Algorithmes (1)
Quelques concepts de base

Objectifs :

- Prendre conscience de démarches algorithmiques, description de certaines procédures algorithmiques (la « pensée algorithmique »)

- Faire comprendre la logique de la rédaction d'un algorithme simple

- Traduire des programmes de calcul simples en algorithmes

I. Définition

1°) Qu'est-ce qu'un algorithme ?

Un **algorithme** est une « recette » qui permet d'écrire un résultat.

Un algorithme est un enchaînement d'étapes ou d'instructions à utiliser dans un certain ordre et dont la réalisation va permettre la résolution d'un problème donné.

2°) Exemple : calcul de l'hypoténuse d'un triangle rectangle connaissant les longueurs des deux côtés de l'angle droit

On va utiliser le théorème de Pythagore.

On va rédiger un **programme de calcul** (qui n'est pas à proprement parler un algorithme).

On décortique toutes les étapes.

→ On calcule le carré d'un côté de l'angle droit.
→ On calcule le carré de l'autre côté de l'angle droit.
→ On calcule la somme des deux (on additionne les deux).
→ On prend la racine carrée du résultat.
→ On obtient la longueur de l'hypoténuse.

N.B. : Les mots « côtés » et « hypoténuse » sont pris au sens de « longueur de côtés » et « longueur de l'hypoténuse ».

3°) Remarque

Un algorithme doit être lisible par tous et après, il est codé en **langage informatique** (cf. cours Algorithmes (2)) afin qu'une machine (calculatrice ou ordinateur) puisse l'exécuter.

Aujourd'hui, les algorithmes prennent place dans tous les domaines de l'activité mathématique comme nous le verrons cette année.

II. Qu'est-ce qui constitue un algorithme ?

1°) Structure générale d'un algorithme

- Un algorithme a un **début** et une **fin**.
Il est constitué d'un **nombre fini d'étapes**.
Il faut signaler le **début** et la **fin**.

- Les différentes étapes vont agir et on va appeler ces étapes des **instructions**.

On effectue un calcul.
On entre une donnée.
On lit le contenu d'une variable.

2°) Définition d'une variable numérique dans un algorithme

Un algorithme fait intervenir un certain nombre de **variables**.
Il peut y avoir autant de variables que l'on veut mais dans un algorithme, il y a toujours au moins une variable.

Pendant l'exécution d'un algorithme, on a besoin de **stocker** des données.
Pour cela, on a besoin de **variables**.
On attribue un **nom** à chaque variable.

Une variable est une **boîte** repérée par un nom (une lettre), qui va contenir une information.

3°) Remarques

Le contenu d'une variable en informatique peut être modifiée pendant l'exécution de l'algorithme.

Dans l'écriture d'un algorithme, il est utile de préciser dès le début le nom des variables que l'on va utiliser en indiquant leur genre. C'est l'étape que l'on appelle **déclaration des variables**.

La notion de variable (informatique) est différente de celle de la notion de variable en mathématique (pour les fonctions). La différence est que dans un algorithme, une variable peut évoluer au fur et à mesure du déroulement de l'algorithme.

III. Comment se présente un algorithme

Déclaration des variables

a, nombre réel
b, nombre entier

Début

….

Instructions

….

Fin

IV. Les différents types d'instructions dans un algorithme

- **Saisir :** instruction qui permet à l'utilisateur de donner une valeur

- **Affecter :** instruction qui permet à l'algorithme d'attribuer une valeur à une variable

- **Afficher :** instruction qui affiche le résultat

V. Exemple qui permet de comprendre les paragraphes précédents

Un vendeur fait une facture à un client qui a acheté le même article en plusieurs exemplaires.

P : quantité d'articles achetés
Q : prix unitaire

Déclaration des variables :
P, réel positif
Q, réel positif
R, réel positif

Début

Instructions :
Saisir P
Saisir Q
Affecter à R la valeur P $\times$ Q
Afficher R

Fin

On rédigera plutôt l'algorithme en langage naturel selon le modèle suivant :

Variables :
P, Q, R, réels positifs

Entrées :
Saisir P
Saisir Q

Traitement :
Affecter à R la valeur P $\times$ Q

Sortie :
Afficher R

On notera dans cet exemple les grandes étapes :
- déclaration des variables précisant leurs types (cette étape sera parfois omise par la suite)
- entrée(s)
- traitement
- sortie(s).

