

目次

第I部 はじめに	1
<hr/>	
第1章 本書の読み進め方	3
<hr/>	
1.1 本書の構成	3
1.1.1 はじめに	4
1.1.2 基本機能	4
1.1.3 抽象化のメカニズム	5
1.1.4 標準ライブラリ	6
1.1.5 サンプルコードと参照	8
1.2 C++ の設計	9
1.2.1 プログラミングスタイル	11
1.2.2 型チェック	15
1.2.3 C言語との互換性	16
1.2.4 言語とライブラリとシステム	17
1.3 C++ の学習	19
1.3.1 C++ プログラミング	21
1.3.2 C++ プログラマへの提言	22
1.3.3 C言語プログラマへの提言	23
1.3.4 Java プログラマへの提言	24
1.4 歴史	25
1.4.1 時系列	26
1.4.2 黎明期	27
1.4.3 1998年版の標準	29
1.4.4 2011年版の標準	32
1.4.5 C++ はどのように使われているのか	35
1.5 アドバイス	37
1.6 参考文献	38
<hr/>	
第2章 C++ を探検しよう：基本	45
<hr/>	
2.1 はじめに	45
2.2 基本	46
2.2.1 Hello, World!	47
2.2.2 型と変数と算術演算	48
2.2.3 定数	50
2.2.4 条件評価と繰返し	51
2.2.5 ポインタと配列と繰返し	52
2.3 ユーザ定義型	55
2.3.1 構造体	55
2.3.2 クラス	56
2.3.3 列挙体	58
2.4 モジュール性	59
2.4.1 分割コンパイル	60
2.4.2 名前空間	62
2.4.3 エラー処理	62
2.5 まとめ	66
2.6 アドバイス	66

第3章 C++ を探検しよう：抽象化のメカニズム	67
3.1 はじめに	67
3.2 クラス	68
3.2.1 具象型	68
3.2.2 抽象型	73
3.2.3 仮想関数	75
3.2.4 クラス階層	76
3.3 コピーとムーブ	80
3.3.1 コンテナのコピー	81
3.3.2 コンテナのムーブ	82
3.3.3 資源管理	84
3.3.4 演算子の抑制	85
3.4 テンプレート	86
3.4.1 パラメータ化された型	86
3.4.2 関数テンプレート	87
3.4.3 関数オブジェクト	88
3.4.4 可変個引数テンプレート	90
3.4.5 別名	91
3.5 アドバイス	92
第4章 C++ を探検しよう：コンテナとアルゴリズム	95
4.1 ライブラリ	95
4.1.1 標準ライブラリ概要	96
4.1.2 標準ライブラリヘッダと名前空間	97
4.2 文字列	98
4.3 ストリーム入出力	99
4.3.1 出力	100
4.3.2 入力	101
4.3.3 ユーザ定義型の入出力	102
4.4 コンテナ	103
4.4.1 vector	103
4.4.2 list	106
4.4.3 map	107
4.4.4 unordered_map	108
4.4.5 コンテナのまとめ	109
4.5 アルゴリズム	110
4.5.1 反復子の利用	111
4.5.2 反復子の型	113
4.5.3 ストリーム反復子	114
4.5.4 述語	116
4.5.5 アルゴリズムのまとめ	117
4.5.6 コンテナアルゴリズム	117
4.6 アドバイス	118
第5章 C++ を探検しよう：並行処理とユーティリティ	119
5.1 はじめに	119
5.2 資源管理	119
5.2.1 unique_ptrとshared_ptr	120

5.3 並行処理	122
5.3.1 タスクと thread	123
5.3.2 引数の受渡し	124
5.3.3 結果の返却	124
5.3.4 データの共有	125
5.3.5 タスク間通信	128
5.4 小規模ユーティリティ	131
5.4.1 時間	131
5.4.2 型関数	132
5.4.3 pairとtuple	134
5.5 正規表現	136
5.6 数学ライブラリ	136
5.6.1 数学関数とアルゴリズム	136
5.6.2 複素数	137
5.6.3 乱数	137
5.6.4 ベクタの算術演算	139
5.6.5 数値の限界値	139
5.7 アドバイス	140

