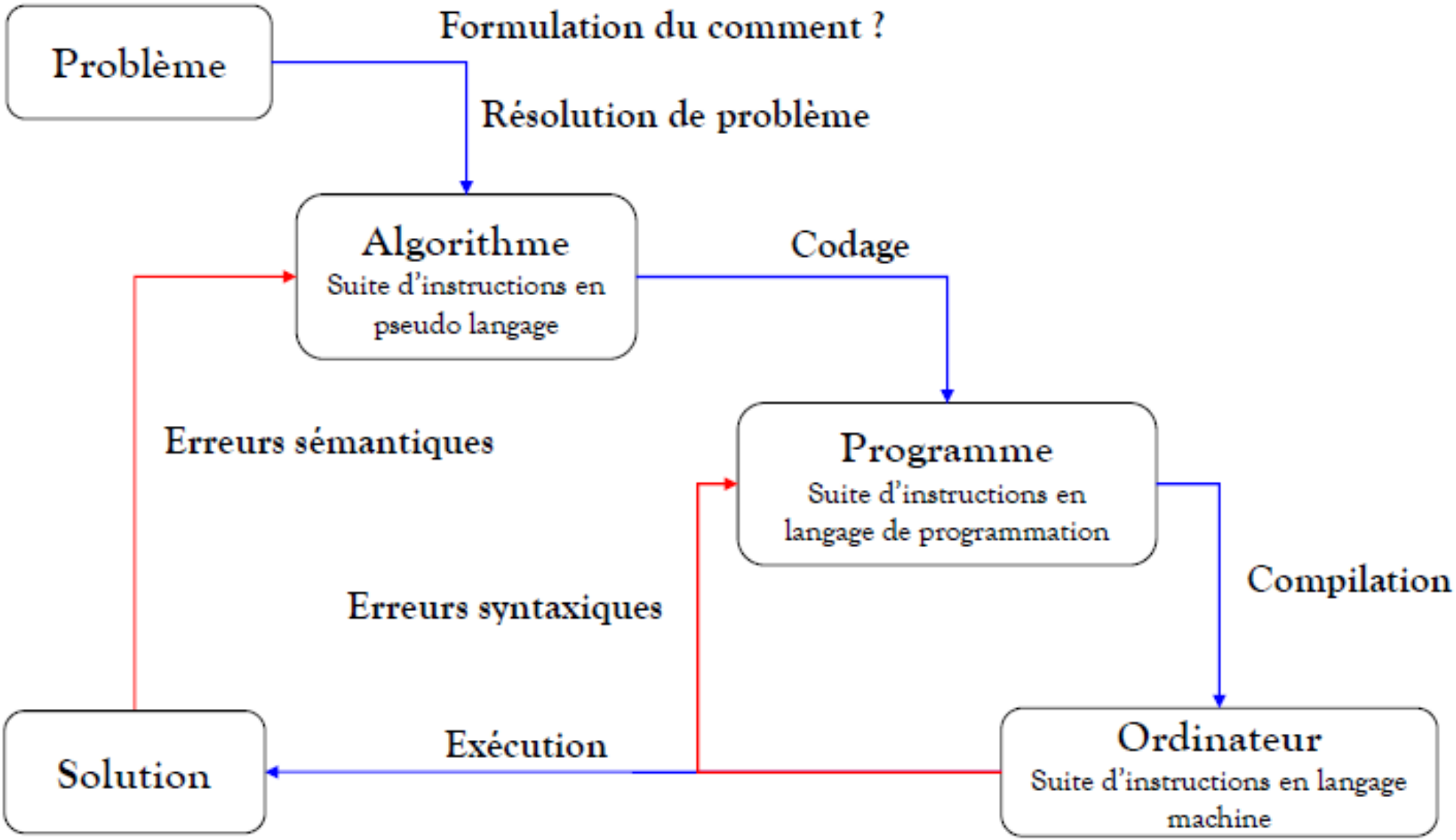
Un algorithme est une suite d'instructions ordonnées, qui une fois exécutée correctement, conduit a un résultat donné.

