Algorithmes et programmation

Seconde

L'équipe des professeurs de mathématiques
Lycée Stendhal

"Le génie consiste à voir ce que tout le monde a vu et à penser ce que personne n'a pensé."

Albert Einstein

Année 2016-2017

Liste des savoirs et savoir-faire du chapitre :

CODE	INTITULE	Bilan		
		A	EA	NA
Ch0301	Définir et utiliser des variables.			
Ch0302	Ecrire un algorithme avec une instruction conditionnelle.			
Ch0303	Ecrire un algorithme avec une boucle pour.			
Ch0304	Ecrire un algorithme avec une boucle tant que.			
Ch0305	Ecrire l'algorithme d'une fonction.			
Ch0306	Décrire et interpréter un algorithme.			
Ch0307	Ecrire des programmes en langage Ti.			
Ch0308	Ecrire des programmes en langage Pythone.			

Compétences dans tous les chapitres :

INTITULE	Bilan		
	A	EA	NA
Chercher			
Modéliser			
Représenter			
Calculer			
Raisonner			
Communiquer			

Table des matières

Chapitre

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 abordera seulement les algorithmes Python et celui des calculatrices TI.

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 :

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

On peut aussi le présenter sous forme d'une fonction :

Fonction1 :	
Déclaration des variables global	
	x : un nombre réel.
Déclaration des variables local :	
	Result : Un nombre réel
Initialisation :	
Traitement :	
	$4-3(x-2)^2 \mapsto$ Result
Sortie :	
	Retourner la valeur de Result.

2 Les variables

En seconde on peut être amené à utiliser plusieurs types de variables :

1. Les nombres entiers (integer).
2. Les nombres à virgule (float).
3. Les variables boolean (true ou false).
4. Les variables sous forme de Chaine de caractères (Char) où chacun des caractère est appelé par l'indice de son emplacement dans la chaine.
   Par exemple, si la chaine se nomme Ch alors le troisième caractère de la chaine est donné par $Ch[3]$.
5. Les variables sous forme de tableaux où chacune des cases est appelée par l'indice de la ligne et de la colonne.
   Par exemple, si Tab est un tableau, la case qui est à l'intersection entre la ligne 2 et la colonne 3 est donné par $Tab[2,3]$.

3 Tests et boucles

3.1 Les tests ou instructions conditionnelles "si"

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".

3.2 Boucle "Pour"

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 100	Faire
	$S + I \mapsto S$
Fin du Pour	
Sortie :	
	Afficher la valeur de S.

3.3 Boucle "Tant que"

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 régulièrement 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 $I \leq 100$	Faire
	$S + I \mapsto S$
	$I + 1 \mapsto I$
Fin du Tant que	
Sortie :	
	Afficher la valeur de S.

4 Les fonctions

Les fonctions sont des blocs de programme qui renvoie un résultat et que l'on peut utiliser dans la suite ou les autres blocs de programme.
Elles se présentent sous la forme :

NomDeLaFonction :	
(Déclaration des variables global	
	Les variables externes.)
Déclaration des variables local :	
	Les variables internes à la fonction
Initialisation :	
Traitement :	
	Instructions donnant le Resultat
Sortie :	
	Retourner la valeur du résultat.

Exemple : On souhaite avoir une fonction qui nous donne la valeur de la somme des n premiers entiers naturels.

SommeEntiers :	
(Déclaration des variables global :	
	n : un nombre entier naturel.)
Déclaration des variables local :	
	k : Un nombre entier naturel.
Initialisation :	
$0 \mapsto$ SommeEntiers	
Traitement :	
Pour k allant de 1 à n	
	SommeEntiers$+k \mapsto$ SommeEntiers
Fin du Pour	
Sortie :	
	Retourner la valeur de SommeEntiers.

