Faculté Polydisciplinaire de Taza

FILIÈRE : SCIENCE DE LA MATIÈRE PHYSIQUE

SEMESTRE :3

MODULE: ANALYSE NUMÉRIQUE ET ALGORITHMIQUE

COURS: ALGORITHMIQUE

ANNÉE UNIVERSITAIRE : 2021/2022

PROFESSEUR: HICHAM ZAARAOUI

Chapitre 1

Introduction à l'algorithmique

Plan

1) Introduction

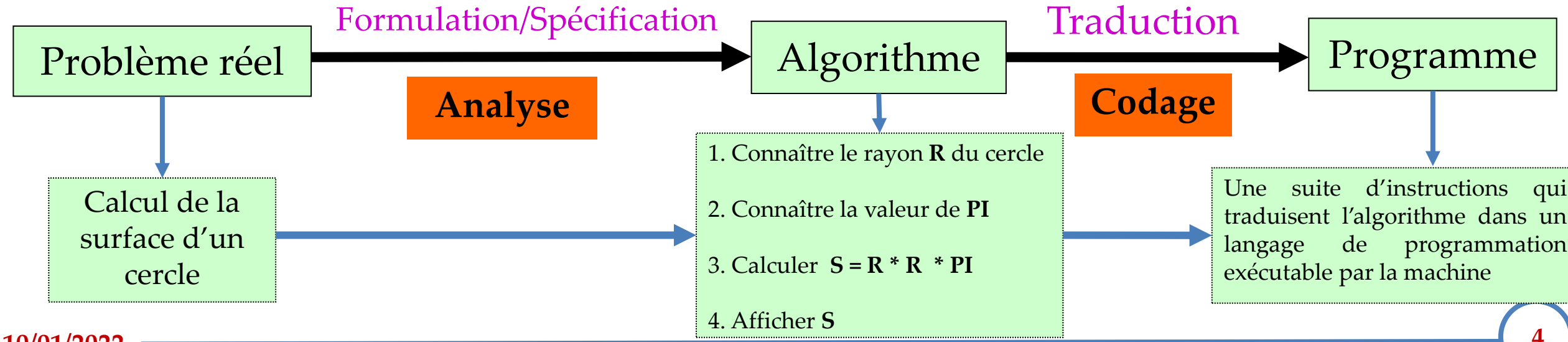
2) Algorithme

3) Les variables et les constantes

4) Les opérateurs

1. Introduction

- L'algorithmique est un terme d'origine arabe, hommage à Al Khawarizmi (780-850). Elle désigne aussi **la discipline** qui étudie **les algorithmes** et leurs applications en Informatique.
- Un algorithme est une méthode de résolution de problème énoncée sous la forme d'une série d'opérations à effectuer.
- La mise en œuvre de l'algorithme consiste en l'écriture de ces opérations dans un langage de programmation et constitue alors la brique de base d'un programme informatique.



1. Introduction

- Tout problème à programmer doit être résolu, d'abord sous forme d'algorithme, puis converti en programme dans un langage de programmation utilisé.

- Pour fonctionner, ***un algorithme doit donc contenir uniquement des instructions compréhensibles par celui qui devra l'exécuter***.

- Un programme est un enchainement d'instruction, écrit dans un langage de programmation, exécutées par un ordinateur, permettant de traiter un problème et de renvoyer des résultats. Il représente la traduction d'un algorithme à l'aide d'un langage de programmation.

2. Algorithme

2.1 Définition

Un algorithme est une suite finie d'instructions qu'on applique dans un ordre bien déterminé à un nombre fini de données pour résoudre un problème donné et arriver à un résultat.

2.2 Propriétés d'un algorithme

- Un algorithme doit:
  - avoir un ***nombre fini d'étapes.***
  - avoir un ***nombre fini d'opérations*** par étape.
  - ***se terminer*** après un nombre fini d'opérations.
  - fournir un ***résultat.***
- Chaque opération doit être:
  - ***définie*** rigoureusement et sans ambiguïté.
  - ***effective,*** c-à-d réalisable par une machine.

