Cours d'algorithmique

Seconde	*Septembre 2009*

Ce cours est réalisé à partir de celui de F. Gaudon (http://mathsfg.net.free.fr/).

Table des matières

1. Avant la programmation

1) Qu'est-ce qu'un algorithme ?

Définition 1 : Un **algorithme** est une succession d'**instructions** (aussi appelées **commandes**) et permettant la résolution d'un problème donné.

Remarque : Le terme d'algorithme vient du nom du mathématicien arabe du IXe siècle Al Khuwarizmi. Ses travaux sur les algorithmes, terme dérivé de son nom, permirent d'introduire la méthode de calcul utilisant les chiffres arabes et la notation décimale (*Encarta*)

Exemple :
```
pour A allant de 1 à 10 par pas de 1
Stocker A^2 dans B
Afficher B
```

L'algorithme précédent calcule et affiche le carré des nombres de 1 à 10. Dans cet algorithme, `Stocker A^2 dans B` est une instruction.

2) Qu'est-ce qu'un langage de programmation ?

Définition 2 : Un **langage de programmation** est un ensemble d'instructions et de règles syntaxiques compréhensible par l'ordinateur et permettant de créer des algorithmes. Un **programme** est la traduction d'un algorithme dans le langage de programmation utilisé.

Exemples : Nous travaillerons par la suite avec les logiciels XCas et AlgoBox, et avec la calculatrice Casio ClassPad.

3) Avant de programmer

a) Créer, modifier ou exécuter un programme

<table>
<tr><th></th><th>AlgoBox</th><th>XCas</th><th>Casio ClassPad</th></tr>
<tr><td>Pour aller dans le mode Programme</td><td>Ouvrir le logiciel</td><td rowspan="4">L'édition d'un programme se fait dans la ligne de commande. Avant de commencer, aller dans le menu Cfg, puis sélectionner configuration du CAS et vérifier que l'onglet PROG STYLE est en mode XCAS comme ce qui suit :
 

.</td><td>Dans le Menu, cliquer sur l'icône 
</td></tr>
<tr><td>Pour créer un nouveau programme</td><td>Cliquer sur le bouton 
</td><td>Cliquer sur l'icône  , ou sélectionner Edit, puis Nouveau fichier</td></tr>
<tr><td>Pour modifier un programme existant</td><td>Cliquer sur le bouton 
</td><td>Cliquer sur l'icône , ou sélectionner Edit, puis Ouvrir fichier</td></tr>
<tr><td>Pour exécuter un programme</td><td>Cliquer sur le bouton 
</td><td>Cliquer sur l'icône  , ou sélectionner Exécuter, puis Exécuter programme</td></tr>
</table>

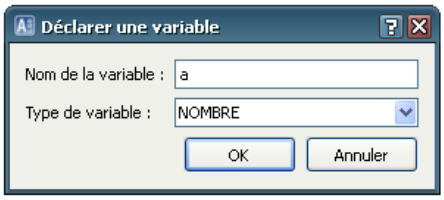
b) Instructions d'un programme

AlgoBox	XCas	Casio ClassPad
Les instructions sont séparées en cliquant sur le bouton 	Les instructions peuvent être séparées par un retour à la ligne SHIFT ENTER . Une ligne peut contenir plusieurs instructions séparées par ; . Attention, les lignes doivent absolument se terminer avec ; .	Les instructions des algorithmes peuvent être séparées par un retour à la ligne EXE . Une ligne peut éventuellement comporter plusieurs instructions séparées par : (ce symbole se trouve en sélectionnant le menu *Ctrl*).

2. Les variables

Définition 3 : On appelle variable tout emplacement de la mémoire de l'ordinateur ou de la calculatrice dans lequel on stocke une information qui peut ^être changée. Une variable est donc constituée :
- d'un nom qui permet de reconnaitre où elle se situe dans la mémoire de l'ordinateur ou de la calculatrice ;
- d'une valeur : le nombre ou plus généralement l'information stockée.

Syntaxe : Pour afficher le « *texte* » suivi de la variable *a* :

AlgoBox	XCas	Casio ClassPad
Avec la souris, se placer sur la ligne VARIABLES, puis cliquer sur le bouton  . Remplir la boîte de dialogue qui s'ouvre de la manière suivante : 	Pour stocker le nombre 3 dans la variable a, on écrira a : = 3	Pour stocker le nombre 3 dans la variable a, on écrira 3 ⇒ a (le symbole ⇒ se trouve en sélectionnant le menu *Ctrl*).