第II部 基本機能 141

第6章 型と宣言	143
6.1 ISO C++ 標準	143
6.1.1 処理系	145
6.1.2 基本ソース文字セット	145
6.2 型	146
6.2.1 基本型	146
6.2.2 論理型	147
6.2.3 文字型	148
6.2.4 整数型	153
6.2.5 浮動小数点数型	155
6.2.6 接頭語と接尾語	156
6.2.7 void	157
6.2.8 大きさ	157
6.2.9 アラインメント	160
6.3 宣言	160
6.3.1 宣言の構造	162
6.3.2 複数の名前の宣言	164
6.3.3 名前	164
6.3.4 スコープ	166
6.3.5 初期化	168
6.3.6 型の導出: autoとdecltype()	172
6.4 オブジェクトと値	175
6.4.1 左辺値と右辺値	175
6.4.2 オブジェクトの生存期間	176
6.5 型別名	177
6.6 アドバイス	178

第7章 ポインタと配列と参照	179
7.1 はじめに	179
7.2 ポインタ	179
7.2.1 void*	180
7.2.2 nullptr	181
7.3 配列	182
7.3.1 配列の初期化	183
7.3.2 文字列リテラル	184
7.4 配列の内部を指すポインタ	188
7.4.1 配列の操作	189
7.4.2 多次元配列	191
7.4.3 配列の受渡し	192
7.5 ポインタと const	194
7.6 ポインタと所有権	196
7.7 参照	197
7.7.1 左辺値参照	198
7.7.2 右辺値参照	201
7.7.3 参照への参照	204
7.7.4 ポインタと参照	204
7.8 アドバイス	207
第8章 構造体と共用体と列挙体	209
8.1 はじめに	209
8.2 構造体	209
8.2.1 struct のレイアウト	211
8.2.2 struct の名前	212
8.2.3 構造体とクラス	213
8.2.4 構造体と配列	215
8.2.5 型の等価性	217
8.2.6 POD	217
8.2.7 フィールド	219
8.3 共用体	220
8.3.1 共用体とクラス	222
8.3.2 無名共用体	223
8.4 列挙体	226
8.4.1 enum class	226
8.4.2 単なる enum	229
8.4.3 名前無し enum	231
8.5 アドバイス	231
第9章 文	233
9.1 はじめに	233
9.2 文の概要	233
9.3 文としての宣言	235
9.4 選択文	236
9.4.1 if 文	236
9.4.2 switch 文	238
9.4.3 条件内の宣言	240

9.5 繰返し文	241
9.5.1 範囲 for 文	241
9.5.2 for 文	242
9.5.3 while 文	243
9.5.4 do 文	244
9.5.5 ループの終了	244
9.6 goto 文	245
9.7 コメントとインデント	246
9.8 アドバイス	247

第10章 式 249

10.1 はじめに	249
10.2 電卓プログラム	249
10.2.1 パーサ	250
10.2.2 入力	254
10.2.3 低水準の入力	258
10.2.4 エラー処理	259
10.2.5 ドライバ	260
10.2.6 ヘッダ	261
10.2.7 コマンドライン引数	261
10.2.8 スタイルについて一言	263
10.3 演算子の概要	263
10.3.1 演算結果	267
10.3.2 評価順序	268
10.3.3 演算子の優先順位	269
10.3.4 一時オブジェクト	270
10.4 定数式	271
10.4.1 シンボル定数	273
10.4.2 定数式中の const	273
10.4.3 リテラル型	274
10.4.4 参照引数	275
10.4.5 アドレス定数式	276
10.5 暗黙の型変換	276
10.5.1 格上げ	276
10.5.2 変換	277
10.5.3 通常の算術変換	279
10.6 アドバイス	280

第11章 主要な演算子 281

11.1 いろいろな演算子	281
11.1.1 論理演算子	281
11.1.2 ビット単位の論理演算子	281
11.1.3 条件式	283
11.1.4 インクリメントとデクリメント	283
11.2 空き領域	285
11.2.1 メモリ管理	287
11.2.2 配列	289
11.2.3 メモリ領域の割当て	290
11.2.4 new の多重定義	291

11.3	並び	293
11.3.1	実装モデル	294
11.3.2	修飾並び	295
11.3.3	非修飾並び	296
11.4	ラムダ式	298
11.4.1	実装モデル	298
11.4.2	ラムダ式への変形	299
11.4.3	キャプチャ	301
11.4.4	呼出しとリターン	305
11.4.5	ラムダ式の型	305
11.5	明示的型変換	306
11.5.1	構築	307
11.5.2	名前付きキャスト	309
11.5.3	C言語形式キャスト	310
11.5.4	関数形式キャスト	310
11.6	アドバイス	311

