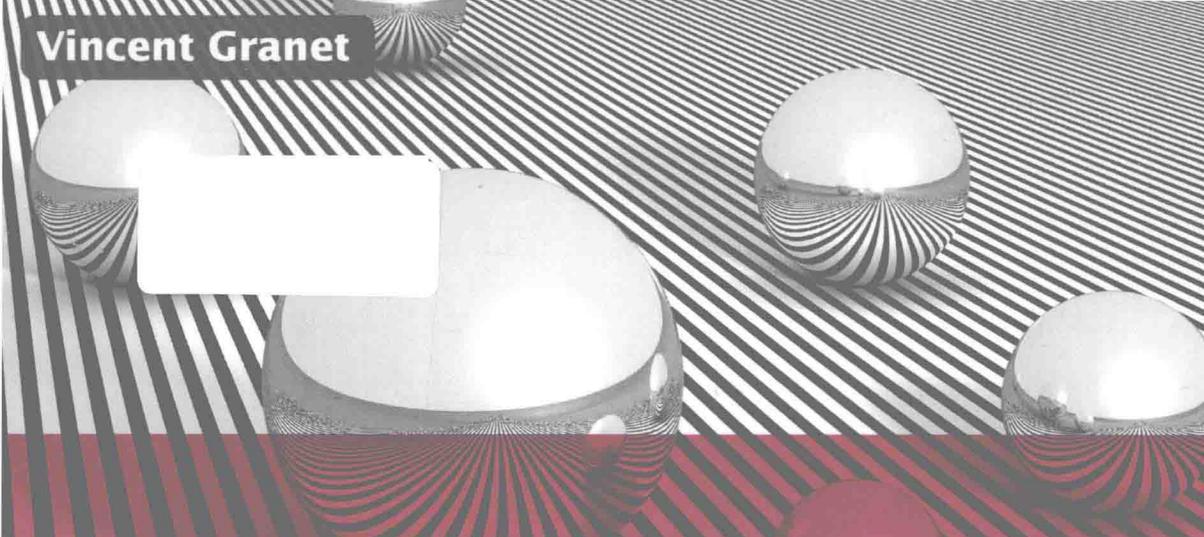


Vincent Granet

An abstract geometric pattern featuring several spheres of varying sizes. The spheres are rendered with a metallic, reflective texture. The background consists of a grid of lines that create a strong sense of perspective and depth, with some lines appearing to converge towards a vanishing point. The overall color palette is primarily black, white, and grey, with a touch of red at the bottom of the pattern.

Algorithmique et programmation en Java

Cours et exercices corrigés

4^e édition

RESSOURCES



NUMÉRIQUES

DUNOD

Vincent Granet

Algorithmique et programmation en Java

Cours et exercices corrigés

4^e édition



DUNOD

Illustration de couverture :

Abstract background - Spheres © Dreaming Andy – Fotolia.com

<p>Le pictogramme qui figure ci-contre mérite une explication. Son objet est d'alerter le lecteur sur la menace que représente pour l'avenir de l'écrit, particulièrement dans le domaine de l'édition technique et universitaire, le développement massif du photocopillage.</p> <p>Le Code de la propriété intellectuelle du 1^{er} juillet 1992 interdit en effet expressément la photocopie à usage collectif sans autorisation des ayants droit. Or, cette pratique s'est généralisée dans les établissements</p>	<p>d'enseignement supérieur, provoquant une baisse brutale des achats de livres et de revues, au point que la possibilité même pour les auteurs de créer des œuvres nouvelles et de les faire éditer correctement est aujourd'hui menacée. Nous rappelons donc que toute reproduction, partielle ou totale, de la présente publication est interdite sans autorisation de l'auteur, de son éditeur ou du Centre français d'exploitation du droit de copie (CFC, 20, rue des Grands-Augustins, 75006 Paris).</p>
	

© Dunod, 2000, 2004, 2010, 2014

5 rue Laromiguière, 75005 Paris
www.dunod.com

ISBN 978-2-10-071287-8

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 donc une contrefaçon sanctionnée par les articles L. 335-2 et suivants du Code de la propriété intellectuelle.

