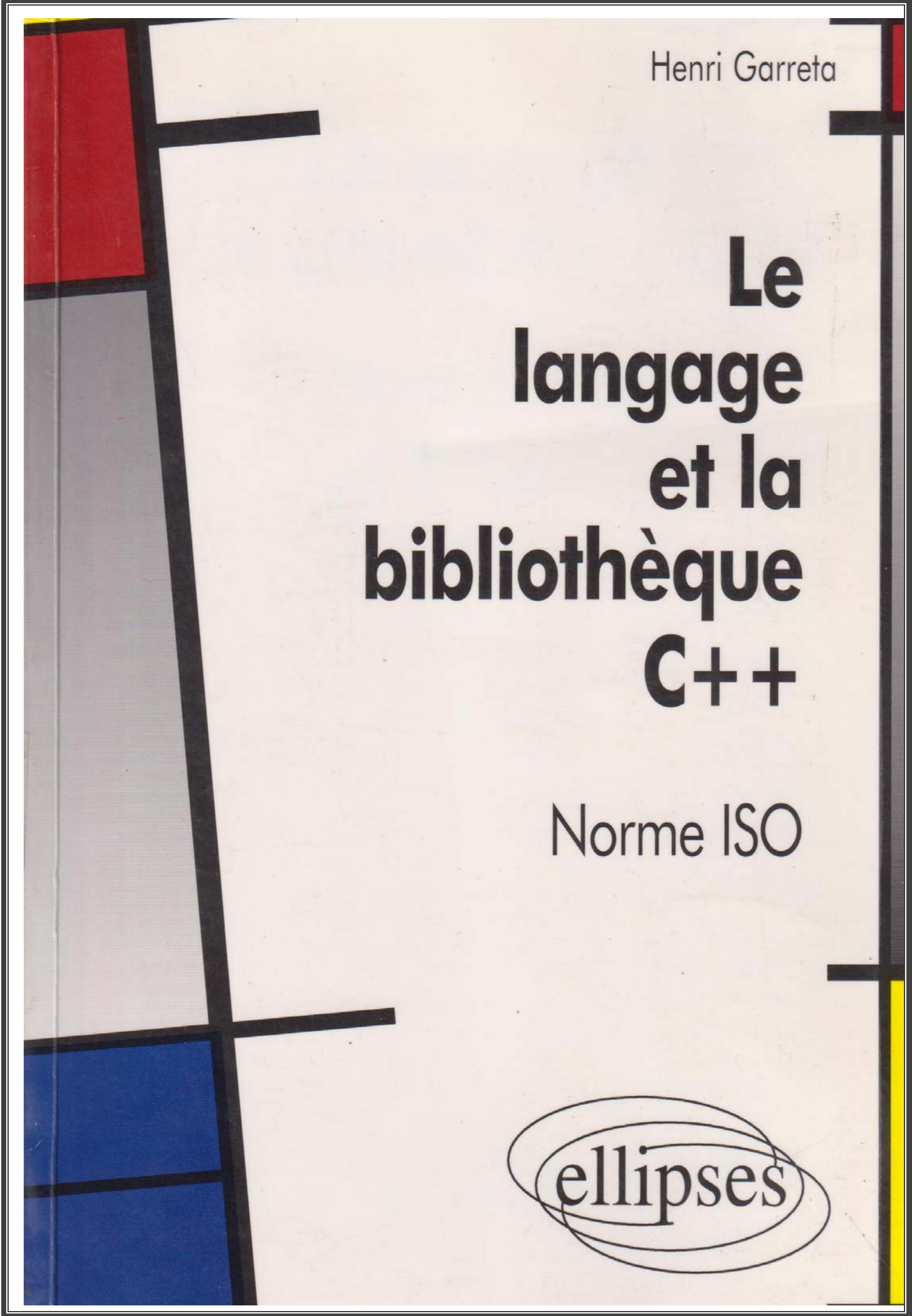
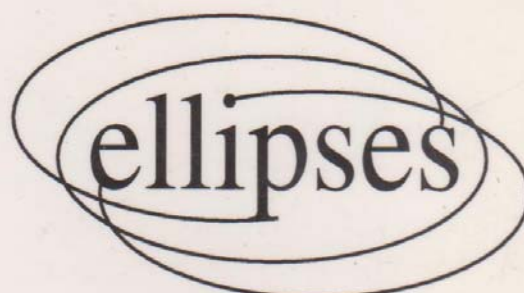




