Chapitre 1

Algorithmique et programmation

1 Initiation à l'algorithmique

Définition

Un algorithme est une suite finie d'instructions permettant de résoudre un problème donné.

Exemples :

• Dans la vie courante : une recette de cuisine, un mode d'emploi pour construire un meuble, un trajet sur une carte routière, une notice d'utilisation ...etc.

• Dans le domaine des mathématiques : déterminer le PGCD de deux nombres (algorithme d'Euclide), résoudre une équation par approche de la solution, calculer une longueur, calculer une image, déterminer à partir de quel moment une expression algébrique dépasse une valeur etc.

Utilités

Les algorithmes peuvent permettre d'effectuer des tâches répétées mais également de résoudre des problèmes plus facilement "qu'à la main". Tous les objets de la vie moderne sont gérés par des algorithmes. Les ordinateurs, les logiciels, Internet, les moteurs de recherches, les calculatrices, les téléphones portables, les objets connectés, etc. Il est donc important d'apprendre à les comprendre et à les construire.

Un algorithme doit être compréhensible pour tout le monde. Il est ensuite traduit dans un programme dans un langage informatique afin qu'une machine puisse l'exécuter simplement et avec efficacité.

Définition

Un programme est donc la traduction d'un algorithme dans un langage adapté à la machine utilisée.

Les langages les plus courants sont Python, C, C++, Java, Php, Pascal, Lisp.

Ce chapitre n'abordera que les algorithmes mais à la fin du livre, je donne quelques traductions d'instructions, dans certains langages que vous serez succeptibles d'utiliser au cours de votre scolarité.

Propriété 1

▷ Un algorithme se présente souvent sous la forme suivante :

Nom de l'algorithme :	
Déclaration des variables :	
	Listes, noms et description des variables que l'on va utiliser dans l'algorithme.
Initialisation :	
	Si besoin, lorsque l'on doit donner une valeur initiale à une variable.
Traitement :	
	Liste des instructions pour répondre au problème.
Sortie :	
	Affichage du (ou des) résultat(s) attendu(s).

Exemple
L'instruction $a \mapsto X$ signifie que la valeur de la variable X reçoit la valeur de la variable a.

Algorithme 1 :	
Déclaration des variables :	
	X, Y : des nombres réels.
Initialisation :	
	Saisir la valeur de X
Traitement :	
	$X - 2 \mapsto Y$
	$Y \times Y \mapsto Y$
	$3 \times Y \mapsto Y$
	$4 - Y \mapsto Y$
Sortie :	
	Afficher la valeur de Y.

L'algorithme 1 (précédent) permet de calculer l'image d'un réel x par la fonction $f : x \mapsto 4-3(x-2)^2$.
L'instruction $X-2 \mapsto Y$ signifie que la variable Y prend la valeur de la variable $X-2$
L'algorithme peut être traduit de cette façon :

$$X \to X-2 \to (X-2)^2 \to 3(X-2)^2 \to 4-3(X-2)^2$$

2 Tests et boucles

2.1 Les tests

Pour résoudre certains problèmes il est important, dans certains cas, de faire des tests pour savoir si l'on doit effectuer une instruction ou pas. Par exemple, si on doit calculer un inverse, il faut tester si le nombre est différent de 0 ou pas. Si on veut calculer une racine carrée d'un réel, il faut tester si le nombre est positif ou pas. Si on veut savoir si un nombre est pair, il faut tester si le reste par la division euclidienne par 2 est nulle ou pas, etc.

Définition
Effectuer un test, revient à écrire **une instruction conditionnelle**. L'instruction conditionnelle effectue des instructions à condition qu'un test soit validé. Dans un algorithme, on code l'instruction conditionnelle de la façon suivante :

Si **condition validée**
Alors
instruction 01
instruction 02
instruction 03
Sinon
instruction 04
instruction 05
instruction 06
Fin du si

Exemple

Algorithme 2	Ecart entre entiers
Déclaration des variables :	
	X, Y, Ecart : des nombres entiers.
Initialisation :	
	Saisir la valeur de X
	Saisir la valeur de Y
Traitement :	
Si $X \leq Y$	
Alors	
	$Y - X \mapsto$ Ecart
Sinon	
	$X - Y \mapsto$ Ecart
Fin du Si	
Sortie :	
	Afficher la valeur "Ecart".