à Maud

Avant-propos

Longtemps attendue, la version 8 de JAVA est sortie en mars 2014, avec de nombreuses nouveautés. Sans doute, la plus importante est l'ajout des *fonctions anonymes* (lambda expressions) et son incidence tant sur le langage que sur son API.

Pour cette quatrième édition d'*Algorithmique et Programmation en Java*, l'ensemble de l'ouvrage a été révisé pour tenir compte des *fonctions anonymes*. En particulier, il inclut un nouveau chapitre qui présente le paradigme fonctionnel et la façon dont il est mis en œuvre avec JAVA 8.

L'informatique est une science mais aussi une technologie et un ensemble d'outils. Ces trois composantes ne doivent pas être confondues, et l'enseignement de l'informatique ne doit pas être réduit au seul apprentissage des logiciels. Ainsi, l'activité de programmation ne doit pas se confondre avec l'étude d'un langage de programmation particulier. Même si l'importance de ce dernier ne doit pas être sous-estimée, il demeure un simple outil de mise en œuvre de concepts algorithmiques et de programmation généraux et fondamentaux. L'objectif de cet ouvrage est d'enseigner au lecteur des méthodes et des outils de construction de programmes informatiques valides et fiables.

L'étude de l'algorithmique et de la programmation est un des piliers fondamentaux sur lesquels repose l'enseignement de l'informatique. Ce livre s'adresse principalement aux étudiants des cycles informatiques et élèves ingénieurs informaticiens, mais aussi à tous ceux qui ne se destinent pas à la carrière informatique mais qui seront certainement confrontés au développement de programmes informatiques au cours de leur scolarité ou dans leur vie professionnelle.

Les seize premiers chapitres présentent les concepts de base de la programmation *impérative* en s'appuyant sur une *methodologie objet*. Le chapitre 13 est également dédié à la *programmation fonctionnelle*. Ils mettent en particulier l'accent sur la notion de preuve des programmes grâce à la notion d'*affirmations* (antécédent, conséquent, invariant) dont la vérification formelle garantit la validité de programmes. Ils introduisent aussi la notion de *complexité* des algorithmes pour évaluer leur performance.

Les onze derniers chapitres étudient en détail les structures de données abstraites classiques (liste, graphe, arbre...) et de nombreux d'algorithmes fondamentaux (recherche, tri, jeux et stratégie...) que tout étudiant en informatique doit connaître et maîtriser. D'autre part, un chapitre est consacré aux interfaces graphiques et à leur programmation avec SWING.

La présentation des concepts de programmation cherche à être indépendante, autant que faire se peut, d'un langage de programmation particulier. Les algorithmes seront décrits dans une notation algorithmique épurée. Pour des raisons pédagogiques, il a toutefois bien fallu faire le choix d'un langage pour programmer les structures de données et les algorithmes présentés dans cet ouvrage. Ce choix s'est porté sur le langage à objets JAVA [GJS96], non pas par effet de mode, mais plutôt pour les qualités de ce langage, malgré quelques défauts. Ses qualités sont en particulier sa relative simplicité pour la mise en œuvre des algorithmes, un large champ d'application et sa grande disponibilité sur des environnements variés. Ce dernier point est en effet important ; le lecteur doit pouvoir disposer facilement d'un compilateur et d'un interprète afin de résoudre les exercices proposés à la fin des chapitres. Enfin, JAVA est de plus en plus utilisé comme langage d'apprentissage de la programmation dans les universités. Pour les défauts, on peut par exemple regretter l'absence de l'héritage multiple, et la présence de constructions archaïques héritées du langage C. Ce livre n'est toutefois pas un ouvrage d'apprentissage du langage JAVA. Même si les éléments du langage nécessaires à la mise en œuvre des notions d'algorithmique et de programmation ont été introduits, ce livre n'enseignera pas au lecteur les finesses et les arcanes de JAVA, pas plus qu'il ne décrira les nombreuses classes de l'API. Le lecteur intéressé pourra se reporter aux très nombreux ouvrages qui décrivent le langage en détail, comme par exemple [GR11, Bro99, Ska00, Eck00].