Henri Garreta

Le langage et la bibliothèque C++

Norme ISO



- 005-2343-2

2-005-343-2

Le langage et la bibliothèque C++

Norme ISO



Henri GARRETA

Maître de Conférences à la Faculté des Sciences
de Luminy (Marseille)

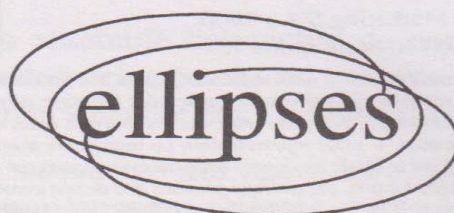


Table des matières

Avant-propos	3
Table des matières	5
Introduction. Programmation orientée objets	11
Chapitre 1. Avant les objets	17
1.1 Mots réservés	17
1.2 Statut des constantes	18
1.3 Statut et placement des définitions	18
1.3.1 Déclarations et définitions	18
1.3.2 Statut des définitions	19
1.4 Conversions de type	20
1.5 Déclaration des fonctions	24
1.5.1 Déclaration préalable	24
1.5.2 Paramètres anonymes	24
1.5.3 Valeurs par défaut des arguments	25
1.5.4 Fonctions en ligne	26
1.5.5 Les macros sont-elles utiles ?	27
1.6 Surcharge des noms de fonctions	27
1.6.1 Résolution de la surcharge	28
1.7 Appel de fonctions écrites en C	31
1.8 Types nouveaux	32
1.8.1 Booléens	32
1.8.2 Références	33
1.8.3 Enumérations, structures, unions et classes	37
1.8.4 Caractères étendus	38

1.9	Gestion dynamique de la mémoire	38
1.9.1	Allocation et restitution de mémoire	38
1.9.2	Surcharge de <i>new</i> et <i>delete</i>	39
1.9.3	Opérateurs de placement	40
1.9.4	Echec de l'allocation de mémoire	41

1.10	Flux standard d'entrée / sortie.	42
-------------	---	-----------

Chapitre 2. Classes **45**

2.1	Objets, classes et membres	45
2.1.1	Membres publics et privés	46
2.1.2	Fonctions membres	48
2.1.3	Comment un objet fait référence à lui même	49

2.2	Constructeurs	50
2.2.1	Constructeur par défaut	53
2.2.2	Constructeur par copie (clonage)	53

2.3	Destructeurs	56
------------	---------------------	-----------

2.4	Construction des membres	56
------------	---------------------------------	-----------

2.5	Membres constants	59
2.5.1	Données membres constantes	59
2.5.2	Fonctions membres constantes	60
2.5.3	Membres mutables	61

2.6	Membres statiques	62
2.6.1	Données membres statiques	63
2.6.2	Constantes membres statiques	63
2.6.3	Fonctions membres statiques	64

2.7	Pointeurs vers membres	65
------------	-------------------------------	-----------

2.8	Types imbriqués	67
------------	------------------------	-----------

2.9	Amis	69
2.9.1	Fonctions amies	69
2.9.2	Classes amies	71

2.10	Autres considérations sur les types	72
2.10.1	Tableaux	72
2.10.2	Unions	73
2.10.3	Classes agrégats	73

Chapitre 3. Opérateurs et conversions **75**

3.1	Surcharge des opérateurs	75
3.1.1	Principe	75
3.1.2	Surcharge par une fonction membre	76
3.1.3	Surcharge par une fonction non membre	76
3.2	Quelques opérateurs remarquables	78

3.2.1	Affectation	78
3.2.2	Opérateurs <i>new</i> et <i>delete</i>	79
3.2.3	Indexation	82
3.2.4	Appel de fonction	84
3.2.5	L'opérateur <i>-></i>	84
3.2.6	Opérateurs <i>++</i> et <i>--</i>	85
3.2.7	Cas des énumérations	86
3.3	Conversions définies par l'utilisateur	87
3.3.1	Constructeurs de conversion	87
3.3.2	Opérateurs de conversion	88
3.3.3	Le cas de l'indirection	89
Chapitre 4. Héritage		91
4.1	L'héritage	91
4.1.1	Principe et syntaxe	91
4.1.2	Membres protégés	92
4.1.3	Héritage privé, protégé et public	93
4.1.4	Redéfinition des fonctions membres	96
4.2	L'héritage, la création et la destruction des objets	97
4.2.1	Création et destruction des objets dérivés	97
4.2.2	Création et destruction des objets, le cas général	98
4.2.3	Membres synthétisés, le cas général	100
4.2.4	<i>Etre ou avoir ?</i>	101
4.3	Polymorphisme	102
4.3.1	Conversion standard vers une classe de base	102
4.3.2	Implémentation de la généralisation des pointeurs	104
4.3.3	Type statique, type dynamique, généralisation	105
4.3.4	Liaison dynamique, fonctions virtuelles	107
4.3.5	Implémentation des fonctions virtuelles	111
4.4	Autres remarques sur les fonctions virtuelles	114
4.4.1	Compilation de l'appel d'une fonction virtuelle	114
4.4.2	Constructeurs, destructeurs et opérateurs virtuels	115
4.5	Classes abstraites. Fonctions virtuelles pures	120
4.6	Identification dynamique du type	122
4.6.1	L'opérateur <i>dynamic_cast</i>	122
4.6.2	L'opérateur <i>typeid</i>	123
4.7	L'héritage multiple	124
4.7.1	L'héritage multiple et ses ambiguïtés	124
4.7.2	Classes de base virtuelles	128
4.7.3	Héritage virtuel et constructeurs	129
Chapitre 5. Modèles		131
5.1	Modèles de fonctions	131