2.3 Etapes d'un algorithme

Pour écrire un algorithme on doit suivre les étapes suivantes:

1) Comprendre le problème
2) Spécifier le problème :
   a) Identifier les **données** ("entrées" ou "**input**" en anglais)
   b) Identifier les **résultats souhaités** ("sorties" ou "**output**" en anglais)
   c) Déterminer les **transformations à faire** sur ces données pour obtenir les résultats

Exemple : Etapes de réalisation de l'algorithme de calcul de la surface d'un cercle

1) Problème : on souhaite calculer la surface d'un cercle
2) Spécification du problème :
   a) **Données** : la constante **PI** et le rayon **R**
   b) **Résultats souhaités** : la surface **S** du cercle
   c) **Traitement à faire sur ces données** : Calculer **PI×R×R** et mettre le résultat dans **S**

2.4 Représentation d'un algorithme

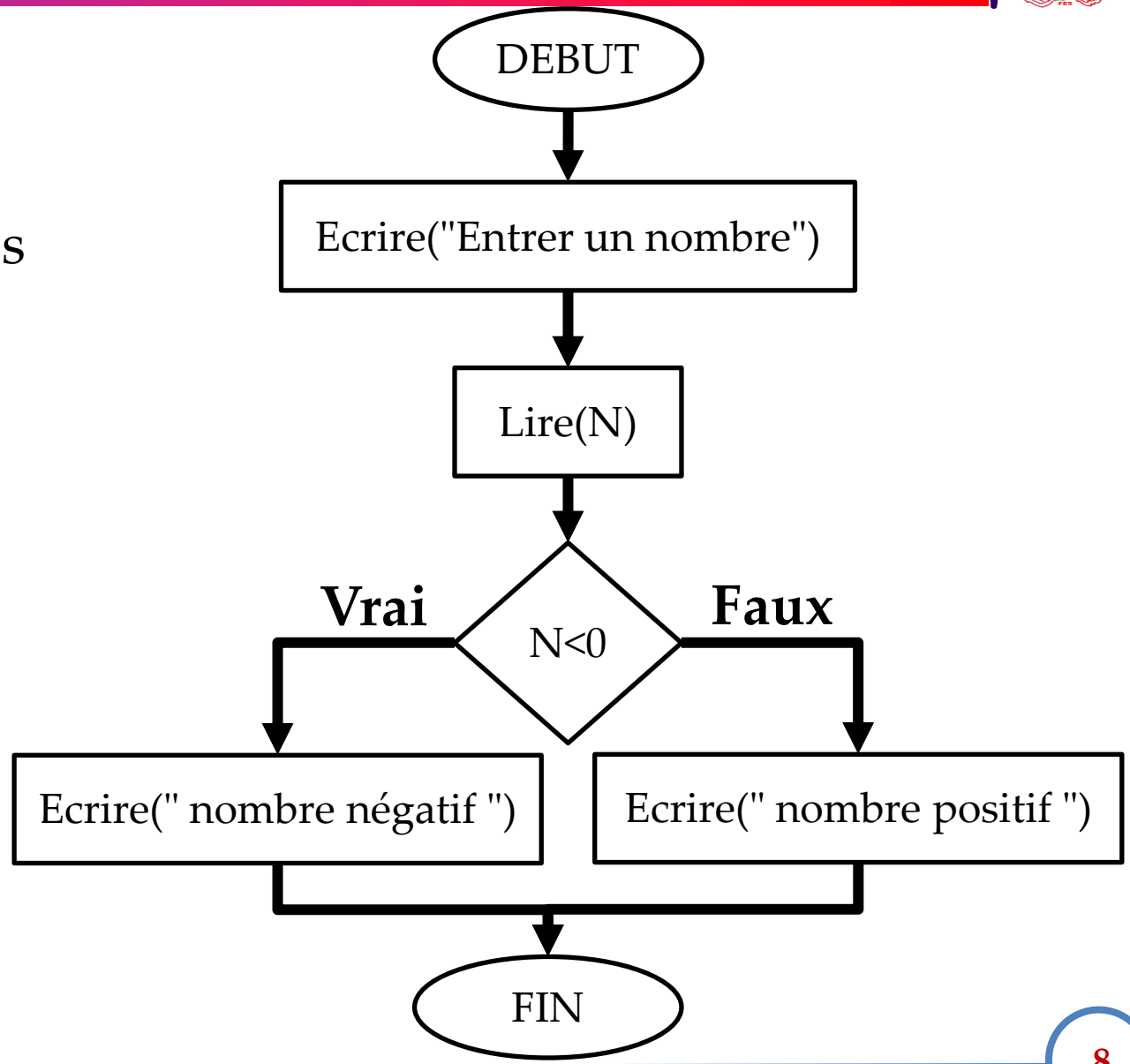
2.3.1 L'Organigramme

C'est une représentation graphique avec des symboles (carrés, losanges, etc.)

Exemple: l'organigramme qui montre le signe d'un nombre

Remarques:

- L'organigramme offre une vue d'ensemble de l'algorithme.
- C'est une représentation quasiment abandonnée aujourd'hui



2.3.2 Le pseudo-code

C'est une représentation textuelle avec une série de conventions ressemblant à un langage de programmation (sans les problèmes de syntaxe).

Exemple: Le pseudo code ci-contre permet de déterminer le signe d'un nombre

```
/*Déterminer le signe d'un nombre*/
Début
        Ecrire("Entrer un nombre");
        Lire(N);
        Si   N<0  Alors
        Ecrire("Nombre négatif");
        Sinon
        Ecrire("Nombre positif");
        FinSi
Fin.
```

Remarques:

- Le pseudo code est plus pratique pour écrire un algorithme.
- C'est une représentation largement utilisée aujourd'hui

2.5 Forme d'un algorithme

Un algorithme est constitué de trois parties principales :

- **L'en-tête** : cette partie sert à donner un nom à l'algorithme. Elle est précédée par le mot Algorithme;
- **La partie déclarative (déclaration des données)** : dans cette partie, on déclare les différents objets que l'algorithme utilise (constantes, variables, etc.) ;
