

Avant propos

C'est en forgeant qu'on devient forgeron...

Ce vieil adage, dont les hommes ont usé depuis la nuit des temps, est mis dans cet ouvrage au service des étudiants de second cycle de mathématiques et d'informatique, des élèves-ingénieurs et de toute autre personne souhaitant goûter les joies et maîtriser les difficultés de l'algorithmique.

Au cœur de l'informatique, cette science est celle de la conception de méthodes efficaces pour résoudre des problèmes à l'aide d'un ordinateur. Elle définit des outils généraux permettant de répondre aux questions suivantes : Sur quelle propriété mathématique d'un problème peut-on établir une méthode de résolution ? Etant donné un algorithme, résout-il le problème posé ? Lorsque l'on dispose de deux méthodes de résolution différentes, laquelle est préférable ?

L'algorithmique a donné lieu à de nombreux ouvrages remarquables depuis plus de trente ans, sur lesquels se fondent les enseignements dispensés dans les universités et écoles d'ingénieurs, et dont nous donnons une liste non exhaustive à la fin de cet ouvrage. L'expérience des auteurs, enseignants chevronnés dans différentes universités, les a depuis longtemps confrontés aux difficultés de leurs étudiants face à cette matière. Celles-ci semblent de plusieurs ordres :

- il est nécessaire de pouvoir expérimenter les algorithmes sur différents exemples pour en comprendre intuitivement le fonctionnement ;
- apprendre des éléments de cours implique de pouvoir refaire, au besoin, les démonstrations qui y ont été présentées, de les rédiger ;
- les algorithmes manipulent des entités qui évoluent dans le temps, et cela ajoute une dimension inhabituelle aux raisonnements mathématiques nécessaires pour les prouver et analyser leurs performances ;
- l'apprentissage souvent simultané d'un langage de programmation peut parfois ajouter des difficultés de compréhension dues à la syntaxe propre du langage.

C'est pour répondre à ces difficultés que les auteurs ont formé le projet de cet ouvrage, à partir d'exercices progressifs conçus lors de leurs années de pratique, et utilisés dans le cadre de travaux dirigés, d'examens, ou pour des devoirs de plus grande envergure. L'ouvrage a pour objectif d'aider l'étudiant dans son apprentissage de la conception et de l'analyse d'algorithmes en insistant sur le raisonnement et sa rédaction, en vue d'écrire dans le langage de son choix des programmes efficaces.

Si la plupart des ouvrages de cours d'algorithmique contiennent des

énoncés d'exercice, peu sont corrigés. C'est donc l'une des originalités de ce livre, que de proposer une correction entièrement rédigée, rigoureuse et complète de chaque question.

On y trouvera, pour chaque notion, des exercices visant la compréhension du cours, qui permettent d'appliquer un algorithme connu à des données numériques, ou de redémontrer, à partir d'éléments de base dont les définitions sont explicitées, les propriétés de certains algorithmes du cours décomposées en questions progressives. On y trouvera aussi des exercices qui enrichissent des algorithmes classiques de nouvelles fonctionnalités, ou qui permettent de les intégrer dans des applications originales.

Concernant la programmation, le parti pris de cet ouvrage est d'employer, pour la rédaction des algorithmes, un pseudo-langage impératif, plutôt qu'un véritable langage de programmation, afin de se concentrer sur ce qui ressort de la conception de l'algorithme, et non sur son implémentation. Ce choix devrait toutefois permettre une implémentation aisée des algorithmes dans des langages comme Pascal, C ou C++, avec sans doute un effort supplémentaire en ce qui concerne les langages objets. Nous indiquons plus loin les conventions d'écriture de ce langage, auxquelles le lecteur pourra se référer pour comprendre, mais aussi pour programmer les algorithmes dans le langage de son choix.

L'ouvrage aborde un panel assez vaste de domaines d'application de l'algorithmique, et devrait couvrir la plupart des cours dispensés actuellement en second cycle mathématique et informatique et en école d'ingénieurs.

On y développe notamment, après un chapitre méthodologique qui traite de la preuve et de l'analyse de la complexité d'un algorithme, les types de données abstraits pour représenter, gérer et manipuler des ensembles dynamiques, et leur implémentation à l'aide de structures de données linéaires ou arborescentes (piles, files, listes, arbres binaires de recherche, arbres équilibrés, tas). On aborde ensuite l'étude des algorithmes de tri. Les chapitres suivants sont consacrés à l'étude de l'algorithmique de base des graphes valués et non valués : connexité, accessibilité, parcours, arbres couvrants et chemins de coût minimum, pour ne citer que les principaux problèmes abordés. Ensuite, deux chapitres consacrés l'un aux algorithmes relatifs à la géométrie plane et l'autre aux algorithmes relatifs aux automates et aux mots achèvent cet ouvrage.