Les corrigés de la plupart des exercices, ainsi que des applets qui proposent une vision graphique de certains programmes présentés dans l'ouvrage sont accessibles sur le site web de l'auteur à l'adresse :

`www.polytech.unice.fr/~vg/fr/livres/algojava`

Cet ouvrage doit beaucoup à de nombreuses personnes. Tout d'abord, aux auteurs des algorithmes et des techniques de programmation qu'il présente. Il n'est pas possible de les citer tous ici, mais les références à leurs principaux textes sont dans la bibliographie. À Olivier Lecarme et Jean-Claude Boussard, mes professeurs à l'université de Nice qui m'ont enseigné cette discipline au début des années 1980. Je tiens tout particulièrement à remercier ce dernier qui fut le tout premier lecteur attentif de cet ouvrage alors qu'il n'était encore qu'une ébauche, et qui m'a encouragé à poursuivre sa rédaction. À Carine Fédèle qui a bien voulu lire et corriger ce texte à de nombreuses reprises, qu'elle en soit spécialement remercié. Enfin, à mes collègues et mes étudiants qui m'ont aidé et soutenu dans cette tâche ardue qu'est la rédaction d'un livre.

Enfin, je remercie toute l'équipe Dunod, Carole Trochu, Jean-Luc Blanc, et Romain Henion, pour leur aide précieuse et leurs conseils avisés qu'ils m'ont apportés pour la publication des quatre éditions de cet ouvrage.

Sophia Antipolis, avril 2014.

Table des matières

AVANT-PROPOS	XV
CHAPITRE 1 • INTRODUCTION	1
1.1 Environnement matériel	1
1.2 Environnement logiciel	4
1.3 Les langages de programmation	5
1.4 Construction des programmes	11
1.5 Démonstration de validité	13
CHAPITRE 2 • ACTIONS ÉLÉMENTAIRES	15
2.1 Lecture d'une donnée	15
2.2 Exécution d'une routine prédéfinie	16
2.3 Écriture d'un résultat	17
2.4 Affectation d'un nom à un objet	17
2.5 Déclaration d'un nom	18
2.5.1 Déclaration de constantes	18
2.5.2 Déclaration de variables	19
2.6 Règles de déduction	19
2.6.1 L'affectation	19
2.6.2 L'appel de procédure	20
2.7 Le programme sinus écrit en Java	20
2.8 Exercices	22
CHAPITRE 3 • TYPES ÉLÉMENTAIRES	23
3.1 Le type entier	24
3.2 Le type réel	25

3.3	Le type booléen	28
3.4	Le type caractère	29
3.5	Constructeurs de types simples	31
3.5.1	Les types énumérés	31
3.5.2	Les types intervalles	32
3.6	Exercices	32
CHAPITRE 4 • EXPRESSIONS		35
4.1	Évaluation	36
4.1.1	Composition du même opérateur plusieurs fois	36
4.1.2	Composition de plusieurs opérateurs différents	36
4.1.3	Parenthésage des parties d'une expression	37
4.2	Type d'une expression	37
4.3	Conversions de type	38
4.4	Un exemple	38
4.5	Exercices	41
CHAPITRE 5 • ÉNONCÉS STRUCTURÉS		43
5.1	Énoncé composé	43
5.2	Énoncés conditionnels	44
5.2.1	Énoncé choix	44
5.2.2	Énoncé si	46
5.3	Résolution d'une équation du second degré	47
5.4	Exercices	50
CHAPITRE 6 • PROCÉDURES ET FONCTIONS		53
6.1	Intérêt	53
6.2	Déclaration d'une routine	54
6.3	Appel d'une routine	55
6.4	Transmission des paramètres	56
6.4.1	Transmission par valeur	57
6.4.2	Transmission par résultat	57
6.5	Retour d'une routine	57
6.6	Localisation	58
6.7	Règles de déduction	60
6.8	Exemples	61
6.9	Exercices	65
CHAPITRE 7 • PROGRAMMATION PAR OBJETS		67
7.1	Objets et classes	67
7.1.1	Création des objets	68
7.1.2	Destruction des objets	69
7.1.3	Accès aux attributs	69