Généralement, ces étapes seront toujours respectées. Elles permettent de structurer l'algorithme et de faciliter sa compréhension.
D'autres règles seront étudiées dans la suite du cours.

Un commentaire (emprunté à la revue Chantiers 154-155) :

Nous avons respecté le code de rédaction imposé par le programme.
On peut cependant émettre une petite critique concernant cette rédaction.
En effet, les instructions d'entrée (Saisir ou Lire) ou de sortie (Afficher) semblent faire référence à une machine et à un utilisateur.
Il n'y a pas de raison dans la plupart des cas que nous étudierons d'introduire dans l'écriture des algorithmes du vocabulaire lié à l'utilisation d'une machine (instructions dites d'entrées / sorties), qui peuvent entretenir dans l'esprit des élèves une confusion entre algorithme « idéal » et programme concret.

Une proposition de meilleure rédaction (qui ne sera cependant pas retenue pour nous) :

Variables :
P, Q, R, réels positifs

Entrées :
P, Q

Traitement :
Affecter à R la valeur P $\times$ Q

Sortie :
R

Cependant, n'oublions pas qu'un algorithme est en général destiné à être programmé, ce que nous ferons dans le chapitre suivant sur des exemples très simples.
Il ne faudra alors pas confondre l'« algorithme » (rédigé en langage naturel) et le « programme » (rédigé en dans un certain langage de programmation).

L'intérêt de notre rédaction en langage naturel est de pouvoir s'adapter à tout type de langage de programmation.

VI. Comment présenter un algorithme

1°) Organigramme

Donne une bonne image mentale de l'algorithme quand celui-ci n'est pas trop compliqué.

Il y a quelques conventions à respecter.

Voir exercices et chapitre suivant.

2°) Rédaction en langage naturel

Comme mentionné ci-dessus, la rédaction respecte certaines règles.

Tout d'abord, la structure générale est marquée par les rubriques : Entrée, Traitement, Sortie.

Ensuite, on notera que :

- les instructions constituant l'entrée (ou les entrées) sont rédigées avec le verbe « Saisir » ou « Lire » ;

- les instructions constituant la sortie (ou les sorties) sont rédigées avec le verbe « Afficher ».

Nous avons mentionné ci-dessus les critiques que l'on peut émettre sur ce mode de rédaction.
Nous nous y confirmerons cependant dans les chapitres suivants.

VII. Historique

1°) La pensée algorithmique

Depuis l'origine, les mathématiciens ont cherché à décrire des méthodes applicables dans tous les cas.
De telles démarches algorithmiques sont enseignées depuis l'école primaire pour résoudre un certain nombre de problèmes.

2°) Origine du mot « algorithme »

Le mot « algorithme » vient du nom du nom d'un mathématicien arabe Al-Khawarizmi.

Al-Khawarizmi, né vers 783, originaire de Khiva dans la région du Khwarezm qui lui a donné son nom, mort vers 850 à Bagdad, est un mathématicien, géographe, astrologue et astronome musulman perse dont les écrits, rédigés en langue arabe, ont permis l'introduction de l'algèbre en Europe.

On pourra chercher d'autres renseignements sur Wikipedia.

3°) L'algorithme d'Euclide

Cet algorithme célèbre a été étudié au collège pour déterminer le PGCD de deux entiers naturels.
Cet algorithme, qui n'est pas très simple, sera revu plus tard lors de l'étude des boucles « Tantque ».

4°) Expansion de l'algorithmique

Avec les développements de l'informatique, les algorithmes sont partout présents aujourd'hui.

VIII. Utilisation de listes et de nombres aléatoires

1°) Utilisation de listes

Dans un algorithme, on doit parfois saisir plusieurs nombres.
L'utilisation de listes peut alors être commode, parfois même indispensable.
Les nombres d'une liste L contenant n valeurs sont notés L(1), L(2), … L(n) ou parfois L[1], L[2], … L[n].