- **Le corps de l'algorithme** : cette partie contient les instructions de l'algorithme. Elle est délimitée par les mots **début** et **fin**.

Remarque:
Lorsqu'un algorithme devient long, il est conseillé d'ajouter des lignes de commentaires dans l'algorithme (/*Commentaire*/), c'est-à-dire des lignes qui servent à donner des indications sur les instructions effectuées.

2.5 Forme d'un algorithme

Algorithme **Nom_de_l'algorithme;**

Déclaration des données
- **Variables**
- **Constantes**
- **Structures**
- **Fonctions**
- **Procédures**

Début

Corps de l'algorithme
(Instructions de traitement)
- Lecture des données
- Instructions à exécuter
- Résultats à afficher

Fin.

2. Algorithme

Exemple : Algorithme de calcul d'un surface d'un cercle

Algorithme **surface_cercle ;**

Constante PI = 3,14; /*Déclaration de la constante PI*/

Variable R : réel; /*Déclaration de la variable R*/

Variable S : réel; /*Déclaration de la variable S*/

Début

Lire(R); /*Cette instruction permet de saisir le contenu de R*/

S ← PI * R * R; /*Cette instruction permet de mettre le résultat du produit dans le contenu de la variable S*/

Ecrire(S); /*Cette instruction permet d'afficher le contenu de S*/

Fin.

3. Les variables et les constantes

- Un algorithme comprend des variables et des constantes représentés par des symboles.

Exemple: Dans l'algorithme du cercle, les variables sont : **S** et **R**, la constante est : **PI=3,14**;

- Dans les langages de programmation, une variable ou une constante sert à stocker la valeur d'une donnée.
- Une variable désigne en fait un emplacement (ou une adresse) mémoire dont le contenu peut changer au cours d'un programme (d'où le nom variable).
- Une **variable** est défini par :
  - Un **nom** : Composer de lettres et chiffres.
  - Un **type** : caractérise l'ensemble des valeurs que peut prendre la variable(entier, réel, caractère, chaîne de caractères, …) et l'espace mémoire occupé par la variable.
  - Une **valeur** : représente l'état actuel de la variable.