7.1.4	Attributs de classe partagés	70
7.1.5	Les classes en Java	70
7.2	Les méthodes	71
7.2.1	Accès aux méthodes	72
7.2.2	Constructeurs	72
7.2.3	Constructeurs en Java	73
7.2.4	Les méthodes en Java	73
7.3	Assertions sur les classes	75
7.4	Exemples	76
7.4.1	Équation du second degré	76
7.4.2	Date du lendemain	79
7.5	Exercices	82
CHAPITRE 8 • ÉNONCÉS ITÉRATIFS		85
8.1	Forme générale	85
8.2	L'énoncé tant que	86
8.3	L'énoncé répéter	87
8.4	Finitude	88
8.5	Exemples	88
8.5.1	Factorielle	88
8.5.2	Minimum et maximum	89
8.5.3	Division entière	89
8.5.4	Plus grand commun diviseur	90
8.5.5	Multiplication	91
8.5.6	Puissance	91
8.6	Exercices	92
CHAPITRE 9 • LES TABLEAUX		95
9.1	Déclaration d'un tableau	95
9.2	Dénotation d'un composant de tableau	96
9.3	Modification sélective	97
9.4	Opérations sur les tableaux	97
9.5	Les tableaux en Java	97
9.6	Un exemple	99
9.7	Les chaînes de caractères	101
9.8	Exercices	102
CHAPITRE 10 • L'ÉNONCÉ ITÉRATIF POUR		105
10.1	Forme générale	105
10.2	Forme restreinte	106
10.3	Les énoncés pour de Java	106
10.4	Exemples	108
10.4.1	Le schéma de HORNER	108

10.4.2	Un tri interne simple	109
10.4.3	Confrontation de modèle	110
10.5	Complexité des algorithmes	114
10.6	Exercices	116
CHAPITRE 11 • LES TABLEAUX À PLUSIEURS DIMENSIONS		119
11.1	Déclaration	119
11.2	Dénotation d'un composant de tableau	120
11.3	Modification sélective	120
11.4	Opérations	121
11.5	Tableaux à plusieurs dimensions en Java	121
11.6	Exemples	122
11.6.1	Initialisation d'une matrice	122
11.6.2	Matrice symétrique	122
11.6.3	Produit de matrices	123
11.6.4	Carré magique	124
11.7	Exercices	126
CHAPITRE 12 • HÉRITAGE		131
12.1	Classes héritières	131
12.2	Redéfinition de méthodes	134
12.3	Recherche d'un attribut ou d'une méthode	135
12.4	Polymorphisme et liaison dynamique	136
12.5	Classes abstraites	137
12.6	Héritage simple et multiple	138
12.7	Héritage et assertions	139
12.7.1	Assertions sur les classes héritières	139
12.7.2	Assertions sur les méthodes	139
12.8	Relation d'héritage ou de clientèle	139
12.9	L'héritage en Java	140
CHAPITRE 13 • FONCTIONS ANONYMES		143
13.1	Paramètres fonctions	144
13.1.1	Fonctions anonymes en Java	145
13.1.2	Foreach et map	147
13.1.3	Continuation	148
13.2	Fonctions anonymes en résultat	150
13.2.1	Composition	150
13.2.2	Curryfication	151
13.3	Fermeture	152
13.3.1	Fermeture en Java	153
13.3.2	Lambda récursives	154
13.4	Exercices	155