On appelle **dimension** d'une liste le nombre d'éléments que l'on peut mettre dans cette liste.

On peut définir une liste comme un tableau horizontal (il faut cependant noter que sur une calculatrice une liste est représentée en tableau, verticalement).

2°) Utilisation de nombres aléatoires

Un algorithme, lorsqu'il s'agit de simuler une expérience aléatoire, fait intervenir des nombres aléatoires, notamment des nombres aléatoires dans [0 ; 1[(touche Random).
En fait, il s'agit de nombres pseudo-aléatoires, les calculatrices ou les logiciels étant munis de programmes permettant de reconstituer une illusion de hasard (« fonction de hasard »).

Nous verrons plus tard comment il est possible générer des entiers aléatoires à partir de nombres au hasard dans [0 ; 1[.

Exercice guidé (vocabulaire-compréhension)

On considère l'algorithme suivant, rédigé en langage naturel.

Variables : x, a, b, c réels

Entrée :
Saisir x

Traitement :
a prend la valeur x^2
b prend la valeur $5x$
c prend la valeur $a - b + 2$

Sortie :
Afficher c

Recopier et compléter les phrases suivantes :

x, a, b, c sont les … de l'algorithme.

« Saisir x », « a prend la valeur x^2 », « b prend la valeur $5x$ », « c prend la valeur $a - b + 2$ », « Afficher c » sont les … de l'algorithme.

Lorsque l'on rentre le nombre 3 en entrée, on a successivement :

a prend la valeur … .

b prend la valeur … .

c prend la valeur … .

Conclusion : Lorsque l'utilisateur entre la valeur $x = 3$, l'algorithme affiche en sortie le nombre … .

Corrigé

Cet exercice a pour objectif de comprendre le fonctionnement d'un algorithme simple « à la main ».

Variables : x, a, b, c réels

Entrée :
Saisir x

Traitement :
a prend la valeur x^2
b prend la valeur $5x$
c prend la valeur $a - b + 2$

Sortie :
Afficher c

x, a, b, c sont les **variables** de l'algorithme.

« Saisir x », « a prend la valeur x^2 », « b prend la valeur $5x$ », « c prend la valeur $a - b + 2$ », « Afficher c » sont les **instructions** de l'algorithme.

Lorsque l'on rentre le nombre 3 en entrée, on a successivement :

a prend la valeur **9**.

b prend la valeur **15**.

c prend la valeur **$9 - 15 + 2 = -4$**.

Conclusion : Lorsque l'utilisateur entre la valeur $x = 3$, l'algorithme affiche en sortie le nombre **-4**.

Synthèse :
savoirs et savoirs-faire du chapitre

- Connaître la **définition d'un algorithme** : suite finie ordonnée d'instructions
Un algorithme fait intervenir des variables, en nombre fini.
Un algorithme a un début et une fin.

- Retenir la notion de **variable informatique** (savoir que le mot « variable » pour les algorithmes n'a pas le même sens que pour les fonctions ou pour les équations, quoique le terme de variable ne soit pas le terme correct dans ce cas ; le terme correct pour les équations est celui d'inconnue).

On peut voir une variable informatique comme une boîte ou un récipient qui sert à stocker des valeurs et dont le contenu évolue au cours du déroulement de l'algorithme.

Lorsqu'une variable change de valeur, la valeur précédente est effacée.

La valeur d'une variable qui s'affiche en sortie correspond toujours à la dernière valeur dans le traitement.

Certaines variables apparaissent seulement à l'intérieur de l'algorithme.
Elles n'apparaissent ni en entrée, ni en sortie.
On dit que ce sont des **variables locales** ou **internes**.

Cette année, nous travaillerons surtout avec des variables numériques.

- Retenir l'**architecture générale** d'un algorithme : **entrée(s)**, **traitement**, **sortie**.

- Connaître les **instructions élémentaires** : **lire** (ou saisir), **affecter la valeur**, **afficher**.
On rédige les instructions dans un langage codifié permettant de programmer facilement sur tout type de machine.

- Savoir comment rédiger en langage naturel une **instruction d'affectation** : « x prend la valeur … » ou « affecter à x la valeur … »
On peut affecter à une variable un nombre ou une expression définie à partir d'autres variables.