3. Exercices

1) Exercice 1

a) À l'issue de l'algorithme suivant, quel nombre est stocké dans la variable A ? dans la variable B ?

$$3 \to A$$
$$4 \to B$$
$$A \to C$$
$$B \to A$$
$$C \to B$$

b) À quoi sert l'algorithme précédent ?

2) Exercice 2

À l'issue de l'algorithme suivant, quel nombre est stocké dans la variable C ? dans la variable B ? dans la variable A ?

$$5 \to A$$
$$6 \to B$$
$$A + B \to C$$
$$A \times B \to B$$
$$C \to A$$

3) Exercice 3

À l'issue de l'algorithme suivant, quel nombre est stocké dans la variable A ?

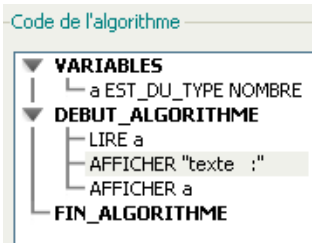
$$5 \to A$$
$$A + 1 \to A$$
$$4 \times A \to A$$

4. Entrées et sorties

1) Commandes d'affichage

Définition 4 : Les commandes d'affichage servent à afficher à l'écran du texte ou la valeur d'une variable.

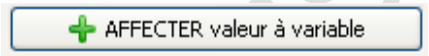
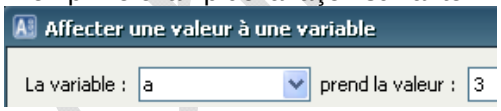
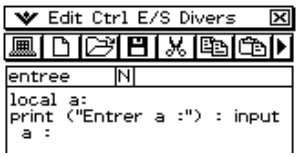
Syntaxe :

AlgoBox	XCas	Casio ClassPad
 Si on souhaite faire afficher la variable à la ligne suivante, il faut cocher le champ « *Ajouter un retour à la ligne* » dans la boîte de dialogue *Afficher un message*.	print ("texte",a);	

2) Commandes d'entrée de valeurs

Définition 5 : Les commandes d'entrée de valeurs permettent à l'algorithme de demander à l'utilisateur un nombre, un caractère ou un texte.

Syntaxe :

AlgoBox	XCas	Casio ClassPad
Avec la souris, se placer sur la ligne DEBUT_ALGORITHME, puis cliquer sur le bouton  , et enfin sur le bouton  . Dans la boîte de dialogue qui s'ouvre, remplir le champ de la façon suivante : 	input("Entrer a :",a);	

5. Exercices sur les entrées et sorties

1) Exercice 4

Que fait l'algorithme suivant ?

Saisir A
Saisir B
A*B → C
2*(A+B) → D
Afficher C
Afficher D

2) Exercice 5

Que fait l'algorithme suivant ?

Saisir D
D/2 → R
3,14*R^2 → A
Afficher A

3) Exercice 6

Écrire un algorithme qui demande d'entrer deux nombres entiers *A* et *B* et calcule le reste de la division euclidienne de *A* et *B*.
On utilisera pour cela la fonction partie entière *int A* qui donne la partie entière d'un nombre *A* (***floor (A)*** avec AlgoBox, ***int(A)*** avec la Casio ClassPad et ***iPart*** avec XCas).

Remarque : Dans le logiciel AlgoBox, l'opérateur « **%** » donne le reste d'une division euclidienne.
Par exemple, 11%3 donne 2.

4) Exercice 7

Écrire un algorithme qui demande d'entrer un nombre puis affiche son image par la fonction *f* définie par $f(x) = 3x^2 + 5x - 9$.

5) Exercice 8

a) Écrire un algorithme qui convertit des secondes en heures, minutes et secondes.
b) Écrire un algorithme qui convertit des heures en jours et heures.

6) Exercice 9

Écrire un algorithme qui demande d'entrer trois nombres *A*, *B* et *C*, et calcule et affiche leur moyenne non pondérée.

7) Exercice 10

Écrire un algorithme qui, l'utilisateur ayant entré le taux annuel d'épargne en pourcentage et le capital initialement placé, calcule et affiche le capital disponible auquel sont ajoutés les intérêts de l'année.

