

Vincent Gouvernelle

LE GUIDE DE SURVIE

C++

L'ESSENTIEL DU CODE ET DES COMMANDES



PEARSON

Table des matières

1	Bases héritées du langage C	1
	<i>Hello world</i> en C	1
	Commentaires	3
	Types fondamentaux	4
	Types élaborés	6
	Structures conditionnelles	9
	Structures de boucle	12
	Sauts	14
	Fonctions	15
	Préprocesseur	16
	Opérateurs et priorité (C et C++)	18
2	Bases du langage C++	23
	<i>Hello world</i> en C++	23
	Les mots-clés	24
	Les constantes	25
	Déclarations de variables	27
	Les nouveaux types de variables du C++	29
	Conversion de type C	32
	Conversion avec <i>static_cast</i>	33
	Conversion avec <i>const_cast</i>	34
	Conversion avec <i>reinterpret_cast</i>	35
	Conversion avec <i>dynamic_cast</i>	36
	Surcharge	37
	Les espaces de noms	40
	Incompatibilités avec le C	42
	Lier du code C et C++	44
	Embarquer une fonction	46
	Constantes usuelles	46

3	Pointeurs et références	47
	Créer et initialiser un pointeur.	48
	Accéder aux données ou fonctions membres	49
	Créer et utiliser une référence	50
	Déclarer un pointeur sur un tableau.	50
	Déclarer un pointeur sur une fonction	54
	Passer un objet en paramètre par pointeur/référence .	57
 4	 Classes et objets	 59
	Ajouter des données à des objets	60
	Lier des fonctions à des objets.	62
	Déterminer la visibilité de fonctions ou de données membres	64
	Expliciter une instance avec le pointeur <i>this</i>	68
	Définir un constructeur/destructeur.	69
	Empêcher le compilateur de convertir une donnée en une autre	71
	Spécifier qu'une fonction membre ne modifie pas l'objet lié.	72
	Rendre une fonction/donnée membre indépendante de l'objet lié.	73
	Comprendre le changement de visibilité lors de l'héritage	74
	Comprendre les subtilités de l'héritage multiple	78
	Empêcher la duplication de données avec l'héritage virtuel	79
	Simuler un constructeur virtuel	80
	Créer un type abstrait à l'aide du polymorphisme	82
	Utiliser l'encapsulation pour sécuriser un objet.	85
	Obtenir des informations de types dynamiquement . .	86
	Transformer un objet en fonction	88

5	Templates et métaprogrammation	95
	Créer un modèle de classe réutilisable	96
	Créer une bibliothèque avec des templates	99
	Utiliser un type indirect dans un template	101
	Changer l'implémentation par défaut fournie par un template	102
	Spécialiser partiellement l'implémentation d'un template	104
	Spécialiser une fonction membre	105
	Exécuter du code à la compilation	106
	Créer des méta-opérateurs/métabranchements	108
	Avantages et inconvénients de la métaprogrammation	111
6	Gestion de la mémoire	113
	Réserver et libérer la mémoire.	113
	Redéfinir le système d'allocation mémoire	114
	Simuler une allocation d'objet à une adresse connue .	115
	Traiter un échec de réservation mémoire	116
	Désactiver le système d'exception lors de l'allocation .	119
	Optimiser l'allocation avec un pool mémoire	120
7	Exceptions.	123
	Principe.	123
	Transmettre une exception	127
	Expliciter les exceptions	129
	Utiliser ses propres implémentations des fonctions <i>terminate()</i> et <i>unexpected()</i>	131
	Utiliser les exceptions pour la gestion des ressources	132
	Exceptions de la STL	134