- Savoir lire et comprendre un algorithme de calcul simple : savoir repérer les variables ; savoir repérer les instructions ; savoir interpréter le résultat obtenu en sortie.

- Savoir faire tourner « à la main » un algorithme simple rédigé en langage naturel ; en particulier, savoir utiliser un **tableau de suivi** des variables de l'algorithme. Nous verrons que cela sera très utile dans des structures algorithmiques plus complexes.

- Savoir rédiger en langage naturel un algorithme simple correspondant à un programme de calcul.

- Parfois, on « déclare » les variables au début de l'algorithme c'est-à-dire que l'on indique leur type.

Déclaration des variables :

- dans un **algorithme** écrit : en général, on le fait toujours au début.

- dans un **programme**, on ne le fait pas toujours ; cela dépend du langage de programmation.

Exercices sur les algorithmes (1)

Les exercices s'articulent autour des compétences suivantes : comprendre, expliquer, interpréter, modifier, écrire un algorithme simple.

Dans les exercices où un algorithme est donné, il est demandé de le recopier pour s'imprégner de la rédaction.

1 On considère l'algorithme suivant rédigé en langage naturel.

Variable : a réel

Entrée :
Saisir a

Traitement :
a prend la valeur $a - 1$
a prend la valeur $2a$

Sortie :
Afficher a

Recopier cet algorithme.

1°) En saisissant 2 pour valeur de a en entrée, quelle valeur de a s'affiche à la fin de l'algorithme ?
2°) Paul affirme que a ne change jamais de valeur. A-t-il raison ? Justifier la réponse.

2 Écrire un algorithme rédigé en langage naturel qui traduit le programme de calcul suivant :

- Choisir deux nombres.
- Calculer le carré du premier.
- Calculer le double du second.
- Faire la somme.
- Afficher le résultat.

On s'inspirera de l'exercice précédent.
On introduira des lettres pour désigner les variables.

Attention : il n'y a pas de calcul à faire. C'est le programme correspondant à l'algorithme qui se chargera de faire les calculs (cf. chapitre suivant).

3 Recopier et compléter l'état des variables dans les tableaux ci-dessous en supposant que l'utilisateur entre 5 comme valeur de x.
Comparer leurs avantages et leurs inconvénients.

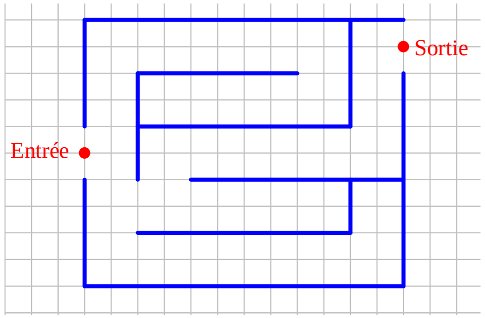
Algorithme 1	Contenus des variables				
Variables : a, b, c, d, x nombres					
Entrée : Saisir x	x :				
Traitement : a prend la valeur $x + 4$	x :	a :			
b prend la valeur $2 \times a$	x :	a :	b :		
c prend la valeur $b - 3$	x :	a :	b :	c :	
d prend la valeur $c - x$	x :	a :	b :	c :	d :
Sortie : Afficher d					

Algorithme 2	Contenus des variables	
Variables : a, x nombres		
Entrée : Saisir x	x :	
Traitement : a prend la valeur $x + 4$	x :	a :
a prend la valeur $2 \times a$	x :	a :
a prend la valeur $a - 3$	x :	a :
a prend la valeur $a - x$	x :	a :
Sortie : Afficher a		

$\boxed{4}$ En utilisant éventuellement plusieurs fois les phrases ci-dessous, décrire le chemin à suivre pour sortir du labyrinthe.

- Avancer de … carreaux.
- Tourner à gauche.
- Tourner à droite.
- C'est l'arrivée.

Reproduire la figure et tracer le trajet pour sortir.



$\boxed{5}$ Les deux algorithmes ci-dessous ont été écrits par Clovis et Darius.

