Support de cours
Module : ASD1

Chapitre 1: Introduction à l'algorithmique

Réalisé par:
Dr. Sakka Rouis Taoufik
https://github.com/srtaoufik/Cours-ASD

1

Qu'est-ce qu'un algorithme ?

- Un algorithme est la description d'une méthode pour effectuer une tâche précise : l'algorithme prend en compte des données de départ et donne les instructions à effectuer pour parvenir au résultat cherché.
- La façon dont les instructions sont données importe peu, mais il est important que ces instructions ne soient pas ambigües et soient valables pour toutes les données de départ.

2

Qu'est-ce une structure de données?

- Une structure de données est un format spécial destiné à **organiser, traiter, extraire** et **stocker** des données.
- S'il existe plusieurs types de structures plus ou moins complexes, tous visent à organiser les données pour répondre à un besoin précis, afin de pouvoir y accéder et les traiter de façon appropriée.

3

Qu'est-ce une structure de données?

- En informatique, une structure de données peut être sélectionnée ou conçue pour stocker des données de manière à pouvoir manipuler ces dernières à l'aide de **plusieurs algorithmes**.
- Chaque structure de données contient des informations sur la valeur des données, les relations entre elles et les fonctions applicables.
- Exemple: sur la structure de donnée **tableau d'entiers en** peut appliquer des algorithmes de recherche ou de tri.

4

Quel langage choisir ?

- Cela dépend à qui l'algorithme est destiné :

  – Si c'est à une personne on utilisera sa langue habituelle,

  – Si c'est à un ordinateur on utilisera un langage de programmation particulier pour cette machine.

- Ici, on suppose que l'on va traduire les algorithmes en des langages de même type qui seront compréhensibles par différentes machines.

On choisira le langage de programmation C pour traduire les algorithmes et ainsi de pouvoir les faire fonctionner.

5

Structure d'un algorithme

- Un algorithme est une séquence d'instructions qui décrit comment résoudre un problème particulier. Tout algorithme est composé de deux parties :

- **Déclaration** des nouveaux types, des constantes et des variables utiles pour la résolution du problème.

- **Traitement :** cette partie est elle-même composée de 3 parties :

Préparation du traitement : initialisation des données nécessaires à la résolution du problème.

Traitement : résolution pas à pas, après décomposition en sous-problèmes si nécessaire

Edition des résultats : impression à l'écran, dans un fichier, etc.

6

Structure d'un algorithme

Algorithme ElèveAuCarré
{déclarations des données (constantes et variables)}
Variables
leNombre, sonCarré**: entiers**
Début
{préparation du traitement}
Ecrire ("Quel nombre voulez-vous élever au carré?")
Lire (leNombre)
{traitement : calcul du carré}
sonCarré ←leNombre * leNombre
{présentation du résultat}
Ecrire ("Le carré de ", leNombre,"est ", sonCarré)
Fin

Exemple 1: Cet algorithme calcule le carré du nombre que lui fournit l'utilisateur.

7

Déclaration de constante

Les constantes sont des données dont la valeur reste fixe durant l'exécution du programme.

Exemples :

Constantes

Pi = 3.14

G = 9.80

8

Déclaration de constante

Algorithme Cercle
Constantes
pi = 3.14
Variables
r, p, s : Réel
Début
Ecrire(”Entrer le rayon du cercle : ”)
Lire(r)
P ← 2 * pi * r
S← pi * r * r
Ecrire (”Périmètre = ”, p)
Ecrire (”Surface = ”, s)
Fin

Exemple 2: Cet algorithme permet le calcul du périmètre et de la surface d’un cercle.

9

Déclaration de variable

- En programmation, une **variable** est un **identificateur** qui sert à repérer un emplacement donné de la mémoire centrale. Cette notion nous permet de manipuler des valeurs sans nous préoccuper de l’emplacement qu’elles occupent effectivement en mémoire.
- La lisibilité des programmes dépend étroitement du choix des noms des variables qui doivent être simples et significatifs. Ainsi, Mont_Fac est un meilleur choix que X pour désigner le montant d’une facture.
- Comme tout ***identificateur***, le nom d’une variable est formé d’une ou de plusieurs lettres ; les chiffres sont également autorisés, à condition de ne pas les mettre au début du nom.
- ➔On peut considérer une variable comme **une donnée dont la valeur est modifiable**.

10

Notion de type

- A chaque variable utilisée dans le programme, il faut associer **un type** qui permet de définir :
  - l'ensemble des valeurs que peut prendre la variable
  - l'ensemble des opérations qu'on peut appliquer sur la variable.
- La syntaxe de l'action de déclaration des variables est la suivante :

Variables

Variable_1, Variable_2, … **: Type1**

Variable_3, Variable_4, … **: Type2**

11

Notion de type

- Les principaux types utilisés en algorithmique sont :
  - le type entier
  - le type réel
  - le type caractère
  - le type chaîne de caractères
  - le type logique ou booléen.

12

Le type entier

- Une variable est dite entière si elle prend ses valeurs dans Z (ensemble des entiers relatifs).
- Les principales opérations définies sur les variables de type entier sont :

Opération	Notation
Addition	+
Soustraction	-
Multiplication	*
Division entière	div
Modulo (reste de la division)	mod

13

Le type entier

- **Exemples**

  13 **div** 5 = 2

  13 **mod** 5 = 3

- L'ensemble de valeurs que peut prendre un entier varie selon le langage de programmation utilisé étant donné que le nombre de bits réservés pour une variable de ce type n'est pas le même.
- A titre d'exemple: en Turbo Pascal, les entiers varient entre –32768 et +32767.