Dans l'exemple précédent, si on tape **SommeEntiers(10)** la fonction va renvoyer la valeur 55 car

$$1+2+3+4+5+6+7+8+9+10=55$$

5 Quelques langages de programmation

Fiche Outils : Programmer (Ti82 et plus)

Significations	Algorithme	Commande Ti	Où est-ce ?
Saisir une valeur après demande(ou sans) et affecte à X cette valeur	Saisir X Ou Lire la valeur de X	**: input " X= " ,X**	Appuyer sur la touche **Prgm** puis aller dans le menu **E/S** ou **I/O** Pour " appuyer sur **Alpha** puis **+**
Afficher à l'écran la valeur de la variable X ou affiche X= puis la valeur de la variable X	Afficher la valeur de X	**: Disp X** Ou **:Disp " X= ",X**	Appuyer sur la touche **Prgm** puis aller dans le menu **E/S** ou **I/O**
La valeur a^2+1 va se stocker dans la mémoire de la variable X	X reçoit la valeur de a^2+1	**:a^2+1 → X**	Touche **STO**
Si les conditions sont vraies alors on exécute instructions1 sinon on exécute instructions2	Si … Alors … Sinon …	**:If** conditions **:Then** : instructions1 **:Else** :instructions2 **:End**	Appuyer sur la touche **Prgm** puis aller dans le menu **CTL**
On veut faire n fois les mêmes instructions	Pour i allant de 1 à n Faire …	**:For(I,1,N)** :instructions **:End**	Appuyer sur la touche **Prgm** puis aller dans le menu **CTL**
Tant que des conditions sont vraies, répéter les instructions	Tant que … Faire …	**:While** conditions :Instuctions **:End**	Appuyer sur la touche **Prgm** puis aller dans le menu **CTL**
Un nombre entier aléatoire entre a et b est stocker dans la variable X	X reçoit un nombre entier aléatoire entre a et b	:**entAléat**(a,b)→X Ou :**randInt**(a,b)→X	Touche **Maths** puis menu **PRB**
Stocker une liste de n nombres dans une liste L_1	Pour i allant de 1 à n Stocker les nombres dans les cellules de L	:For(I,1,n) :Input "X=",X :X→**L_1(I)** :End	Touche **2nde** puis **1**
Stocker une liste de n nombres entiers aléatoires dans une liste L_1	Pour i allant de 1 à n Stocker les nombres dans les cellules de L	:For(I,1,n) :entAléat(a,b)→**L_1(I)** :End	Touche **2nde** puis **1**
Faire une pause dans l'affichage des résultats	Pause	**:Pause**	Appuyer sur la touche **Prgm** puis aller dans le menu **CTL**

Pour insérer une ligne, aller à la fin de la ligne précédente puis appuyer sur **2nde** puis **Del/Suppr** puis sur **Enter**.
Pour insérer un caractère, aller là où vous voulez l'insérer puis appuyer sur **2nde** puis **Del/Suppr**.
Ne pas hésiter à aller voir dans tous les menus de **Prgm** et de **Maths** et aussi **Stats** pour découvrir toutes les possibilités que vous offre votre calculatrice.
Pratiquement toutes les commandes se trouvent dans le catalogue **2nde 0.**

Les lettres

La touche **Alpha** permet de taper des lettres ainsi que le symbole " pour les instructions input et disp

Les symboles

La touche **Tests** (Touche **2nde** puis **Maths**) permet d'accéder aux symboles de comparaison
=, ≠, ≤ puis ≥

Effacer un programme

La touche **Meme** (Touche **2nde** puis **+**) puis le menu **2 :Efface** puis le menu **7 :Prgm** permet d'effacer des programmes dont on ne se sert plus.

Créer un programme

Touche **Prgm** puis **Nouv** puis touche **Enter**. On tape le nom du programme puis touche **Enter** et ensuite on peut taper le code du programme

Modifier ou terminer un programme

Touche **Prgm** puis menu **Edit** puis touche **Enter**. On choisit le programme à modifier dans la liste puis touche **Enter**.