2.2 Les boucles

Définition

Lorsque l'on doit répéter une instruction un nombre de fois connu à l'avance , on utilise **une boucle itérative**.
Dans un algorithme, une boucle itérative est codée de la façon suivante :

Pour variable allant de	la valeur Début à la valeur Fin
Faire	
	instruction 01
	instruction 02
	instruction 03
Fin du Pour	

La variable utilisée dans la boucle "Pour", est appelée un "compteur" et à chaque étape sa valeur est automatiquement augmentée de 1.

Exemple

Algorithme 3	Somme des 100 premiers entiers
Déclaration des variables :	
	S, I : des nombres entiers.
Initialisation :	
	$0 \mapsto S$
Traitement :	
Pour I allant de 1 à 10	Faire
	$S \mapsto S + I$
Fin du Pour	
Sortie :	
	Afficher la valeur de S.

Définition

Lorsque l'on doit répéter une instruction sans connaître le nombre d'itérations, on utilise **une boucle conditionnelle**. La boucle est répétée tant que la condition indiquée est vérifiée.
Dans un algorithme une boucle conditionnelle est codée de la façon suivante :

Tant que condition vérifiée	
Faire	
	instruction 01
	instruction 02
	instruction 03
Fin du Tant que	

On utilise réguliérement des compteurs à l'intérieur de la boucle conditonnelle, mais il faut faire attention à bien l'initialiser et à l'incrémenter à l'intérieur de la boucle..

Exemple

Algorithme 4	Somme des 100 premiers entiers
Déclaration des variables :	
	S, I : des nombres entiers.
Initialisation :	
	$0 \mapsto S$
	$0 \mapsto I$
Traitement :	
Tant que $I \leq 10$	Faire
	$S + I \mapsto S$
	$I + 1 \mapsto I$
Fin du Tant que	
Sortie :	
	Afficher la valeur de S.

3 Quelques algorithmes importants

3.1 Algorithme donnant la solution d'une équation par dichotomie

On note f une fonction définie sur $[a,b]$.
On souhaite écrire un algorithme qui permet de déterminer une valeur approchée à 10^{-n} près de la solution de l'équation $f(x) = k$ ou k est un nombre appartenant à l'ensemble d'arrivée $f([a,b])$.
Première remarque : Si $f(x) = k$ alors $f(x) - k = 0$ donc il suffit de savoir écrire l'algorithme pour $f(x) = 0$ et de remplacer $f(x)$ par $f(x) - k$ pour obtenir la valeur cherchée.
Deuxième remarque : On suppose aussi que la fonction a au moins, pour 0, un antécédent et qu'elle change de signe avant et après cet antécédent. On suppose donc que $f(a)$ et $f(b)$ ont un signe différent et donc que $f(a) \times f(b) < 0$.
On va utiliser une boucle conditionnelle (Tant que) contenant un test conditionnel (Si). Tant que l'écart entre a et b est plus grand que 10^{-n} alors on calcule le milieu de l'intervalle $[a,b]$ qui est $m = \dfrac{a+b}{2}$. Si $f(a)$ et $f(m)$ ne sont pas de même signe alors b prend la valeur de m et on recommence, sinon a prend la valeur de m et on recommence. A la fin, on obtient un intervalle de dimension inférieure à 10^{-n} et qui contient la solution cherchée.
On nomme cette méthode de recherche, **un algorithme par dichotomie** (division en deux parties).