Algorithme

Caractéristiques:

- ✓ Clair : Pas d'ambiguïtés, compréhensible
- ✓ Déterministe : Avec un ensemble de données donné, il faut qu'il fournisse le même résultat quelle que soit la machine.
- ✓ Fini : il doit se terminer quelle que soit la machine, le temps et la date d'exécution.
- ✓ Efficace : l'algorithme doit effectuer le travail demandé avec le minimum de ressources.

Algorithme et programme



Structure d'un algorithme

Entête	**Algorithme** Nom_algorithme
Déclarations	**Const** : Liste des constantes, **Var** : Liste des variables **Struct** : Liste des structures
Corps	**début** Action 1; Action 2; { Commentaires } … Action n; **fin**

Plan

- Partie 1: Définitions et structure d'un algorithme
- Partie 2: Notion de variable et type
- Partie 3: Affectation, lecture et affichage
- Partie 4: De l'algorithmique au langage C

La notion de variable

- Dans un programme informatique, on a, en permanence, besoin de stocker provisoirement des valeurs :
  - données;
  - résultats obtenus par le programme

- Pour stocker une valeur au cours d'un programme, on utilise une **variable.**

- Dans la mémoire de l'ordinateur, physiquement, une variable correspond à un emplacement de mémoire, repéré par une adresse

La notion de variable

- Exemple : verre d'eau

Contenant

Contenu

Le contenu change
Mais le contenant
Reste le même

- Correspondance
  - Verre → Variable
    - Verre : objet contenant un liquide
    - Variable : zone mémoire contenant une valeur
  - Eau → Valeur

Déclaration d'une variable

- Une variable est une entité qui contient une information, elle est caractérisée par:
  - un nom, on parle d'identifiant (unique)
  - une valeur : information associée à une variable à un instant donné
  - un type, qui caractérise l'ensemble des valeurs que peut prendre la variable

- Déclaration: Identifiant de la variable : son type

- Exemple: Nom : chaîne de caractères
  Age : entier
  Distance : réel

Déclaration d'une variable

- Le type d'une variable caractérise
  - L'espace des valeurs que peut prendre une variable donnée
  - L'ensemble des actions que l'on peut effectuer sur une variable
  - Apparaît dans l'entête de l'algorithme avec la déclaration des variables

Type de données

- Le type d'une variable est inchangeable. Il est déclaré une seule fois et reste le même.

- Le contenu de la variable doit être du même type. On affecte pas un réel à une variable de type entier

Deux grandes catégories de types:

- Simples: booléen, entier, réel, …

- Complexes: des structures complexes (que nous verrons dans la suite du cours)

Les types simples

Les types de variables les plus courants en algorithmique

- Type numérique
  - Entier : ensemble des **entiers relatifs Z**
  - Réel : ensemble des **nombres réels R**
- Type alphanumérique
  - Caractère : toujours noté entre apostrophes
  - Chaîne de caractères : toujours notée entre guillemets
- Type booléen
  - Booléen : stocke uniquement les valeurs logiques VRAI et FAUX

Expressions et opérateurs

- Une expression
  - est une combinaison d'opérateur(s) et d'opérande(s)
  - est évaluée durant l'exécution de l'algorithme
  - possède une valeur (son interprétation) et un type

- Un opérateur est un symbole d'opération qui permet d'agir sur des variables pour produire un résultat.
- Une opérande est une entité (variable, constante ou expression) utilisée par un opérateur.

Exemple :

- a+b est appelée une expression
- a et b sont les opérandes
- + est l'opérateur

Expressions et opérateurs

- Un opérateur peut être Unaire ou Binaire :
  - **Unaire** s'il n'admet qu'une seule opérande, par exemple l'opérateur non
  - **Binaire** s'il admet deux opérandes, par exemple l'opérateur +

- Un opérateur est associé à un type de donnée et ne peut être utilisé qu'avec des variables, des constantes ou des expressions de ce type.