第12章 関数 313

12.1	関数宣言	313
12.1.1	なぜ関数なのか?	314
12.1.2	関数宣言の構成要素	314
12.1.3	関数定義	315
12.1.4	値の返却	316
12.1.5	inline 関数	318
12.1.6	constexpr 関数	319
12.1.7	[[noreturn]] 関数	322
12.1.8	局所変数	322
12.2	引数の受渡し	323
12.2.1	参照引数	323
12.2.2	配列の引数	326
12.2.3	並び引数	327
12.2.4	可変個引数	328
12.2.5	デフォルト引数	332
12.3	関数多重定義	333
12.3.1	多重定義の自動解決	333
12.3.2	多重定義と返却型	335
12.3.3	多重定義とスコープ	336
12.3.4	複数引数の解決	336
12.3.5	多重定義の手動解決	337
12.4	事前条件と事後条件	338
12.5	関数へのポインタ	339
12.6	マクロ	343
12.6.1	条件コンパイル	346
12.6.2	定義済みマクロ	347
12.6.3	プラグマ	348
12.7	アドバイス	348

第13章 例外処理 351

13.1	エラー処理	351
13.1.1	例外	352

13.1.2	従来のエラー処理	354
13.1.3	ごまかしの対処	355
13.1.4	例外のもう一つの顔	356
13.1.5	例外を使えないとき	357
13.1.6	階層的エラー処理	358
13.1.7	例外と効率	360
13.2	例外安全性の保証	361
13.3	資源管理	363
13.3.1	finally	366
13.4	不変条件の強制	368
13.5	例外の送出と捕捉	372
13.5.1	例外の送出	372
13.5.2	例外の捕捉	376
13.5.3	例外とスレッド	383
13.6	vectorの実装	383
13.6.1	単純なvector	384
13.6.2	メモリ処理の分離	388
13.6.3	代入	390
13.6.4	要素数の変更	392
13.7	アドバイス	395

第14章 名前空間 397

14.1	構成上の問題	397
14.2	名前空間	398
14.2.1	明示的修飾	400
14.2.2	using 宣言	401
14.2.3	using 指令	402
14.2.4	実引数依存探索	403
14.2.5	名前空間はオープン	405
14.3	モジュール化とインタフェース	406
14.3.1	モジュールとしての名前空間	407
14.3.2	実装	409
14.3.3	インタフェースと実装	411
14.4	名前空間を用いた構成	412
14.4.1	利便性と安全性	412
14.4.2	名前空間別名	413
14.4.3	名前空間の合成	414
14.4.4	合成と選択	415
14.4.5	名前空間と多重定義	416
14.4.6	バージョン管理	418
14.4.7	入れ子の名前空間	420
14.4.8	名前無し名前空間	421
14.4.9	C言語のヘッダ	421
14.5	アドバイス	423

第15章 ソースファイルとプログラム 425

15.1	分割コンパイル	425
15.2	結合	426
15.2.1	ファイル内局所名	429
15.2.2	ヘッダ	429

15.2.3	単一定義則	432
15.2.4	標準ライブラリヘッダ	434
15.2.5	C++ 以外のコードとの結合	434
15.2.6	結合と関数へのポインタ	436
15.3	ヘッダの利用	437
15.3.1	単一ヘッダ構成	437
15.3.2	複数ヘッダ構成	441
15.3.3	インクルードガード	445
15.4	プログラム	446
15.4.1	非局所変数の初期化	447
15.4.2	初期化と並行処理	448
15.4.3	プログラムの終了	449
15.5	アドバイス	450

第Ⅲ部 抽象化のメカニズム 451

第 16 章 クラス 453

16.1	はじめに	453
16.2	クラスの基礎	454
16.2.1	メンバ関数	455
16.2.2	デフォルトのコピー	456
16.2.3	アクセス制御	456
16.2.4	class と struct	458
16.2.5	コンストラクタ	459
16.2.6	explicit コンストラクタ	461
16.2.7	クラス内初期化子	463
16.2.8	クラス内関数定義	464
16.2.9	変更可能性	464
16.2.10	自己参照	467
16.2.11	メンバアクセス	469
16.2.12	static メンバ	470
16.2.13	メンバ型	472
16.3	具象クラス	473
16.3.1	メンバ関数	476
16.3.2	ヘルパ関数	478
16.3.3	演算子の多重定義	480
16.3.4	具象クラス的重要性	480
16.4	アドバイス	482