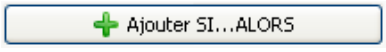
6. Structures conditionnelles

1) Si … alors … sinon

Définition 6 : Ces instructions permettent de tester si une condition est vraie ou fausse, et de poursuivre le programme d'une manière différente selon que la condition est vraie ou fausse.

Syntaxe en algorithmique :

Si
condition
Alors
instructions si la condition est vraie
Sinon
instructions si la condition est fausse
FinSi

	AlgoBox	XCas	Casio ClassPad
Si		if	If
Alors		{instruction ;instruction ;}	Then
Sinon	Dans la boîte de dialogue qui s'est ouverte, cocher le champ « *Ajouter SINON* » Puis sélectionner la ligne suivant celle nommée DEBUT_SINON , et rentrer l'instruction souhaitée.	else	Else
FinSi		;	IfEnd

Exemple : Écrire un algorithme qui demande à l'utilisateur d'entrer deux nombres A et B et qui affiche le plus grand des deux.

AlgoBox	XCas	Casio ClassPad

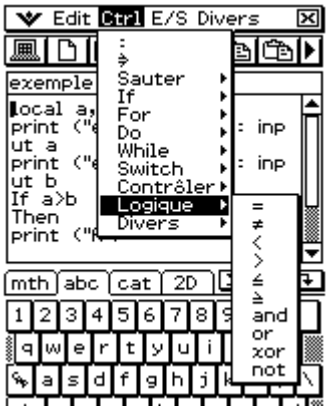
VARIABLES
  A EST_DU_TYPE NOMBRE
  B EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  LIRE A
  LIRE B
  SI (A>B) ALORS
  DEBUT_SI
  AFFICHER "A"
  FIN_SI
  SINON
  DEBUT_SINON
  AFFICHER "B"
  FIN_SINON
FIN_ALGORITHME
	input(" A = ? ",a);
	input(" B = ? ",b);
	if (a>b)
	{print("A",a);}
	else
	{print("B",b);};
		local a,b
		print ("entrer A :") : input a
		print ("entrer B :") : input b
		If a>b
		Then
		print ("A")
		Else
		print ("B")
		IfEnd

2) Opérateurs relationnels et logiques

Définitions 7 : Pour tester une condition on utilise les *opérateurs relationnels* suivants :
- $a = b$ teste l'égalité de a et de b ;
- $a < b$ teste si a est strictement inférieur à b ;
- $a \leq b$ teste si a est inférieur ou égal à b ;
- $a > b$ teste si a est strictement supérieur à b ;
- $a \geq b$ teste si a est supérieur ou égal à b ;
- $a \neq b$ teste si a est différent de b.

On utiliser aussi pour les conditions plus complexes les opérateurs logiques « et » (AND), « ou » (OR) et « non » (NOT).

Où les trouver ? :

<table>
<tr><th>AlgoBox</th><th>XCas</th><th>Casio ClassPad</th></tr>
<tr><td colspan="2">• Pour vérifier si x est égal à 2, la condition à écrire est :
x==2
• Pour vérifier si x est différent de 2, la condition à écrire est :
x!=2
• Pour vérifier si x est strictement inférieur à 2, la condition à écrire est :
x<2
• Pour vérifier si x est inférieur ou égal à 2, la condition à écrire est
x<=2
• Pour vérifier si x est strictement supérieur à 2, la condition à écrire est :
x>2
• Pour vérifier si x est supérieur ou égal à 2, la condition à écrire est :
x>=2</td><td rowspan="2">Les commandes se trouvent dans le menu Ctrl, puis Logique :
 
</td></tr>
<tr><td>• La condition à écrire pour vérifier que x est strictement compris entre 1 et 5 est :
x>1 ET x<5
• La condition à écrire pour vérifier que x est égal à 3 OU à 5 est :
x==3 OU x==5</td><td>• condition1 && condition2 teste si les deux conditions sont vraies simultanément
• condition1 || condition2 teste si l'une au moins des deux conditions est vraie
• ! condition teste si la négation de la condition est vraie</td></tr>
</table>

7. Exercices sur les structures conditionnelles

1) Exercice 11

Concevoir un algorithme correspondant au problème suivant :

- on demande à l'utilisateur d'entrer un nombre (qui sera représenté par la variable *a*)
- si le nombre entré est différent de 1, l'algorithme doit stocker dans une variable *b* la valeur de $\frac{3}{x-2}$ et afficher la valeur de *b*. On ne demande pas de traiter le cas contraire

2) Exercice 12

Écrire un programme qui demande l'âge de l'utilisateur et répond "vous êtes mineur" ou "vous êtes majeur" suivant le cas.