CHAPITRE 14 • LES EXCEPTIONS	157
14.1 Émission d'une exception	157
14.2 Traitement d'une exception	158
14.3 Le mécanisme d'exception de Java	159
14.3.1 Traitement d'une exception	159
14.3.2 Émission d'une exception	160
14.4 Exercices	161
CHAPITRE 15 • LES FICHIERS SÉQUENTIELS	163
15.1 Déclaration de type	164
15.2 Notation	164
15.3 Manipulation des fichiers	165
15.3.1 Écriture	165
15.3.2 Lecture	166
15.4 Les fichiers de Java	167
15.4.1 Fichiers d'octets	167
15.4.2 Fichiers d'objets élémentaires	169
15.4.3 Fichiers d'objets structurés	173
15.5 Les fichiers de texte	173
15.6 Les fichiers de texte en Java	174
15.7 Exercices	178
CHAPITRE 16 • RÉCURSIVITÉ	181
16.1 Récursivité des actions	182
16.1.1 Définition	182
16.1.2 Finitude	182
16.1.3 Écriture récursive des routines	182
16.1.4 La pile d'évaluation	185
16.1.5 Quand ne pas utiliser la récursivité ?	186
16.1.6 Récursivité directe et croisée	188
16.2 Récursivité des objets	190
16.3 Exercices	192
CHAPITRE 17 • STRUCTURES DE DONNÉES	195
17.1 Définition d'un type abstrait	196
17.2 L'implémentation d'un type abstrait	198
17.3 Utilisation du type abstrait	200
CHAPITRE 18 • STRUCTURES LINÉAIRES	203
18.1 Les listes	203
18.1.1 Définition abstraite	204
18.1.2 L'implémentation en Java	205
18.1.3 Énumération	216

18.2	Les piles	219
18.2.1	Définition abstraite	220
18.2.2	L'implémentation en Java	221
18.3	Les files	223
18.3.1	Définition abstraite	224
18.3.2	L'implémentation en Java	225
18.4	Les dèques	226
18.4.1	Définition abstraite	226
18.4.2	L'implémentation en Java	227
18.5	Exercices	228
CHAPITRE 19 • GRAPHERS		231
19.1	Terminologie	232
19.2	Définition abstraite d'un graphe	233
19.3	L'implémentation en Java	234
19.3.1	Matrice d'adjacence	235
19.3.2	Listes d'adjacence	238
19.4	Parcours d'un graphe	239
19.4.1	Parcours en profondeur	239
19.4.2	Parcours en largeur	240
19.4.3	Programmation en Java des parcours de graphe	241
19.5	Exercices	243
CHAPITRE 20 • STRUCTURES ARBORESCENTES		245
20.1	Terminologie	246
20.2	Les arbres	247
20.2.1	Définition abstraite	248
20.2.2	L'implémentation en Java	249
20.2.3	Algorithmes de parcours d'un arbre	251
20.3	Arbre binaire	252
20.3.1	Définition abstraite	254
20.3.2	L'implémentation en Java	255
20.3.3	Parcours d'un arbre binaire	257
20.4	Représentation binaire des arbres généraux	259
20.5	Exercices	260
CHAPITRE 21 • TABLES		263
21.1	Définition abstraite	264
21.1.1	Ensembles	264
21.1.2	Description fonctionnelle	264
21.1.3	Description axiomatique	264
21.2	Représentation des éléments en Java	264
21.3	Représentation par une liste	266

21.3.1	Liste non ordonnée	266
21.3.2	Liste ordonnée	268
21.3.3	Recherche dichotomique	270
21.4	Représentation par un arbre ordonné	272
21.4.1	Recherche d'un élément	273
21.4.2	Ajout d'un élément	273
21.4.3	Suppression d'un élément	274
21.5	Les arbres AVL	276
21.5.1	Rotations	277
21.5.2	Mise en œuvre	280
21.6	Arbres 2-3-4 et bicolores	284
21.6.1	Les arbres 2-3-4	284
21.6.2	Mise en œuvre en Java	288
21.6.3	Les arbres bicolores	289
21.6.4	Mise en œuvre en Java	294
21.7	Tables d'adressage dispersé	299
21.7.1	Le problème des collisions	300
21.7.2	Choix de la fonction d'adressage	301
21.7.3	Résolution des collisions	302
21.8	Exercices	306
CHAPITRE 22 • FILES AVEC PRIORITÉ		311
22.1	Définition abstraite	311
22.2	Représentation avec une liste	312
22.3	Représentation avec un tas	312
22.3.1	Premier	313
22.3.2	Ajouter	314
22.3.3	Supprimer	315
22.3.4	L'implémentation en Java	316
22.4	Exercices	319
CHAPITRE 23 • ALGORITHMES DE TRI		321
23.1	Introduction	321
23.2	Tris internes	322
23.2.1	L'implantation en Java	322
23.2.2	Méthodes par sélection	323
23.2.3	Méthodes par insertion	328
23.2.4	Tri par échanges	333
23.2.5	Comparaisons des méthodes	337
23.3	Tris externes	339
23.4	Exercices	342