3. Les variables et les constantes

3.1 Déclaration des variables

- La **partie déclarative (déclaration des données)** consiste à énumérer toutes les variables dont on aura besoin au cours de l'algorithme.
- Chaque déclaration doit comporter le nom de la variable (identificateur) et son type.
- La syntaxe d'une déclaration de variable numérique en pseudo-langage aura la forme: **Variable nom_variable : type;**

Exemple:

Variable volume : réel.
Variables a,b,c,d :entiers.
Variable Nom_prenom : chaîne.
Variable présent : logique.

Remarques:

- Le nom de la variable doit obligatoirement commencer par une lettre suivie d'une suite de lettres et de chiffres et il ne doit pas contenir d'espace.
- Le contenu d'une variable peut être modifié par une action.

3. Les variables et les constantes

3.2 Type des variables

Les types de variables définissent l'ensemble des valeurs susceptibles d'être prise par une variable et les opérations possibles sur cette variable. Les **types de bases** sont :

- **Type entier :** sert à manipuler les nombres entiers positifs ou négatifs.
- **Type réel :** quant à lui sert à manipuler les nombres à virgules.
- **Type caractère:** Les variables de ce type peuvent contenir tous les caractères du code ASCII (lettres majuscules, minuscules, chiffres, symboles, …)

Exemples : 'A', 'a', '1', '?', '453'…

- **Type chaîne de caractère :** Une variable de ce type peut contenir une suite de caractères concaténés (les uns à cotés des autres).

Exemples: " Nom, Prénom", "code postale: 1000", …

- **Type logique ou booléen**: Une variable de ce type ne peut contenir que deux valeurs **VRAI** ou **FAUX**.

3.3 Déclaration des constantes

- Comme pour les variable, à une constante correspond un emplacement mémoire réservé auquel on accède par le nom qui lui a été attribué, mais dont la valeur stockée ne sera jamais modifiée au cours du programme.

- La syntaxe d'une déclaration d'une constante en pseudo-langage aura la forme:

Constante nom_constante=valeur;

Exemple:

Constante PI=3,14;

4. Les opérateurs

Les opérateurs permettent de produire un effet sur les variables de l'algorithme en effectuant des calculs, des vérifications d'égalités ou d'inégalités. Ils sont divisés en trois classes :

4.1 Les opérateurs arithmétiques

Opérateur	Signification	syntaxe
+	addition	x+ y
-	soustraction	x - y
*	multiplication	x * y
/	division	x/y
mod	modulo	x mod y
div	Division Euclidienne	x div y

- **x mod y :** le reste de la division euclidienne de x par y
- **x div y :** le quotient de x par y.

Exemple :

- **(15 mod 4) = 3**
- **(10 div 2) = 5**
- **(11 div 2) = 5**

4. Les opérateurs

4.2 Les opérateurs de comparaison

Opérateur	Signification	syntaxe
=	égal	x = y
<>	différent	x <> y
<	inférieur	x < y
>	supérieur	x > y
<=	Inférieur ou égal	x <= y
>=	Supérieur ou égal	x >= y

4.3 Les opérateurs logique

Opérateur	Signification	syntaxe
et	Intersection	x et y
ou	réunion	x ou y
non	complémentaire	non y

4.4 Opérateur alphanumérique : &

Cet opérateur permet de concaténer, autrement dit d'agglomérer, deux chaînes de caractères.

Exemple: on considère l'algorithme suivant:

```
Algorithme  Concaténation
        Variables A, B, C : chaine
Début
        A ← " Science"
        B ← "Physique"
        C ← A & B
Fin
```

La valeur de C à la fin de l'exécution de l'algorithme est " Science Physique"