Clovis

Entrée :
Saisir x

Traitement :
a prend la valeur $x + 3$
a prend la valeur a^2
a prend la valeur $a - 1$

Sortie :
Afficher a

Darius

Entrée :
Saisir x

Traitement :
a prend la valeur $x + 6$
a prend la valeur $a \times x$
a prend la valeur $a + 8$

Sortie :
Afficher a

Recopier ces deux algorithmes.

1°) a) Recopier et compléter le tableau suivant.

Entrée	**Sortie**	
	Clovis	**Darius**
2		
– 5		
$\frac{1}{2}$		

b) Que peut-on conjecturer ? Le démontrer.
2°) a) Clovis et Darius ont obtenu les nombres 0, – 1 et 8 en sortie.
Retrouver les nombres x qu'ils ont utilisés.
b) Peuvent-ils obtenir en sortie le nombre – 2 ?

Corrigé

$\boxed{1}$ Fonctionnement d'un algorithme à la main

Objectifs de cet exercice : comprendre la structure d'un algorithme simple. Comprendre la notion de variable dans un algorithme.

Variable : *a* réel

Entrée :
Saisir *a*

Traitement :
a prend la valeur $a - 1$
a prend la valeur $2a$

Sortie :
Afficher *a*

1°) Pour $a = 2$

$2 - 1 = 1$
$2 \times 1 = 2$

À la fin de l'algorithme, **le nombre de sortie est** $\boxed{2}$.

2°) Paul a tort.
En changeant la valeur de *a*, par exemple en prenant 3, on s'aperçoit que la valeur de sortie sera différente de celle prise au départ.

Pour $a = 3$

$3 - 1 = 2$
$2 \times 2 = 4$

À la fin de l'algorithme, **le nombre de sortie est** $\boxed{4}$.
$4 \neq 2$

Remarque :
La valeur de *a* évolue au cours de l'algorithme.
Cet exercice permet de bien comprendre la notion de variable informatique par rapport à la notion de variable en mathématiques (pour les fonctions).

$\boxed{2}$ Rédaction d'un algorithme

Objectifs de cet exercice : passer d'un programme de calcul à un algorithme simple ; rédiger en langage naturel les instructions élémentaires d'un algorithme.

Variables : *a*, *b*, *c*, *d*, *e* réels

Entrées :
Saisir *a*
Saisir *b*

Traitement :
c prend la valeur a^2
d prend la valeur $2b$
e prend la valeur $c + d$

Sortie :
Afficher *e*

Pourquoi a-t-on besoin de plusieurs variables ?
On va répartir le calcul.

c et *d* s'appellent des **variables internes**.

$\boxed{3}$ État et nombre de variables dans un algorithme

Objectifs de cet exercice : suivre l'évolution des variables dans un algorithme ; étudier l'influence du nombre de variables dans un algorithme.

Tableaux d'avancement des deux algorithmes :

Algorithme 1	Contenus des variables				
Variables : *a*, *b*, *c*, *d*, *x* nombres					
Entrée : Saisir *x*	*x* : 5				
Traitement : *a* prend la valeur $x + 4$	*x* : 5	*a* : 9			
b prend la valeur $2 \times a$	*x* : 5	*a* : 9	*b* : 18		
c prend la valeur $b - 3$	*x* : 5	*a* : 9	*b* : 18	*c* : 15	
d prend la valeur $c - x$	*x* : 5	*a* : 9	*b* : 18	*c* : 15	*d* : 10
Sortie : Afficher *d*					

Cet algorithme fait intervenir 5 variables.

Algorithme 2	Contenus des variables	
Variables : *a*, *x* nombres		
Entrée : Saisir *x*	*x* : 5	
Traitement : *a* prend la valeur *x* + 4	*x* : 5	*a* : 9
a prend la valeur 2 × *a*	*x* : 5	*a* : 18
a prend la valeur *a* – 3	*x* : 5	*a* : 15
a prend la valeur *a* – *x*	*x* : 5	*a* : 10
Sortie : Afficher *a*		

Cet algorithme fait intervenir seulement deux variables.

