

PRÉPA SCIENTIFIQUE

1^{re} et 2^e années



Thomas **Petit**

TOUS LES ALGORITHMES

PROGRAMMATION

PYTHON



Thomas **Petit**



TOUS LES ALGORITHMES

PROGRAMMATION

PYTHON

Prépa scientifique 1^{re} et 2^e années



Retrouvez toutes les publications de Thomas Petit
sur le site des éditions Ellipses :

www.editions-ellipses.fr



ISBN 9782340052208
© Ellipses Édition Marketing S.A., 2018
32, rue Bargue 75740 Paris cedex 15



Le Code de la propriété intellectuelle n'autorisant, aux termes de l'article L. 122-5.2° et 3°a), d'une part, que les « copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective », et d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, « toute représentation ou reproduction intégrale ou partielle faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause est illicite » (art. L. 122-4).

Cette représentation ou reproduction, par quelque procédé que ce soit constituerait une contrefaçon sanctionnée par les articles L. 335-2 et suivants du Code de la propriété intellectuelle.

www.editions-ellipses.fr

Avant-propos



Les scientifiques, jusqu'à récemment, se reconnaissaient à leurs instruments : l'astronome à sa lunette, le chimiste à sa cornue... et le mathématicien à sa blouse blanche et sa craie. Aujourd'hui tous utilisent le même outil : l'ordinateur.

Michel Serres (1930-)

Pourquoi ce livre ?

En prépa, un bon moyen de préparer l'épreuve informatique est de bien connaître des algorithmes classiques. N'importe quel sujet pouvant tomber, ce n'est pas forcément l'aboutissement du projet que l'examineur recherche mais de voir si le candidat maîtrise les principes de bases de Python : programmation d'une fonction, maîtrise de la récursivité, manipulation des listes. Les mathématiques nous offrent l'opportunité de maîtriser ces principes et de voir des algorithmes puissants, la plupart historiques, permettant d'obtenir des approximations (de solutions d'équations, de constantes célèbres, de calculs d'intégrales, etc.) ou permettant la gestion de listes (tris, recherche d'éléments, etc.).

Comment se procurer Python ?

Sur Internet, en téléchargement gratuit aux adresses suivantes :

<http://www.python.org/> ou www.continuum.io/downloads (pour la version Anaconda et Spyder, qui dispose de bonnes bibliothèques graphiques).

Comment est construit ce livre ?

De manière à programmer tout de suite, sans avoir à se taper 30 pages d'explications théoriques, de syntaxe épouvantable... On s'est mis à la place d'un lecteur qui veut voir un algorithme fonctionner immédiatement et effectuer la tâche précise qui lui est dédiée. La mise en page choisie (un algo par page, en gros) vous permettra de repérer ou de retrouver très vite l'algorithme que vous cherchez !

Tous les algorithmes sont écrits sous Python Idle sauf quelques-uns (immédiatement identifiables car possédant des numéros de lignes) sous Anaconda Spyder, pour des raisons de bibliothèque graphique ou d'outils matriciels.

Note : les algorithmes signalés par un astérisque (*) sont ceux mentionnés explicitement dans le programmes de prépa (donc exigibles).

Que faire si un algorithme ne marche pas ?

Surtout ne pas s'énerver. Il y a de fortes chances que la cause soit une faute de frappe ou même un oubli (une parenthèse, une majuscule, une virgule au lieu d'un point, etc.), mais où ? Le logiciel ne nous donne pas hélas beaucoup d'indications sur l'endroit où se trouve l'erreur (on a l'impression qu'il s'en fiche royalement). Ne désespérez pas trop non plus : placer des « print » à des endroits stratégiques (pour repérer où l'ordinateur a changé une valeur qu'il n'aurait pas dû) peut être bénéfique. Sinon, la bonne vieille aide Internet (avec forums ou tutos pour la plupart en anglais...) peut s'avérer salvatrice voire miraculeuse (à condition d'être patient, voire bilingue...) car l'erreur que vous avez commise a probablement été commise par quelqu'un d'autre, avant vous, et qui a la bonne idée d'en témoigner...

Les erreurs en programmation restent souvent une occasion de progresser et provoquent une joie immense lorsqu'on les élimine. S'il y a une erreur, prenez un bon bol d'air avant toute crispation ! C'est à force de recopier et de lire des lignes de programme, qu'on se forge une culture de la programmation et qu'on devient soi-même un bon programmeur. Courage !

Sommaire



Chapitre 1.

Tous les algorithmes de 1^{re} année9

Résolution d'une équation par dichotomie*	10
Résolution d'une équation par la méthode de la fausse position	11
Résolution d'une équation par la méthode de la sécante	12
Résolution d'une équation par la méthode de Newton*	13
Calcul (approché) d'une intégrale par la méthode des rectangles*	14
Calcul (approché) d'une intégrale par la méthode des trapèzes*	15
Calcul (approché) d'une intégrale par la méthode de Simpson	16
Approximation de e par le calcul intégral et la dichotomie	17
Résolution d'une équation différentielle (cas simple) par la méthode d'Euler*	18
Résolution d'une équation différentielle (cas général) par la méthode d'Euler*	19
Algorithme qui donne la liste des diviseurs d'un entier	20
Algorithme qui teste si un entier est premier	21
Algorithme qui donne la liste des diviseurs premiers d'un entier	22
Indicatrice d'Euler	23
Algorithme qui détermine la liste des nombres parfaits	24
Algorithme qui détermine les couples de nombres amicaux	25
Algorithme qui détermine les nombres intouchables	26
Algorithme qui donne la liste des nombres premiers	27
Algorithme qui donne la liste des nombres premiers jumeaux	28
Algorithme qui donne la liste des nombres premiers de Sophie Germain	29
Algorithme de décomposition d'un entier en produit de facteurs premiers	30
Algorithme de la division euclidienne	31
Algorithme qui détermine le reste modulo n d'un entier	32
Extracteur de chiffres d'un entier n	33
Algorithme qui détermine les nombres symétriques	34
Algorithme qui détermine les nombres premiers « reimerp »	35