Exécuter un programme existant

Touche **Prgm** puis menu **Exec** puis touche **Enter**. On choisit le programme à exécuter dans la liste puis touche **Enter**.

Fiche Outils : Programmer (Casio Grpah35+ et plus)

Significations	Algorithme	Commande Ti	Où est-ce ?
Saisir une valeur après demande(ou sans) et affecte à X cette valeur	Saisir X Ou Lire la valeur de X	**"X=" :?→X**	Le ? se trouve dans **F4**
Afficher à l'écran la valeur de la variable X ou affiche X= puis la valeur de la variable X	Afficher la valeur de X	**"X=" :X**	Les lettres, ", et → sont dans le format alphanumérique.
La valeur a^2+1 va se stocker dans la mémoire de la variable X	X reçoit la valeur de a^2+1	**a ^ 2+1 → X**	Les lettres et " sont dans le format alphanumérique et → à l'aide de la touche STO
Si les conditions sont vraies alors on exécute instructions1 sinon on exécute instructions2	Si … Alors … Sinon …	**If** conditions **Then** Instruction1(s) **Else** Instruction2(s) **IfEnd**	L'accès aux commandes **Prgm** se fait à l'aide des touches **Shift Vars**.
On veut faire n fois les mêmes instructions	Pour i allant de 1 à n Faire …	**For 1→I to n** instructions **Next**	L'accès aux commandes **Prgm** se fait à l'aide des touches **Shift Vars**.
Tant que des conditions sont vraies, répéter les instructions	Tant que … Faire …	**While** conditions Instuction(s) **WhileEnd**	L'accès aux commandes **Prgm** se fait à l'aide des touches **Shift Vars**.
Un nombre entier aléatoire entre a et b est stocker dans la variable X	X reçoit un nombre entier aléatoire entre a et b	**RanInt#(a,b)→X**	Appuyer sur **OPTN** puis **F6** puis **PROB** puis **RAND** puis **Int**
Stocker une liste de n nombres dans une liste L_1	Pour i allant de 1 à n Stocker les nombres dans les cellules de L	For 1→I to n "X="?→X X→**List 1[I]** Next	On obtient **List 1** par les touches **SHIFT 1**
Stocker une liste de n nombres entiers aléatoires dans une liste L_1	Pour i allant de 1 à n Stocker les nombres dans les cellules de L	For 1→I to n RanInt#(a,b)→**List 1[I]** Next	

Quelques commandes utiles

Partie entière (Intg(x)) : OPTN + F6 + NUM + Intg
Partie décimale (Frac(x)) : OPTN + F6+ NUM + Frac
Reste de la division (Mod(A,B)) : OPTN + F6 + NUM + F6 + MOD

Pratiquement toutes les commandes se trouvent dans le catalogue **SHIFT 4.**

Ne pas hésiter à aller voir dans tous les menus découvrir toutes les possibilités que vous offre votre calculatrice.
Pour quitter l'éditeur de programme touche : **EXIT**

Les lettres

La touche **Alpha** permet de taper des lettres .

Effacer un programme

Menu **Prgm** puis **EXE**, sélectionner le programme puis menu **F4 Del** puis touche **F1 Yes**.

Accéder et Créer un programme

Pour accéder aux programmes choisir **Prgm** puis **EXE**, dans le menu principal.
Pour créer un nouveau programme choisir **New(F3)**. On tape le nom du programme puis touche **Exe** et ensuite on peut taper le code du programme

Modifier ou terminer un programme

Menu **Prgm**, sélectionner le programme puis menu **Edit** puis touche **Exe**.

Exécuter un programme existant

Menu **Prgm**, sélectionner le programme puis touche **Exe**.