14

Le type réel ou décimal

- Il existe plusieurs types de réels représentant chacun un ensemble particulier de valeurs prises dans **R** (ensemble des nombres réels).
- Ici encore, cette distinction se justifie par le mode de stockage des informations dans le langage de programmation.
- **Exemples :**

Variables

A : réel

B : double

15

Le type caractère

- Un caractère peut appartenir au domaine des chiffres de ”0” à ”9”, des lettres (minuscules et majuscules) et des caractères spéciaux (”*”, ”/”, ”{”, ”$”, ”#”, ”%” …).

- Un caractère sera toujours noté entre des guillemets. Le caractère espace (blanc) sera noté ” ”.

16

Le type caractère

- Les opérateurs définis sur les données de type caractère sont :

Opération	Notation
Égal	=
Différent	#
Inférieur	<
Inférieur ou égal	<=
Supérieur	>
Supérieur ou égal	>=

- La comparaison entre les caractères se fait selon leurs codes ASCII. Par exemple :

" " < "0" < "1" < "A" < "B" < "a" < "b" < "{"

17

Le type logique ou booléen

- Une variable logique ne peut prendre que les valeurs "Vrai" ou "Faux".
- Elle intervient dans l'évaluation d'une condition.
- Les principales opérations définies sur les variables de type logique sont :
  - la négation (**NON**)
  - l'intersection (**ET**)
  - l'union (**OU**).

18

Déclaration de nouveau type

- En plus de ces types prédéfinis, le programmeur a la possibilité de définir lui-même de nouveaux types en fonction de ses besoins.

Types

Saison = (”A”,”H”,”P”,”E”)

Tnote = 0 .. 20

Variables

s : Saison

note : Tnote

…

La variable s de type saison ne peut prendre que les valeurs : ”A”, ”H”, ”P” ou ”E”.

19

L’instruction d’affectation

- Le rôle de cette instruction consiste à placer une valeur dans une variable.
- Sa syntaxe générale est de la forme :

  Variable←Expression

- L’instruction : **A ← 6**

signifie « mettre la valeur 6 dans la case mémoire identifiée par A ».

- L’instruction : **B ← (A + 4) Mod 3**

signifie «range dans B la valeur 1 (A toujours égale à 6) ».

20

L'instruction d'affectation

- **Remarque**

La valeur ou le résultat de l'expression à droite du signe d'affectation doit être de même type ou de type compatible avec celui de la variable à gauche.

21

L'instruction d'affectation

Exercice d'application 1:

Quelles seront les valeurs des variables A, B et C après l'exécution des instructions suivantes :

A ← 5

B ← 3

C ← A + B

A ← 2

C ← B – A

22

L'instruction d'affectation

- **Exercice d'application 2:**

Donner toutes les raisons pour lesquelles l'algorithme suivant est incorrect :

Algorithme Incorrect

x, y : Entier

z : Réel

Début

z ← x + 2

y ← z

x*2 ← 3 + z

y ← 5y + 3

Fin.

23

L'instruction de lecture

- Dans certains cas, nous pourrons être amenés à transmettre des informations (données) à notre programme par l'intermédiaire d'un **périphérique d'entrée comme le clavier.**
- Cela sera réalisé par l'instruction de lecture. La forme générale de cette instruction est :

  Lire (Variable1, Variable2, ...)

- Ainsi, l'instruction : **Lire (A)**

signifie « lire une valeur à partir du périphérique d'entrée, qui est le clavier par défaut, et la ranger dans la variable identifiée par A »

24

L'instruction d'écriture

- Pour qu'un programme présente un intérêt pratique, il devra pouvoir nous communiquer un certain nombre d'informations (résultats) par l'intermédiaire d'un **périphérique de sortie comme l'écran ou l'imprimante**.
- C'est le rôle de l'instruction d'écriture.
- La forme générale de cette instruction est :

  Ecrire (expression1, expression2, ...)

25

Exercices d'application

- **Exercice 1:**

Écrire un algorithme qui permute la valeur de deux variables c1 et c2 de type caractère.

Solution : Le principe est d'utiliser une variable intermédiaire (tout comme on utilise un récipient intermédiaire si l'on veut échanger le contenu de deux bouteilles).

- **Exercice 2:**

Écrire un algorithme qui affiche le cube d'un nombre réel saisi au clavier.

- **Exercice 3:**

Écrire un algorithme qui calcule et affiche la résistance d'un composant électrique en utilisant la loi d'Ohm :

U=R*I avec I: Intensité en Ampère (A), R : Résistance en Ohm (Ω) et U : Tension en Volte (V)

26

Exercices d'application

- **Exercice 4:** Écrire un algorithme qui lit deux entiers au clavier et qui affiche ensuite leur somme et leur produit.
- **Exercice 5:** Quelles seront les valeurs des variables A et B après l'exécution des instructions suivantes :

  A← 5
  B←7
  A← A + B
  B ←A - B
  A←A – B
- **Exercice 6:** Soient 3 variables A, B et C. Écrire une séquence d'instructions permettant de faire une permutation circulaire de sorte que la valeur de A passe dans B, celle de B dans C et celle de C dans A. On utilisera une seule variable supplémentaire.

27