3) Exercice 13

Écrire un programme qui demande la température extérieure en degrés Celsius et affiche "il gèle" si le nombre est négatif et "alerte à la canicule" si le nombre est supérieur à 30.

4) Exercice 14

a) Qu'affiche l'algorithme suivant ?

```
1000 → tirelire
19 → âge
Si (âge > = 19 et tirelire > = 1000)
alors afficher « Vous pouvez ouvrir un compte »
sinon afficher « ouverture de compte impossible »
```

b) Écrire le code correspondant à l'algorithme précédent pour la calculatrice, pour XCas et pour AlgoBox.

5) Exercice 15

Écrire un algorithme qui, à partir d'un nombre entré par l'utilisateur, affiche ce même nombre s'il est positif et son opposé s'il est négatif (le nombre obtenu est appelé la valeur absolue du nombre entré).

6) Exercice 16

Écrire un algorithme qui demande d'entrer les coordonnées de quatre points *A*, *B*, *C* et *D*, et affiche si le quadrilatère *ABCD* est un parallélogramme ou non.

7) Exercice 17

Écrire un algorithme qui, à partir de la donnée de la longueur de chacun des trois côtés d'un triangle, teste si le triangle est rectangle.

8. Les boucles

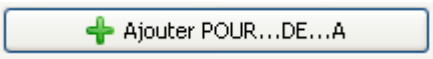
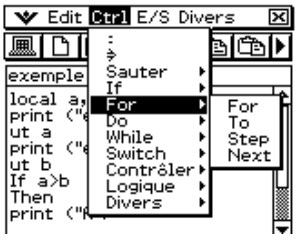
Définition 8 : Les boucles sont utilisées pour qu'une séquence d'instructions soit répétée un nombre donné de fois ou tant qu'une condition n'est pas remplie.

1) Boucle POUR

Définition 9 : Ces instructions sont utilisées pour contrôler les boucles en incrémentant (augmentant) une variable. La variable est augmentée d'une valeur de départ jusqu'à une valeur d'arrivée d'un pas donné (l'incrément).

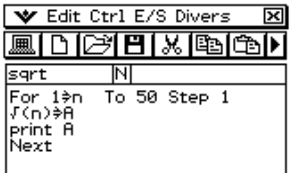
Syntaxe en algorithmique :

Pour
variable
allant de
valeur de départ
à
valeur d'arrivée
faire
instructions
fin

AlgoBox	XCas	Casio ClassPad
Cliquer sur le bouton  et remplir la boîte de dialogue. Puis donner les instructions. *Remarques* : *- La variable servant de compteur pour la boucle doit être du type NOMBRE et doit être déclarée préalablement (comme toutes les variables).* *- Cette variable est automatiquement augmentée de 1 à chaque fois.* *- On peut utiliser la valeur du compteur pour faire des calculs à l'intérieur de la boucle, mais les instructions comprises entre **DEBUT_POUR** et **FIN_POUR** ne doivent en aucun cas modifier la valeur de la variable qui sert de compteur.*	**For (**variable **:=** valeur de départ **;** variable **<=** valeur finale **;** variable **:=** variable **+** incrément**)** **{**instruction **;** instruction **;** … instruction **;} ;**	Les commandes se trouvent dans le menu *Ctrl*, puis *For* :  **For** valeur de départ → variable **To** valeur d'arrivée **Step** incrément Instructions **Next**

Exemple : Écrire un algorithme qui affiche la racine carrée de tous les entiers de 1 jusqu'à 50.