Avantages et inconvénients entre les deux algorithmes :

On s'emmêle moins avec le 1er algorithme qu'avec le 2e.
Réduire le nombre de variables dans un algorithme permet de diminuer la place prise en mémoire par un programme dans une machine. En revanche, l'algorithme peut perdre en lisibilité.
Compte tenu de la puissance des machines à l'heure actuelle, on cherche souvent un compromis entre la réduction du nombre de variables et la conservation d'une bonne lisibilité.

4 Algorithmes sur les labyrinthes

Objectifs de cet exercice : décrire une situation algorithmique avec des mots empruntés au langage courant

L'objectif de cet exercice est de décrire une situation algorithmique sous la forme d'une suite d'instructions à effectuer les unes après les autres.

- Avancer de 1 carreau vers la droite.
- Tourner à droite.
- Avancer de 2 carreaux.
- Tourner à gauche
- Avancer de 2 carreaux.
- Tourner à gauche.
- Avancer de 2 carreaux.
- Tourner à droite.
- Avancer de 8 carreaux.
- Tourner à gauche.
- Avancer de 4 carreaux.
- Tourner à droite.
- Avancer de 1 carreau.
- C'est la sortie.

On se met à la place d'une personne qui serait placée à l'entrée du labyrinthe.

5

1°) a)

Entrée	Sortie	
	Clovis	Darius
2	**24**	**24**
– 5	**3**	**3**
$\frac{1}{2}$	$\frac{45}{4}$	$\frac{45}{4}$

b)

• Formulons une conjecture.

Les résultats de la question précédente permettant de penser que les deux algorithmes donnent le même résultat en sortie lorsque l'on saisit le même nombre en entrée.

• Démontrons cette conjecture.

Soit A le nombre obtenu en sortie avec le l'algorithme de Clovis pour un nombre *x* saisi en entrée.

Soit B le nombre obtenu en sortie avec le l'algorithme de Darius pour un nombre *x* saisi en entrée.

On a : $A=(x+3)^2-1$ et $B=x(x+6)+8$.

Développons chacune de ces expressions.

$$\begin{aligned} A &= (x+3)^2-1 \\ &= x^2+6x+9-1 \\ &= x^2+6x+8 \end{aligned}$$

$$\begin{aligned} B &= x(x+6)+8 \\ &= x^2+6x+8 \end{aligned}$$

On en déduit que A = B. **La conjecture est donc démontrée.**

2°)

a) **Retrouvons les nombres *x* utilisés par Clovis et Darius pour obtenir 0, – 1 et 8 en sortie.**

• Pour le nombre 0 en sortie, on résout dans $\mathbb{R}$ l'équation $A=0$ (1).

(1) est successivement équivalente à :

$(x+3)^2-1=0$

$[(x+3)-1][(x+3)+1]=0$

$(x+2)(x+4)=0$

$x=-2$ ou $x=-4$

On obtient le nombre 0 en sortie si et seulement le nombre d'entrée est **– 2 ou – 4**.

• **Pour le nombre – 1 en sortie,** on résout dans $\mathbb{R}$ l'équation $\text{A} = -1$ (2).

(2) est successivement équivalente à :

$(x+3)^2 - 1 = -1$

$(x+3)^2 = 0$

$x = -3$

On obtient le nombre – 1 en sortie si et seulement le nombre d'entrée est **– 3**.

• **Pour le nombre 8 en sortie,** on résout dans $\mathbb{R}$ l'équation $\text{B} = 8$ (3).

(3) est successivement équivalente à :

$x(x+6) + 8 = 8$

$x(x+6) = 0$

$x = 0$ ou $x = -6$

On obtient le nombre 8 en sortie si et seulement le nombre d'entrée est **0 ou – 6**.

On observera que l'on a à chaque fois choisi l'expression la plus adaptée pour la résolution de l'équation.

b) **Cherchons s'ils peuvent obtenir en sortie le nombre – 2.**

On résout dans $\mathbb{R}$ l'équation $\text{A} = -2$ (4).

(4) est successivement équivalente à :

$(x+3)^2 - 1 = -2$

$(x+3)^2 = -1$

Cette équation n'admet pas de solution dans $\mathbb{R}$ car un carré est toujours positif ou nul.

Ainsi, on en conclut qu'**il n'est pas possible d'obtenir le nombre – 2 en sortie**.