CHAPITRE 24 • ALGORITHMES SUR LES GRAPHS	345
24.1 Composantes connexes	345
24.2 Fermeture transitive	347
24.3 Plus court chemin	349
24.3.1 Algorithme de Dijkstra	349
24.3.2 Algorithme A*	354
24.3.3 Algorithme IDA*	356
24.4 Tri topologique	357
24.4.1 L'implémentation en Java	359
24.4.2 Existence de cycle dans un graphe	360
24.4.3 Tri topologique inverse	360
24.4.4 L'implémentation en Java	361
24.5 Exercices	361
 CHAPITRE 25 • ALGORITHMES DE RÉTRO-PARCOURS	 365
25.1 Écriture récursive	365
25.2 Le problème des huit reines	367
25.3 Écriture itérative	369
25.4 Problème des sous-suites	370
25.5 Jeux de stratégie	372
25.5.1 Stratégie <i>MinMax</i>	372
25.5.2 Coupure α - β	375
25.5.3 Profondeur de l'arbre de jeu	378
25.6 Exercices	379
 CHAPITRE 26 • INTERFACES GRAPHIQUES	 383
26.1 Systèmes interactifs	383
26.2 Conception d'une application interactive	385
26.3 Environnements graphiques	387
26.3.1 Système de fenêtrage	387
26.3.2 Caractéristiques des fenêtres	389
26.3.3 Boîtes à outils	392
26.3.4 Générateurs	394
26.4 Interfaces graphiques en Java	394
26.4.1 Une simple fenêtre	394
26.4.2 Convertisseur Celcius-Fahrenheit	397
26.4.3 Un composant graphique pour visualiser des couleurs	401
26.4.4 Applets	403
26.5 Exercices	405
 BIBLIOGRAPHIE	 407
 INDEX	 411

Chapitre 1

Introduction

Les informaticiens, ou les simples usagers de l'outil informatique, utilisent des systèmes informatiques pour concevoir ou exécuter des programmes d'application. Nous considérons qu'un environnement informatique est formé d'une part d'un ordinateur et de ses équipements externes, que nous appellerons *environnement matériel*, et d'autre part d'un système d'exploitation avec ses programmes d'application, que nous appellerons *environnement logiciel*. Les programmes qui forment le logiciel réclament des méthodes pour les construire, des langages pour les rédiger et des outils pour les exécuter sur un ordinateur.

Dans ce chapitre, nous introduirons la terminologie et les notions de base des ordinateurs et de la programmation. Nous présenterons les notions d'environnement de développement et d'exécution d'un programme, nous expliquerons ce qu'est un langage de programmation et nous introduirons les méthodes de construction des programmes.

1.1 ENVIRONNEMENT MATÉRIEL

Un *automate* est un ensemble fini de composants physiques pouvant prendre des états identifiables et reproductibles en nombre fini, auquel est associé un ensemble de changements d'états non instantanés qu'il est possible de commander et d'enchaîner sans intervention humaine.