Pour
n
allant de
1
à
50
faire
calculer la racine carrée de *n* et la faire afficher
fin

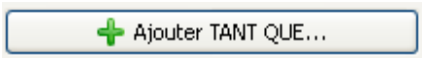
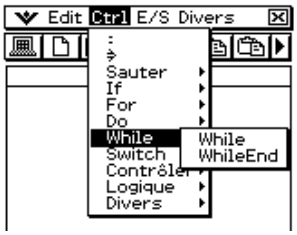
AlgoBox	XCas	Casio ClassPad
VARIABLES n EST_DU_TYPE NOMBRE racine_carrée EST_DU_TYPE NOMBRE DEBUT_ALGORITHME POUR n ALLANT_DE 1 A 50 DEBUT_POUR racine_carrée PREND_LA_VALEUR sqrt(n) AFFICHER racine_carrée FIN_POUR FIN_ALGORITHME	for (n:=0; n<=10;n:=n+1) {a:=sqrt(n); print(n,a);}	

2) Boucle TANT QUE

Définition 10 : Exécute un groupe de commandes tant qu'une condition est vraie. La condition est testée en début de boucle.

Syntaxe en algorithmique :

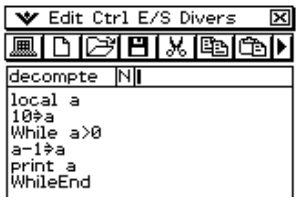
Tant que condition
instructions
faire
instructions
fin tant que

AlgoBox	XCas	Casio ClassPad
Cliquer sur le bouton  et remplir la boîte de dialogue. Puis donner les instructions.	**While** (condition) {instruction ; instruction ; … instruction ;} ;	Les commandes se trouvent dans le menu *Ctrl*, puis *While* :  **While** condition instructions **WhileEnd**

Exemple : Écrire un algorithme qui le décompte de 9 à 0.

10 → a
Tant que a > 0 faire
a - 1 → a
Afficher a
Fin tant que

AlgoBox	XCas	Casio ClassPad

VARIABLES
a EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
a PREND_LA_VALEUR 10
TANT_QUE (a>0) FAIRE
DEBUT_TANT_QUE
a PREND_LA_VALEUR a-1
AFFICHER a
FIN_TANT_QUE
FIN_ALGORITHME
	a:=10;
	while (a>0)
	{a:=a-1;
	print(a);};
		

9. Exercices sur boucles

1) Exercice 18

a) Écrire un algorithme qui calcule la somme des nombres entiers de 0 _a 50.
b) Écrire un algorithme qui calcule le produit des nombres entiers de 1 _a 7
c) Écrire un algorithme qui calcule la somme des 20 premiers nombres impairs.
d) Écrire un algorithme qui calcule la somme des 20 premiers nombres pairs.

2) Exercice 19

Écrire un algorithme (algorithme d'Euclide) permettant de calculer le PGCD de deux nombres entiers A et B entrés.

3) Exercice 20

Écrire un algorithme qui calcule la variance et l'écart type d'une série de nombres entrés par l'utilisateur. L'algorithme demandera le nombre de nombres que comprend la série avant de demander d'entrer la série de nombres.

4) Exercice 21

Écrire un algorithme qui, une somme initiale ayant été demandée à l'utilisateur ainsi qu'une durée de placement en année et un taux de placement en pourcentage à intérêts composés, affiche la somme disponible au bout de la durée de placement.

5) Exercice 22

Un individu a emprunté à un ami une somme de 2500 euros (prêt sans intérêts).
Pour rembourser son ami, il prévoit de lui remettre 110 euros par mois. Mais comme cela ne correspond pas à un nombre pile de mois, il se demande quel sera le montant à rembourser le dernier mois.
Écrire un algorithme permettant de résoudre le problème posé.

6) Exercice 23

On cherche à connaître le plus petit entier N tel que 2^N soit supérieur ou égal à 10000.
Pour résoudre ce problème de façon algorithmique, l'idée est de calculer les puissances consécutives de 2 jusqu'à ce qu'on atteigne 10000 en utilisant une boucle TANT QUE.

Remarque : La fonction permettant de calculer 2^N avec AlgoBox est : pow(2,N).

7) Exercice 24

On considère le problème suivant :
- On lance une balle d'une hauteur initiale de 300 cm.
- On suppose qu'à chaque rebond, la balle perd 10 % de sa hauteur (la hauteur est donc multipliée par 0,9 à chaque rebond).
- On cherche à savoir le nombre de rebonds nécessaire pour que la hauteur de la balle soit inférieure ou égale à 10 cm.

Écrire un algorithme permettant de résoudre ce problème.