第 17 章 構築と後始末とコピーとムーブ 483

17.1	はじめに	483
17.2	コンストラクタとデストラクタ	485
17.2.1	コンストラクタと不変条件	486
17.2.2	デストラクタと資源	487
17.2.3	基底とメンバのデストラクタ	488
17.2.4	コンストラクタとデストラクタの呼出し	489
17.2.5	virtual デストラクタ	490
17.3	クラスオブジェクトの初期化	491
17.3.1	コンストラクタがない場合の初期化	491

17.3.2	コンストラクタによる初期化	492
17.3.3	デフォルトコンストラクタ	495
17.3.4	初期化子並びコンストラクタ	497
17.4	メンバと基底の初期化	501
17.4.1	メンバの初期化	501
17.4.2	基底初期化子	503
17.4.3	委譲コンストラクタ	504
17.4.4	クラス内初期化子	505
17.4.5	static メンバの初期化	507
17.5	コピーとムーブ	508
17.5.1	コピー	508
17.5.2	ムーブ	515
17.6	デフォルト演算の生成	519
17.6.1	明示的なデフォルト	519
17.6.2	デフォルト演算	521
17.6.3	デフォルト演算の利用	521
17.6.4	関数の delete	525
17.7	アドバイス	527

第 18 章 演算子の多重定義 529

18.1	はじめに	529
18.2	演算子関数	530
18.2.1	単項演算子と2項演算子	532
18.2.2	演算子本来の意味	533
18.2.3	演算子とユーザ定義型	533
18.2.4	オブジェクトのやりとり	534
18.2.5	名前空間内の演算子	535
18.3	複素数型	537
18.3.1	メンバ演算子と非メンバ演算子	538
18.3.2	混合算術演算	539
18.3.3	変換	539
18.3.4	リテラル	542
18.3.5	アクセッサ関数	543
18.3.6	ヘルパ関数	544
18.4	型変換	545
18.4.1	変換演算子	545
18.4.2	explicit 変換演算子	547
18.4.3	曖昧さ	547
18.5	アドバイス	549

第 19 章 特殊な演算子 551

19.1	はじめに	551
19.2	特殊な演算子	551
19.2.1	添字演算	551
19.2.2	関数呼出し	552
19.2.3	参照外し	554
19.2.4	インクリメントとデクリメント	556
19.2.5	メモリ確保と解放	558
19.2.6	ユーザ定義リテラル	559
19.3	文字列クラス String	562

19.3.1	基本演算	563
19.3.2	文字へのアクセス	564
19.3.3	内部表現	565
19.3.4	メンバ関数	567
19.3.5	ヘルパ関数	570
19.3.6	作成した文字列クラスの利用例	572
19.4	フレンド	572
19.4.1	フレンドの探索	574
19.4.2	フレンドとメンバ	575
19.5	アドバイス	577

第20章 派生クラス 579

20.1	はじめに	579
20.2	派生クラス	580
20.2.1	メンバ関数	583
20.2.2	コンストラクタとデストラクタ	584
20.3	クラス階層	585
20.3.1	型フィールド	585
20.3.2	仮想関数	587
20.3.3	明示的修飾	590
20.3.4	オーバーライド制御	591
20.3.5	基底メンバの using 宣言	595
20.3.6	返却型緩和	597
20.4	抽象クラス	599
20.5	アクセス制御	601
20.5.1	protected メンバ	604
20.5.2	基底クラスへのアクセス	606
20.5.3	using 宣言とアクセス制御	607
20.6	メンバへのポインタ	608
20.6.1	メンバ関数へのポインタ	608
20.6.2	データメンバへのポインタ	611
20.6.3	基底のメンバと派生のメンバ	611
20.7	アドバイス	612

第21章 クラス階層 613

21.1	はじめに	613
21.2	クラス階層の設計	613
21.2.1	実装継承	614
21.2.2	インタフェース継承	617
21.2.3	別の実装	620
21.2.4	オブジェクト作成の局所化	623
21.3	多重継承	624
21.3.1	インタフェースの多重継承	625
21.3.2	多重実装クラス	625
21.3.3	曖昧さの解決	627
21.3.4	基底クラスの反復	630
21.3.5	仮想基底クラス	632
21.3.6	複製か仮想基底クラスか	637
21.4	アドバイス	640