8	Itérateurs	137
	Les différents concepts.	138
	Comprendre les <i>iterator_traits</i>	140
	Calculer la distance entre deux itérateurs.	142
	Déplacer un itérateur vers une autre position	144
	Comprendre les itérateurs sur flux d'entrée/lecture . . .	145
	Comprendre les itérateurs sur flux de sortie/écriture. .	146
	Utiliser les itérateurs de parcours inversé	146
	Utiliser les itérateurs d'insertion.	148
	Utiliser les itérateurs d'insertion en début de conteneur .	149
	Utiliser les itérateurs d'insertion en fin de conteneur . . .	150
9	Conteneurs standard	151
	Créer un conteneur.	152
	Ajouter et supprimer dans un conteneur séquentiel . . .	153
	Parcourir un conteneur.	154
	Accéder à un élément d'un conteneur.	156
	Créer et utiliser un tableau	158
	Créer et utiliser une liste chaînée	160
	Créer et utiliser une file à double entrée	161
	Créer et utiliser une pile	163
	Créer et utiliser une queue.	164
	Créer et utiliser une queue de priorité.	165
	Créer et utiliser un ensemble	166
	Créer et utiliser une table associative.	167
	Créer et utiliser une table de hachage	170
	Connaître la complexité des fonctions membres des conteneurs.	173
10	Chaînes de caractères	177
	Créer une chaîne	178
	Connaître la longueur d'une chaîne	180

Comparer des chaînes	180
Échanger le contenu de deux chaînes.	181
Rechercher une sous-chaîne.	182
Extraire une sous-chaîne.	184
Remplacer une partie d'une chaîne	185
Insérer dans une chaîne	187
Concaténer des chaînes	188
Effacer une partie d'une chaîne	189
Lire des lignes dans un flux	190
11 Fichiers et flux	193
Ouvrir un fichier	194
Tester l'état d'un flux.	195
Lire dans un fichier.	197
Écrire dans un fichier.	200
Se déplacer dans un flux.	201
Manipuler des flux	202
Manipuler une chaîne de caractères comme un flux . .	205
Écrire dans une chaîne de caractère comme dans un flux	206
Lire le contenu d'une chaîne comme avec un flux . . .	206
12 Algorithmes standard	209
Calculer la somme des éléments d'une séquence.	214
Calculer les différences entre éléments consécutifs d'une séquence	215
Chercher la première occurrence de deux éléments consécutifs identiques	217
Rechercher un élément dans une séquence.	218
Copier les éléments d'une séquence dans une autre . .	219
Copier les éléments d'une séquence dans une autre en commençant par la fin	221

Copier les n premiers éléments d'une séquence dans une autre	222
Compter le nombre d'éléments correspondant à une valeur donnée	223
Compter le nombre d'éléments conformes à un test donné	224
Tester si deux séquences sont identiques	225
Chercher la sous-séquence d'éléments tous égaux à un certain élément.	226
Initialiser une séquence	228
Chercher le premier élément tel que...	228
Chercher le premier élément parmi...	229
Appliquer une fonction/foncteur sur tous les éléments d'une séquence.	230
Initialiser une séquence à l'aide d'un générateur de valeurs.	231
Tester si tous les éléments d'une séquence sont dans une autre	232
Calculer le produit intérieur (produit scalaire généralisé) de deux séquences.	234
Initialiser les éléments d'une séquence avec une valeur (en l'incrémentant).	235
Transformer une séquence en tas et l'utiliser	236
Comparer lexicographiquement deux séquences	238
Chercher le premier/dernier endroit où insérer une valeur sans briser l'ordre d'une séquence.	241
Fusionner deux séquences triées	243
Récupérer le plus petit/grand élément	245
Récupérer le plus petit/grand élément d'une séquence	246
Trouver le premier endroit où deux séquences diffèrent	247
Générer la prochaine plus petite/grande permutation lexicographique d'une séquence	248