Algorithme 5	Solution Equation Dichotomie
Déclaration des variables :	
	a, b, m : des nombres réels.
	n : un nombre entier positif
Initialisation :	
	Donner la valeur de n
Traitement :	
Tant que $b-a \geq 10^{-n}$	Faire
	$\dfrac{a+b}{2} \mapsto m$
	Si $f(a) \times f(m) \leq 0$ alors
	$m \mapsto b$
	Sinon
	$m \mapsto a$
	Fin du Si
Fin du Tant que	
Sortie :	
	Affichage des valeurs de a et b.

3.2 Algorithme du seuil

On note f une fonction strictement croissante sur $[0;+\infty[$.
On cherche à déterminer, à partir de quel nombre entier positif, les images sont toutes supérieures à une valeur seuil A.
On va utiliser une boucle conditionnelle (tant que). On initialise un compteur à 0 et tant que l'image de ce compteur est plus petite que A alors on ajoute 1 au compteur et on recommence.
Exemple :
On veut savoir à partir de quel entier $f(x) = x^2 + x + 1$ est supérieur à A, où A est un nombre que l'on va donner au début.

Algorithme 6	Algorithme du seuil
Déclaration des variables :	
	A : un nombre réel.
	C : un nombre entier positif
Initialisation :	
	Donner la valeur de A
	$0 \mapsto C$
Traitement :	
Tant que $C^2 + C + 1 < A$	Faire
	$C + 1 \mapsto C$
Fin du Tant que	
Sortie :	
	Afficher la valeur de C.

3.3 Exercices

On note S la somme $S = 1 + 2 + 3 + ... + (n-1) + n$
et P le produit $P = 1 \times 2 \times 3 \times ... \times (n-1) \times n$

Exercice 1

Construire un algorithme qui demande une valeur de n, entier positif, et qui affiche en sortie la somme S.

Correction

Algorithme 7	Somme des n premiers entiers
Déclaration des variables :	
	S, I, n : des nombres entiers positifs.
Initialisation :	
	Donner une valeur de n
	$0 \mapsto S$
Traitement :	
Pour I allant de 1 à n	Faire
	$S + I \mapsto S$
Fin du Pour	
Sortie :	
	Afficher la valeur de S.

Exercice 2

Construire un algorithme qui détermine à partir de quelle valeur de n, la somme S est supérieure ou égale à 10000.

Correction

Algorithme 8	Seuil et somme
Déclaration des variables :	
	S, n : des nombres entiers positifs.
Initialisation :	
	$0 \mapsto S$
	$0 \mapsto n$
Traitement :	
Tant que $S < 10000$	Faire
	$n+1 \mapsto n$
	$S+n \mapsto S$
Fin du tant que	
Sortie :	
	Afficher la valeur de n.

Exercice 3

Construire un algorithme qui demande une valeur de n, entier positif, et qui affiche en sortie le produit P.

Correction

Algorithme 9	Produit des n premiers entiers
Déclaration des variables :	
	P, I, n : des nombres entiers positifs.
Initialisation :	
	Donner une valeur de n
	$1 \mapsto P$
Traitement :	
Pour I allant de 1 à n	Faire
	$P \times I \mapsto P$
Fin du Pour	
Sortie :	
	Afficher la valeur de P.

Attention : Il faut bien initialiser la variable P à 1 sinon le résultat donnera toujours 0 puisque le premier facteur est nul :

$$0 \times 1 \times 2 \times 3 \times ... \times n = 0$$

Exercice 4

Construire un algorithme qui détermine à partir de quelle valeur de n le produit P est supérieur ou égal à 10000.

Correction

Algorithme 10	Seuil et produit
Déclaration des variables :	
	P, n : des nombres entiers positifs.
Initialisation :	
	$1 \mapsto S$
	$0 \mapsto n$
Traitement :	
Tant que $S < 10000$	Faire
	$n+1 \mapsto n$
	$S \times n \mapsto S$
Fin du Tant que	
Sortie :	
	Afficher la valeur de n.