Détermination du PGCD par l'algorithme d'Euclide.....	36
Algorithme qui détermine une relation de Bézout.....	37
Evaluation d'un polynôme en un point.....	38
Evaluation d'un polynôme en un point par la méthode de Hörner.....	39
Multiplication de deux polynômes.....	40
Algorithme qui détermine les racines d'un polynôme de degré 2.....	41
Algorithme qui détermine les racines d'un polynôme de degré 3 (Cardan).....	42
Polynôme interpolateur de Lagrange (degré 1) passant par deux points.....	43
Polynôme interpolateur de Lagrange (degré 2) passant par trois points.....	44
Calcul de la trace d'une matrice.....	45
Calcul de la transposée d'une matrice.....	46
Calcul de la puissance d'une matrice.....	47
Calcul d'un déterminant d'ordre 2.....	48
Calcul d'un déterminant d'ordre 3.....	49
Résolution d'un système linéaire d'ordre 2 par les formules de Cramer.....	50
Résolution d'un système linéaire d'ordre 3 par les formules de Cramer.....	51
Résolution d'un système linéaire par la méthode du pivot de Gauss*.....	52
Résolution d'un système linéaire par la méthode itérative de Jacobi.....	53
Recherche d'un élément dans une liste*.....	54
Recherche du maximum d'une liste*.....	55
Recherche du minimum d'une liste*.....	56
Calcul de la moyenne et de la variance d'une liste*.....	57
Calcul de l'écart type d'une liste*.....	58
Recherche par dichotomie d'un élément dans une liste triée*.....	59
Recherche d'un mot dans un texte (ou chaîne de caractères) *.....	60
Droite de régression linéaire.....	61
Tracé d'une courbe paramétrée.....	62
Longueur d'une courbe paramétrée.....	63
Génération de points aléatoires sur le cercle unité.....	64
Théorème de Cesàro.....	65

Chapitre 2.

Tous les algorithmes de 2^e année67

Fonction push (Mettre dessus) *	68
Fonction pop (Retirer du dessus) *	69
Echanger deux termes d'une liste	70
Conversion décimal-binaire	71
Conversion binaire-décimal	72
Puissance x^n	73
Somme de puissances $\sum_{k=0}^n x^k$	74
Suite récurrente linéaire d'ordre 1	75
Suite récurrente linéaire d'ordre 2	76
Dichotomie (récursive)	77
Approximation de \sqrt{a} par l'algorithme de Babylone	78
Approximation de $\sqrt{2}$ par les fractions de Bhaskara-Brouckner	79
Approximation de $\sqrt{2}$ par Bombelli	80
Approximation du nombre d'or $\varphi = \frac{1 + \sqrt{5}}{2}$	81
Approximation de la constante d'Euler γ	82
Factorielle $n!$	83
Approximation de e^x	84
Approximation de $\cos(x)$ et $\sin(x)$	85
Coefficient binomial $\binom{n}{p} = \frac{n!}{p!(n-p)!}$	86
Triangle de Pascal	87
Fonction zêta de Riemann	88
Approximation de π par Archimède	89
Approximation de π par Cues	90
Approximation de π par Viète	91

Approximation de π par Al-Kashi	92
Approximation de π par Machin	93
Approximation de π par Ramanujan	94
Approximation de π par Gauss-Legendre-Brent-Salamin	95
Approximation de π par des fractions	96
Tours de Hanoï.....	97
Flocon de Von Koch	98
Anti-flocon de Von Koch	99
Triangle de Sierpinski	100
Carré de Sierpinski	101
Nombre de partitions d'un entier	102
Tri à bulles (Bubblesort)	103
Tri par sélection (Selection Sort)	104
Tri par insertion (Insertion sort) *	105
Tri rapide (Quicksort) *	106
Tri par fusion (Mergesort) *	107
Recherche du maximum et du minimum d'une liste de nombres	108
Recherche de la médiane d'une liste de nombres*	109
Suite de Syracuse et temps de vol	110
Suite de Syracuse et altitude maximale	111
Ensemble de Mandelbrot.....	112
Ensemble de Julia	113
Fougère de Barnsley (Barnsley's Fern).....	114
Aide-mémoire des instructions Python.....	115





Chapitre 1.

Tous les algorithmes de 1^{re} année

Chapitres concernés :

Fonctions, résolution approchée d'équations

Calcul intégral (approché)

Équations différentielles

Arithmétique

Polynômes

Matrices

Déterminants

Systèmes linéaires

Listes

Courbes paramétrées

Probabilités

TOUS LES ALGORITHMES

PROGRAMMATION

P Y T H O N

Suivant l'adage, c'est en forgeant qu'on devient forgeron.

Dans ce livre, nous avons fait le pari que c'est en programmant qu'on devient programmeur. Tu trouveras donc dans cet ouvrage :

- ▶ Les algorithmes incontournables de prépa.
- ▶ Une présentation immédiate des algorithmes pour t'y retrouver facilement, sans te décourager.
- ▶ Des bons algorithmes simples, testés et qui fonctionnent ! (en Python)

Ce livre s'adresse donc aux étudiants de prépa mais aussi de Licence, de BTS et d'IUT dans la programmation et la compréhension des algorithmes.

***Thomas Petit** est professeur agrégé de Mathématiques. Auteur également d'ouvrages des collections Méthod'S et Coach aux éditions Ellipses, il enseigne actuellement au lycée Marie Curie de Nogent-sur-Oise.*

Du même auteur