第22章 実行時型情報 641

22.1	はじめに	641
22.2	クラス階層の移動	641
22.2.1	dynamic_cast	643
22.2.2	多重継承	646
22.2.3	static_castとdynamic_cast	648
22.2.4	インタフェースの復元	649
22.3	ダブルディスパッチとVisitorパターン	653
22.3.1	ダブルディスパッチ	653
22.3.2	Visitorパターン	656
22.4	構築と解体	657
22.5	型の識別	658
22.5.1	拡張型情報	660
22.6	RTTIの利用と悪用	661
22.7	アドバイス	663

第23章 テンプレート 665

23.1	導入と概要	665
23.2	単純な文字列テンプレート	668
23.2.1	テンプレート定義	670
23.2.2	テンプレート具現化	671
23.3	型チェック	672
23.3.1	型の等価性	674
23.3.2	エラー検出	674
23.4	クラステンプレートのメンバ	676
23.4.1	データメンバ	676
23.4.2	メンバ関数	676
23.4.3	メンバ型別名	677
23.4.4	static メンバ	677
23.4.5	メンバ型	678
23.4.6	メンバテンプレート	678
23.4.7	フレンド	683
23.5	関数テンプレート	684
23.5.1	関数テンプレートの引数	686
23.5.2	関数テンプレートの引数の導出	687
23.5.3	関数テンプレートの多重定義	689
23.6	テンプレート別名	694
23.7	ソースコードの構成	695
23.7.1	結合	697
23.8	アドバイス	698

第24章 ジェネリックプログラミング 699

24.1	はじめに	699
24.2	アルゴリズムとリフティング	700
24.3	コンセプト	704
24.3.1	コンセプトの特定	704
24.3.2	コンセプトと制約	708
24.4	コンセプトの具象化	710

24.4.1	公理	713
24.4.2	複数引数のコンセプト	714
24.4.3	値コンセプト	715
24.4.4	制約判定	716
24.4.5	テンプレート定義判定	717
24.5	アドバイス	720
第 25 章 特殊化		721
25.1	はじめに	721
25.2	テンプレートの仮引数と実引数	722
25.2.1	引数としての型	722
25.2.2	引数としての値	723
25.2.3	引数としての処理	725
25.2.4	引数としてのテンプレート	727
25.2.5	デフォルトのテンプレート引数	728
25.3	特殊化	730
25.3.1	インタフェースの特殊化	732
25.3.2	一次テンプレート	734
25.3.3	特殊化の順番	736
25.3.4	関数テンプレートの特殊化	736
25.4	アドバイス	738
第 26 章 具現化		741
26.1	はじめに	741
26.2	テンプレート具現化	742
26.2.1	具現化はいつ必要となるのか?	743
26.2.2	具現化の手動制御	744
26.3	名前バインド	745
26.3.1	従属名	747
26.3.2	定義位置でのバインド	748
26.3.3	具現化位置でのバインド	749
26.3.4	複数の具現化位置	751
26.3.5	テンプレートと名前空間	753
26.3.6	過剰な ADL	753
26.3.7	基底クラス内の名前	755
26.4	アドバイス	758
第 27 章 テンプレートと階層		759
27.1	はじめに	759
27.2	パラメータ化と階層	760
27.2.1	型の生成	762
27.2.2	テンプレートの変換	764
27.3	クラステンプレートの階層	765
27.3.1	インタフェースとしてのテンプレート	767
27.4	基底クラスとしてのテンプレート引数	767
27.4.1	データ構造の組立て	767
27.4.2	クラス階層の線形化	771
27.5	アドバイス	776