Un *ordinateur* est un automate *déterministe* à composants électroniques. Tous les ordinateurs, tout au moins les ordinateurs monoprocesseurs, sont construits, peu ou prou, sur le modèle proposé en 1944 par le mathématicien américain d'origine hongroise VON NEUMANN. Un ordinateur est muni :

- D'une *mémoire*, dite *centrale* ou *principale*, qui contient deux sortes d'informations : d'une part l'information traitante, les *instructions*, et d'autre part l'information trai-

tée, les *données*. Cette mémoire est formée d'un ensemble de cellules, ou *mots*, ayant chacune une *adresse unique*, et contenant des instructions ou des données. La représentation de l'information est faite grâce à une codification *binaires*, 0 ou 1. On appelle *longueur de mot*, caractéristique d'un ordinateur, le nombre d'éléments binaires, appelés *bits*, groupés dans une simple cellule. Les longueurs de mots usuelles des ordinateurs actuels (ou passés) sont, par exemple, 8, 16, 24, 32, 48 ou 64 bits. Cette mémoire possède une capacité *finie*, exprimée en gigaoctets (Go) ; un *octet* est un ensemble de 8 bits, un kilo-octet (Ko) est égal à 1024 octets, un mégaoctet est égal à 1024 Ko, un gigaoctet (Go) est égal à 1024 Mo, et enfin un téraoctet (To) est égal à 1024 Go. Actuellement, les tailles courantes des mémoires centrales des ordinateurs individuels varient entre 4 Go à 8 Go¹.

- D'une *unité centrale de traitement*, formée d'une *unité de commande* (UC) et d'une *unité arithmétique et logique* (UAL). L'unité de commande extrait de la mémoire centrale les instructions et les données sur lesquelles portent les instructions ; elle déclenche le traitement de ces données dans l'unité arithmétique et logique, et éventuellement range le résultat en mémoire centrale. L'unité arithmétique et logique effectue sur les données qu'elle reçoit les traitements commandés par l'unité de commande.
- De *registres*. Les registres sont des unités de mémorisation, en petit nombre (certains ordinateurs n'en ont pas), qui permettent à l'unité centrale de traitement de ranger de façon temporaire des données pendant les calculs. L'accès à ces registres est très rapide, beaucoup plus rapide que l'accès à une cellule de la mémoire centrale. Le rapport entre les temps d'accès à un registre et à la mémoire centrale est de l'ordre de 100.
- D'*unités d'échanges* reliées à des périphériques pour échanger de l'information avec le monde extérieur. L'unité de commande dirige les unités d'échange lorsqu'elle rencontre des instructions d'entrée-sortie.

Jusqu'au milieu des années 2000, les constructeurs étaient engagés dans une course à la vitesse avec des microprocesseurs toujours plus rapides. Toutefois, les limites de la physique actuelle ont été atteintes et depuis 2006 la tendance nouvelle est de placer plusieurs (le plus possible) microprocesseurs sur une même puce (le circuit-intégré). Ce sont par exemple les processeurs 64 bits Core i7 d'INTEL ou AMD FX d'AMD, qui possèdent de 4 à 8 processeurs.

Les ordinateurs actuels possèdent aussi plusieurs niveaux de mémoire. Ils introduisent, entre le processeur et la mémoire centrale, des mémoires dites *caches* qui accélèrent l'accès aux données. Les mémoires caches peuvent être *primaires*, c'est-à-dire situées directement sur le processeur, ou *secondaires*, c'est-à-dire situées sur la carte mère. Certains ordinateurs introduisent même un troisième niveau de cache. En général, le rapport entre le temps d'accès entre les deux premiers niveaux de mémoire cache est d'environ 10 (le cache de niveau 1 est le plus rapide et le plus petit). Le temps d'accès entre la mémoire cache secondaire et la mémoire centrale est lui aussi d'un rapport d'environ 10.

Les *équipements externes*, ou *périphériques*, sont un ensemble de composants permettant de relier l'ordinateur au monde extérieur, et notamment à ses utilisateurs humains. On peut distinguer :

1. Notez qu'avec 32 bits, l'espace adressage est de 4 Go mais qu'en général les systèmes d'exploitation ne permettent d'utiliser qu'un espace mémoire de taille inférieure. Les machines 64 bits actuelles, avec un système d'exploitation adapté, permettent des tailles de mémoire centrale supérieures à 4 Go.