5.2	Modèles de classes	134
5.2.1	Syntaxe	134
5.2.2	Qualification <i>typename</i>	136
5.2.3	Plus sur l'instanciation des modèles	136
5.2.4	Spécialisation	139
5.2.5	Modèles de classe et héritage	140
5.2.6	L'instanciation en pratique	143
Chapitre 6. Exceptions		145
6.1	Exceptions	145
6.1.1	Principe et syntaxe	145
6.1.2	Sélection du gestionnaire d'interception	148
6.1.3	Les exceptions qu'une fonction laisse échapper	150
6.1.4	La classe exception	151
6.1.5	Les fonctions <i>terminate</i> et <i>unexpected</i>	152
6.2	Exceptions et libération des ressources	153
6.2.1	Allocateurs	154
Chapitre 7. Espaces de noms		157
7.1	La visibilité des identificateurs	157
7.2	Espaces de noms	158
7.2.1	Notion et syntaxe	158
7.2.2	Utilisation partielle de l'opérateur de résolution de portée	160
7.2.3	Déclaration et directive <i>using</i>	161
7.2.4	Autres éléments des espaces de noms	162
Chapitre 8. Bibliothèque standard (début)		165
8.1	Structure générale	165
8.2	Support du langage	166
8.2.1	Types dépendant de l'implémentation	166
8.2.2	Propriétés de l'implémentation	167
8.2.3	Démarrage et terminaison des programmes	168
8.2.4	Gestion dynamique de la mémoire	169
8.2.5	Identification dynamique des types	170
8.2.6	Gestion des exceptions	170
8.2.7	Autres fonctions du « runtime »	172
8.3	Diagnostics	173
8.4	Chaînes de caractères	173
8.4.1	La classe string	173
8.4.2	Fonctions non membres	177
8.4.3	Exemples (rubrique « les string, c'est bien »)	178
8.5	Particularités locales	178

8.6	Composants numériques	184
8.6.1	Complex	184
8.6.2	Valarray	184
8.6.3	Numeric	189
8.7	Flux d'entrée-sortie	191
8.7.1	Structure générale	191
8.7.2	Lecture et écriture	193
8.7.3	Manipulateurs	196
8.7.4	Flux basés sur des chaînes de caractères	200
8.7.5	Flux basés sur des fichiers	201
8.7.6	Flux standard	203

Chapitre 9. Structures de données et algorithmes (la STL) 205

9.1	Utilitaires généraux	205
9.1.1	Utilitaires	205
9.1.2	Objets fonctions	207
9.1.3	Mémoire : allocateurs et auto-pointeurs	213
9.2	Conteneurs	216
9.2.1	Zoologie des conteneurs	216
9.2.2	Éléments communs à tous les conteneurs	217
9.2.3	Séquences : vector, list, deque	223
9.2.4	Types abstraits : stack, queue, priority_queue	226
9.2.5	Conteneurs associatifs : set, map	229
9.2.6	Conteneurs de bits : vector<bool>, bitset	233
9.3	Itérateurs	236
9.3.1	Notion	236
9.3.2	Programmation avec des itérateurs	238
9.3.3	Itérateurs prédéfinis	239
9.4	Algorithmes	242
9.4.1	Algorithmes de consultation de séquences	244
9.4.2	Algorithmes qui modifient des séquences	248
9.4.3	Algorithmes des séquences ordonnées	256

Références	267
-------------------	------------

Index	268
--------------	------------