第 28 章 メタプログラミング		779
28.1	はじめに	779
28.2	型関数	781
28.2.1	型別名	784
28.2.2	型述語	786
28.2.3	関数の選択	787
28.2.4	特性	788
28.3	制御構造	790
28.3.1	選択	790
28.3.2	繰返しと再帰	793
28.3.3	メタプログラミングを利用すべき場面	794
28.4	条件付き定義: Enable_if	795
28.4.1	Enable_if の利用	797
28.4.2	Enable_if の実装	799
28.4.3	Enable_if とコンセプト	799
28.4.4	Enable_if の利用例	800
28.5	コンパイル時リスト: Tuple	803
28.5.1	単純な出力関数	805
28.5.2	要素アクセス	806
28.5.3	make_tuple	808
28.6	可変個引数テンプレート	809
28.6.1	型安全な printf()	809
28.6.2	技術的詳細	812
28.6.3	転送	813
28.6.4	標準ライブラリの tuple	815
28.7	SI 単位系の例題	818
28.7.1	Unit	818
28.7.2	Quantity	819
28.7.3	Unit リテラル	821
28.7.4	ユーティリティ関数	822
28.8	アドバイス	824
第 29 章 行列の設計		825
29.1	はじめに	825
29.1.1	基本的な Matrix の使い方	825
29.1.2	Matrix の要件	827
29.2	Matrix テンプレート	828
29.2.1	構築と代入	830
29.2.2	添字演算とスライシング	831
29.3	Matrix の算術演算	833
29.3.1	スカラー演算	834
29.3.2	行列の加算	835
29.3.3	乗算	836
29.4	Matrix の実装	838
29.4.1	slice()	838
29.4.2	Matrix のスライス	838
29.4.3	Matrix_ref	840
29.4.4	初期化子並びによる Matrix の初期化	841
29.4.5	Matrix のアクセス	843
29.4.6	ゼロ次元の Matrix	845

29.5 線形方程式の解	846
29.5.1 古典的なガウスの消去法	847
29.5.2 ピボット	848
29.5.3 動作確認	849
29.5.4 複合演算	850
29.6 アドバイス	852

第IV部 標準ライブラリ 855

第30章 標準ライブラリの概要 857

30.1 はじめに	857
30.1.1 標準ライブラリの機能	858
30.1.2 設計上の制約	859
30.1.3 解説方針	860
30.2 ヘッダ	861
30.3 言語の支援	865
30.3.1 initializer_list の支援	866
30.3.2 範囲 for 文の支援	866
30.4 エラー処理	867
30.4.1 例外	867
30.4.2 アサーション	872
30.4.3 system_error	872
30.5 アドバイス	882

第31章 STL コンテナ 885

31.1 導入	885
31.2 コンテナの概要	885
31.2.1 コンテナの内部表現	888
31.2.2 要素の要件	890
31.3 処理の概要	893
31.3.1 メンバ型	896
31.3.2 コンストラクタとデストラクタと代入	897
31.3.3 要素数と容量	898
31.3.4 反復子	899
31.3.5 要素アクセス	900
31.3.6 スタック処理	900
31.3.7 リスト処理	901
31.3.8 その他の処理	902
31.4 コンテナ	903
31.4.1 vector	903
31.4.2 リスト	907
31.4.3 連想コンテナ	910
31.5 コンテナアダプタ	921
31.5.1 stack	921
31.5.2 queue	923
31.5.3 priority_queue	923
31.6 アドバイス	924

第32章 STL アルゴリズム 927

32.1 はじめに	927
32.2 アルゴリズム	927
32.2.1 シーケンス	928
32.3 ポリシー引数	930
32.3.1 計算量	931
32.4 シーケンスを更新しないアルゴリズム	932
32.4.1 for_each()	932
32.4.2 シーケンス述語	932
32.4.3 count()	933
32.4.4 find()	933
32.4.5 equal() と mismatch()	934
32.4.6 search()	935
32.5 シーケンスを更新するアルゴリズム	936
32.5.1 copy()	936
32.5.2 unique()	937
32.5.3 remove() と reverse() と replace()	938
32.5.4 rotate() と random_shuffle() と partition()	939
32.5.5 順列	940
32.5.6 fill()	941
32.5.7 swap()	942
32.6 ソートと探索	943
32.6.1 2分探索	946
32.6.2 merge()	947
32.6.3 集合アルゴリズム	947
32.6.4 ヒープ	949
32.6.5 lexicographical_compare()	950
32.7 最小値と最大値	950
32.8 アドバイス	952

第33章 STL 反復子 953

33.1 はじめに	953
33.1.1 反復子モデル	953
33.1.2 反復子カテゴリ	955
33.1.3 反復子の特性	956
33.1.4 反復子の処理	959
33.2 反復子アダプタ	960
33.2.1 逆進反復子	960
33.2.2 挿入反復子	962
33.2.3 ムーブ反復子	964
33.3 範囲アクセス関数	964
33.4 関数オブジェクト	965
33.5 関数アダプタ	966
33.5.1 bind()	967
33.5.2 mem_fn()	968
33.5.3 function	969
33.6 アドバイス	971