  - **On ne peut pas additionner un entier et un caractère**

Expressions et opérateurs

- **Opérateurs Booléens :** Non, Et, Ou, Ou Exclusif

- Non

a	non a
Vrai	Faux
Faux	Vrai

- Ou

a	b	a ou b
Vrai	Vrai	Vrai
Vrai	Faux	Vrai
Faux	Vrai	Vrai
Faux	Faux	Faux

- Et

a	b	a et b
Vrai	Vrai	Vrai
Vrai	Faux	Faux
Faux	Vrai	Faux
Faux	Faux	Faux

- Ou exclusif

a	b	a ouExclusif b
Vrai	Vrai	Faux
Vrai	Faux	Vrai
Faux	Vrai	Vrai
Faux	Faux	Faux

Expressions et opérateurs

Opérateurs sur les numériques

- On retrouve tout naturellement : **+, -, *, /, ^**
- Pour les entiers : **div** et **mod**, permettent respectivement de calculer une division entière et le reste de cette division
- L'opérateur d'**égalité** permet de savoir si les deux opérandes sont égales. Le résultat d'une expression contenant cet opérateur est un booléen.
- On a aussi l'opérateur **d'inégalité : ≠**
- Et pour les types possédant un ordre les opérateurs de comparaison **<, ≤, >, ≥**

Expressions et opérateurs

Opérateurs alphanumérique

- & : la concaténation
  - Cet opérateur permet de **concaténer** deux **chaînes de caractères**

Plan

- Partie 1: Définitions et structure d'un algorithme
- Partie 2: Notion de variable et type
- Partie 3: Affectation, lecture et affichage
- Partie 4: De l'algorithmique au langage C

L'instruction d'affectation

- Affecter une variable c'est lui attribuer une valeur
- L'instruction d'affectation se note avec le signe ←
- On peut affecter à une variable la valeur d'une autre variable.
- On peut affecter à une variable le résultat d'une opération en fonction d'autres variables.

Important : Une instruction d'affectation ne modifie que ce qui est situé à gauche de l'affectation ←

Exemple :

a ← 24,5

b ← a

a ← b + 5,5

L’instruction d’affectation

Une instruction d’affection doit respecter trois conditions :

- ✓ à gauche de l'affectation, on doit trouver **un nom de variable**, et uniquement cela. Dans le cas contraire, **il s'agit certainement d'une erreur !**

- ✓ à droite de l'affectation, on doit trouver une **expression ;**

- ✓ l'expression (située a droite de l'affectation) doit être **du même type** que la variable (située à gauche de l'affectation).

La lecture et l'écriture

L'instruction de lecture

- Une **instruction de lecture** permet à l'utilisateur de **rentrer des valeurs** au **clavier** pour qu'elles soient utilisées par le **programme**

Lire (A)

Dès que le programme rencontre une instruction Lire, l'exécution s'interrompt et attend la frappe d'une valeur au clavier.

La lecture et l'écriture

L'instruction d'écriture

- Une **instruction d'écriture** permet au programme de **communiquer des valeurs** à l'utilisateur en les **affichant** à l'**écran.**

Ecrire ("La valeur de B est :" , B)

Exemple

Exemple: Calcul de la surface d'un rectangle

Algorithme CalculSurface1

Const: Longueur= 4.32 , Largeur=3.77: réel

Var: Surface : réel

Début

Surface ← Longueur * Largeur

Ecrire ("La surface du rectangle est :", Surface)

Fin

Exemple

Exemple: Calcul de la surface d'un rectangle

Algorithme CalculSurface_2

Var: Longueur, Largeur, Surface : réel

Début

Ecrire("Donnez la longueur en m")

Lire(Longueur)

Ecrire("Donnez la largeur en m")

Lire(Largeur)

Surface ← Longueur * Largeur

Ecrire ("La surface du rectangle est :", Surface)

Fin

Plan

- Partie 1: Définitions et structure d'un algorithme
- Partie 2: Notion de variable et type
- Partie 3: Affectation, lecture et affichage
- Partie 4: De l'algorithmique au langage C

Eléments de base

Le langage C permet de découper un programme en modules.