Dans chaque section, les exercices sont présentés dans un ordre de difficulté croissant, sauf nécessité de regroupement thématique, et souvent les questions d'un exercice respectent une certaine progressivité dans la difficulté.

D'autres algorithmes auraient mérité de figurer dans ce livre, plus proches

de l'algorithmique numérique, ou de la recherche opérationnelle. Avec quelques encouragements, les auteurs pourraient envisager une suite...

Prérequis

Le livre nécessite les connaissances de mathématiques de niveau DEUG, en particulier une bonne maîtrise du raisonnement par récurrence, ainsi que la connaissance de base d'un langage de programmation. Chacun des chapitres demande des prérequis, explicités dans les exercices lorsque cela semble nécessaire.

Il est recommandé de traiter les exercices en ayant d'autre part étudié un cours oral ou un livre de référence. Toutefois, de nombreux exercices peuvent être traités en se référant uniquement aux définitions et résultats de base rappelés en début de sous-chapitre. Cet en-tête de chapitre n'a pas pour objectif d'être un abrégé de cours. Les définitions présentées sont loin d'être exhaustives, mais sont celles jugées indispensables pour traiter les exercices avec le plus d'autonomie possible.

Conventions relatives à la présentation des algorithmes

Les algorithmes se présentent sous la forme de segments de code, de fonctions ou de procédures. Les types des paramètres, des fonctions et des variables sont toujours explicités. Par défaut, les passages de paramètres se font par valeur. Un texte accompagne l'algorithme lorsqu'un paramètre est modifié, et que le passage doit se faire par référence.

Comme dans tous langages de programmation impératifs, les structures de contrôle suivantes sont employées :

- **si** condition **alors** instruction **sinon** instruction **fin** ;
- **tant que** condition **faire** instruction **fin** ;
- **répéter** instruction **jusqu'à** condition **fin** ;
- **pour** variable de valeur initiale **à** valeur finale **faire** instruction **fin** ;
- **pour tout** élément d'un ensemble **faire** instruction **fin** ;

Les tableaux sont indexés par des entiers, et un élément d'indice i d'un tableau T s'écrit $T[i]$, de même les éléments d'un tableau M à deux dimensions se notent $M[i, j]$.

La principale spécificité du pseudo-langage est le traitement des pointeurs. D'abord, nous nous affranchissons des problèmes liés à l'allocation dynamique de mémoire (les objets manipulés sont supposés alloués). Ensuite, pour ne pas alourdir la présentation des algorithmes, lorsqu'une va-

riable x désigne un pointeur sur un enregistrement à plusieurs champs, par exemple un enregistrement comportant des champs nommés a et b , alors on notera $x.a$ la variable correspondant au champ a de l'enregistrement pointé par x . La même notation sera employée lorsque x représente l'enregistrement lui-même. Un pointeur qui ne pointe sur aucun enregistrement a la valeur nul.

Conventions relatives à l'index

Les entrées de l'index font référence à des notions de base traitées dans l'ouvrage. Lorsqu'une notion fait l'objet d'une définition explicite, ou d'un exercice qui lui est spécifiquement consacré, le numéro de page référencé apparaît en gras. Lorsqu'il s'agit d'un algorithme, le numéro de page apparaît en italique.

Remerciements

Les auteurs se souviennent avec tendresse du vide de leurs premiers regards qu'une explication miraculeuse a un jour allumé d'une petite flamme.

Ils saluent leurs maîtres, qui leur ont inculqué une forme de pensée archaïque consistant à chercher en toute circonstance la maîtrise absolue de la situation, par l'intermédiaire d'une preuve au pire longue et laborieuse, au mieux élégante et belle.

Ils arrosent d'une larme émue les bancs des facultés, usés par ceux qui ont, depuis quinze ans, à leur insu, participé à la conception de cet ouvrage.

Ils remercient chaleureusement les collègues dont ils ont sans vergogne pillé les archives.

Enfin, ils se réjouissent des jours de liberté intellectuelle que leur procurera la fin de cette aventure.