Faire en sorte que le n ième élément soit le même que si la séquence était triée	250
Trier les n premiers éléments d'une séquence.	251
Copier les n plus petits éléments d'une séquence.	252
Calculer une somme partielle généralisée d'une séquence.	253
Couper la séquence en deux en fonction d'un prédicat	254
Calculer x^i (fonction puissance généralisée)	256
Copier aléatoirement un échantillon d'une séquence . . .	257
Copier aléatoirement un sous-échantillon (de n éléments), en préservant leur ordre d'origine . . .	258
Mélanger les éléments d'une séquence	259
Supprimer certains éléments d'une séquence.	260
Copier une séquence en omettant certains éléments. .	262
Remplacer certains éléments d'une séquence	263
Inverser l'ordre de la séquence	264
Effectuer une rotation des éléments de la séquence . .	264
Chercher une sous-séquence	265
Construire la différence de deux séquences triées . . .	268
Construire l'intersection de deux séquences triées . . .	270
Construire la différence symétrique des deux séquences triées	272
Construire l'union de deux séquences triées	273
Trier une séquence	274
Échanger le contenu de deux variables, itérateurs ou séquences	275
Transformer une (ou deux) séquences en une autre . .	276
Supprimer les doublons d'une séquence (dans ou lors d'une copie)	278
Copier à l'aide du constructeur par copie	281
Initialiser à l'aide du constructeur par copie	282

13 BOOST	285
Mettre en forme des arguments selon une chaîne de formatage	286
Convertir une donnée en chaîne de caractères	289
Construire et utiliser une expression régulière	291
Éviter les pertes mémoire grâce aux pointeurs intelligents	295
Créer des unions de types sécurisées	300
Parcourir un conteneur avec <i>BOOST_FOREACH</i>	304
Générer des messages d'erreur pendant le processus de compilation	306
14 Programmation multithread avec QT	311
Créer un thread	311
Partager des ressources	312
Se protéger contre l'accès simultané à une ressource avec les mutex	313
Contrôler le nombre d'accès simultanés à une ressource avec les sémaphores	316
Prévenir le compilateur de l'utilisation d'une variable dans plusieurs threads	318
15 Base de données	319
Savoir quand utiliser SQLite	321
Créer et utiliser une base avec l'interpréteur SQLite	328
Créer/ouvrir une base (SQLite)	331
Lancer une requête avec SQLite	335
Fermer une base (SQLite)	337
Créer une table (requête SQL)	338
Accéder aux données d'une table (requête SQL)	347
Définir un environnement ODBC (wxWidgets)	349
Se connecter à une base (wxWidgets)	350
Créer la définition de la table et l'ouvrir (wxWidgets)	352

Utiliser la table (wxWidgets).	355
Fermer la table (wxWidgets).	356
Fermer la connexion à la base (wxWidgets).	357
Libérer l'environnement ODBC (wxWidgets)	358
16 XML	359
Charger un fichier XML	360
Manipuler des données XML.	363

Annexes

A. Bibliothèques et compilateurs.	369
Compilateurs	369
IDE et RAD	370
Bibliothèques.	372
Bibliothèques à dominante graphique.	379
Utilitaires	382
B Les ajouts de la future norme C++ (C++0x)	385
Variable locale à un thread	386
Unicode et chaînes littérales.	386
Délégation de construction	388
Héritage des constructeurs	389
Constructeur et destructeur par défaut	392
union étendue	394
Opérateurs de conversion explicites.	395
Type énumération fort	395
Listes d'initialisation	396
Initialisation uniformisée	397
Type automatique	399
Boucle <i>for</i> ensembliste.	400
Syntaxe de fonction unifiée	401
Concepts	402

Autorisation de <i>sizeof</i> sur des membres.	406
Amélioration de la syntaxe pour l'utilisation des templates.	407
Template externe.	407
Expressions constantes généralisées	408
Les <i>variadic templates</i>	410
Pointeur nul	411
Tuple	412
Lambda fonctions et lambda expressions	414
 C Conventions d'appel x86	 417
Convention d'appel <i>cdecl</i>	419
Convention d'appel <i>pascal</i>	421
Convention d'appel <i>stdcall</i>	422
Convention d'appel <i>fastcall</i>	422
Convention d'appel <i>thiscall</i>	423
 Index	 425