La fonction principale est la fonction main qui va être exécutée la première.

```
#include <stdio.h>  /* instructions d'inclusion de bibliothèques par exemple ici la
    biliothèque  des entrées-sorties*/

main( )
{

    /* partie déclarative */

    /* partie instructions (actions) */
}
```

- **Remarque: Chaque instruction doit se terminer par ;**

Déclaration de variable

syntaxe : <type> <liste de variables>

- Le C propose les types simples suivants :
  - int, long, short, float, double, char
- On peut donc suivre les règles de traduction suivantes :

Algo. ⇒ C

Algorithmique	C
Entier,Naturel	int, long, short
Réel	float, double
Caractère	char
Booléen	int (1=VRAI, 0=FAUX)
Chaîne de caractères	*Voir les tableaux et pointeurs*

Traduire les opérateurs

Algo. ⇒ C

On traduit les opérateurs en respectant les règles suivantes :

Algorithmique	C
$=, \neq$	==, !=
$<, \leq, >, \geq$	<, <=, >, >=
et, ou, non	&&, ||, !
+, -, *, /	+, -, *, /
div, mod	/, %

Attention

En C l'affectation est une opération (opérateur =)

Traduire les opérateurs

Opérateurs ++ et −−

Les opérateurs unaires ++ et −− sont des opérateurs particuliers qui peuvent avoir jusqu'à deux effets de bord :

- En dehors de toute affectation, elle incrémente l'opérande associée, par exemple
  - `i++` et `++i` sont équivalents à `i=i+1`
- Lorsqu'ils sont utilisés dans une affectation, tout dépend de la position de l'opérateur par rapport à l'opérande, par exemple :
  - `j=i++` est équivalent à `j=i ;i=i+1 ;`
  - `j=++i` est équivalent à `i=i+1 ;j=i ;`

Traduire les opérateurs

Instructions simples

Elles finissent toujours par un ' ;'

Exemple

```
a=a+1;
```

Instructions composées

Les instructions composées qui permettent de considérer une succession d'instructions comme étant une seule instruction.
Elles commencent par "{" et finissent par "}"

Exemple

```
{a=a+1;b=b+2;}
```

Traduire l'instruction lire

scanf

- L'instruction `scanf` (du module `stdio.h`) permet à l'utilisateur de saisir des informations au clavier
- Syntaxe :

  scanf("chaîne de formatage",pointeur var1, ...)

  - La chaîne de formatage spécifie le type des données attendues, par exemple :
    - %d pour les entiers (int, short, long)
    - %f pour les réels (float, double)
    - %s pour les chaînes de caractères
    - %c pour les caractères

Par exemple

```
int i;
float x;
scanf("%d%f",&i,&x);
```

Traduire l'instruction Ecrire

printf

- L'instruction `printf` (du module `stdio.h`) permet d'afficher des informations à l'écran
- Syntaxe :

  printf ("chaîne de caractères" [, variables])

  - Si des variables suivent la chaîne de caractères, cette dernière doit spécifier comment présenter ces variables :
    - %d pour les entiers (int, short, long)
    - %f pour les réels (float, double)
    - %s pour les chaînes de caractères
    - %c pour les caractères
  - La chaîne de caractères peut contenir des caractères spéciaux :
    - \n pour le retour chariot
    - \t pour les tabulations

Traduire Ecrire

- Par exemple :

```
int  i=1;
float  x=2.0;
 printf ("Bonjour\n");
 printf ("i  = %d\n",i);
 printf ("i  = %d, x = %f\n",i,x);
```

- . . . affiche :

```
Bonjour
i = 1
i = 1, x=2.0
```