Fiche Outils : Programmer (Python)

Significations	Algorithme	Commande Ti
Saisir une valeur après demande(ou sans) et affecte à X cette valeur	Saisir X Ou Lire la valeur de X	**X=input() ;**
Afficher à l'écran la valeur de la variable X ou affiche X= puis la valeur de la variable X	Afficher la valeur de X	**print("X=",X) ;**
La valeur a^2+1 va se stocker dans la mémoire de la variable X	X reçoit la valeur de a^2+1	**X=a*a+1 ;**
Si les conditions sont vraies alors on exécute instructions1 sinon on exécute instructions2	Si ... Alors ... Sinon ...	**if** conditions : instructions1 ; **else :** instructions2 ;
On veut faire n fois les mêmes instructions	Pour i allant de 1 à n Faire ...	**for i in range (1,n+1) :** instructions;
Tant que des conditions sont vraies, répéter les instructions	Tant que ... Faire ...	**while** conditions : Instuctions ;
Ecrire une fonction	**Def NomFonction(**paramètres**) :** Instructions ; **Return** ValeurReponse	

Quelques instructions importantes Ajouter **import math** **import random** au début de votre programme.	
math.pi	Valeur exacte de pi
math.pow(x,n)	x puissance n
a%b	Reste de la division de a par b
math.sqrt(x)	Racine carré de x
math.exp	Exponentielle de x
math.fabs	Valeur absolue de x
math.e	Valeur exacte de e
math.ceil(x)	Plus petit entier >= x
math.floor(x)	Plus grand entier <= x
Random.randint(a,b)	Choisir au hasard un nombre entre a et b
random.uniform(0,1)	Choiisir au hasard un nombre entre 0 et 1
X==y	Teste si x est égal à y
X !=y	Teste si X est différent de y
X+=1	Incrémenter de 1 la valeur de x
a>=2 and a<=8	Teste si a est dans [2 ;8]
a<2 or a>8	Teste si a<2 ou a>8

Exemple de fonction factorielle en python :

De façon classique

```
def fact(n):
  if n>0 :
    result=1 ;
    for i in range (1,n+1) :
result=result*i ;
    return result ;
  else:
      return 1 ;
```

De façon récursive

```
def fact(n):
  if n>0 :
    return n*fact(n-1);
  else:
      return 1
```

6 Exemples d'algorithmes et de fonctions

6.1 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.

6.2 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.

6.3 Fonction factorielle

Cette fonction prend en entrée un entier naturel (n) et renvoie le produit

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

Factorielle :	
(Déclaration des variables global	
	n ; Entier naturel.)
Déclaration des variables local :	
	k, $Result$: entiers naturels
Initialisation :	
Traitement :	
Si $n > 0$	
	$1 \mapsto Result$
	Pour k allant de 1 à n faire
	$Result \times k \mapsto Result$
Sinon	
	$1 \mapsto Result$
Sortie :	
	Retourner $Result$.

Exemple : **Factorielle(5)** renvoie la valeur 120 car

$$1 \times 2 \times 3 \times 4 \times 5 = 120$$

6.4 Fonction diviseur

Cette fonction prend en entrée deux entiers naturels a et b (avec $a \leq b$) et renvoie "true" si b divise a et "false" sinon.

Diviseur :
(Déclaration des variables global
a, b ; Entiers naturels.)
Déclaration des variables local :
$Result$: Variable boolean
Initialisation :

Traitement :
Si le reste de la division de a par b est égal à 0
True $\mapsto Result$
Sinon
False $\mapsto Result$
Sortie :
Retourner $Result$.

Exemples :

Diviseur(15,3) renvoie **True**.

Diviseur(15,4) renvoie **False**.

6.5 Exercices

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

Exercice 1

Construire un algorithme qui détermine à partir de quelle valeur de n, la somme S est supérieure ou égale à 10 000.

Exercice 2

Construire un algorithme qui détermine à partir de quelle valeur de n, la somme S est supérieure ou égale à 10 000.

Exercice 3

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

Exercice 4

Construire un algorithme qui détermine à partir de quelle valeur de n le produit P est supérieur ou égal à 10 000.