第 34 章 メモリと資源	973
34.1 はじめに	973
34.2 “コンテナ相当”	973
34.2.1 array	974
34.2.2 bitset	977
34.2.3 vector<bool>	981
34.2.4 タプル	982
34.3 資源管理ポインタ	986
34.3.1 unique_ptr	987
34.3.2 shared_ptr	990
34.3.3 weak_ptr	994
34.4 アロケータ	996
34.4.1 デフォルトアロケータ	997
34.4.2 アロケータの特性	999
34.4.3 ポインタの特性	1000
34.4.4 スコープ付きアロケータ	1000
34.5 ガーベジコレクションインタフェース	1002
34.6 未初期化メモリ	1005
34.6.1 一時バッファ	1006
34.6.2 raw_storage_iterator	1006
34.7 アドバイス	1007
第 35 章 ユーティリティ	1009
35.1 はじめに	1009
35.2 時刻	1009
35.2.1 duration	1010
35.2.2 time_point	1013
35.2.3 クロック	1015
35.2.4 時間特性	1016
35.3 コンパイル時の有理数演算	1017
35.4 型関数	1018
35.4.1 型特性	1018
35.4.2 型生成器	1023
35.5 小規模なユーティリティ	1028
35.5.1 move()とforward()	1028
35.5.2 swap()	1029
35.5.3 関係演算子	1029
35.5.4 比較演算とtype_infoのハッシュ演算	1030
35.6 アドバイス	1031
第 36 章 文字列	1033
36.1 はじめに	1033
36.2 文字クラス	1033
36.2.1 文字クラス判定関数	1033
36.2.2 文字特性	1034
36.3 文字列	1036
36.3.1 stringとC言語スタイルの文字列	1037
36.3.2 コンストラクタ	1038

36.3.3 基本演算	1040
36.3.4 文字列の入出力	1042
36.3.5 数値変換	1042
36.3.6 STLライクな処理	1044
36.3.7 findファミリ	1046
36.3.8 部分文字列	1048
36.4 アドバイス	1049

第 37 章 正規表現 1051

37.1 正規表現	1051
37.1.1 正規表現の表記	1052
37.2 regex	1056
37.2.1 照合結果	1059
37.2.2 書式化	1062
37.3 正規表現の関数	1063
37.3.1 regex_match()	1063
37.3.2 regex_search()	1065
37.3.3 regex_replace()	1065
37.4 正規表現の反復子	1067
37.4.1 regex_iterator	1067
37.4.2 regex_token_iterator	1069
37.5 regex_traits	1071
37.6 アドバイス	1072

第 38 章 入出力ストリーム 1073

38.1 はじめに	1073
38.2 入出力ストリームの階層	1075
38.2.1 ファイルストリーム	1076
38.2.2 文字列ストリーム	1078
38.3 エラー処理	1080
38.4 入出力処理	1081
38.4.1 入力処理	1081
38.4.2 出力処理	1085
38.4.3 操作子	1087
38.4.4 ストリームの状態	1088
38.4.5 書式化	1092
38.5 ストリーム反復子	1099
38.6 バッファリング	1100
38.6.1 出力ストリームとバッファ	1104
38.6.2 入力ストリームとバッファ	1105
38.6.3 バッファ反復子	1106
38.7 アドバイス	1108

第 39 章 ロケール 1111

39.1 文化的な違いの取扱い	1111
39.2 locale クラス	1114
39.2.1 名前付き locale	1116
39.2.2 string の比較	1120

39.3	facet クラス	1121
39.3.1	locale 内 facet へのアクセス	1122
39.3.2	単純なユーザ定義 facet	1122
39.3.3	locale と facet の利用	1125
39.4	標準 facet	1126
39.4.1	string の比較	1128
39.4.2	数値の書式化	1131
39.4.3	金額の書式化	1136
39.4.4	日付と時刻の書式化	1142
39.4.5	文字クラス	1144
39.4.6	文字コードの変換	1148
39.4.7	メッセージ	1152
39.5	便利なインタフェース	1155
39.5.1	文字クラス	1155
39.5.2	文字の変換	1156
39.5.3	文字列の変換	1156
39.5.4	バッファの変換	1158
39.6	アドバイス	1159

第 40 章 数値演算 1161

40.1	はじめに	1161
40.2	数値の限界値	1162
40.2.1	限界値マクロ	1164
40.3	標準数学関数	1164
40.4	複素数	1166
40.5	数値配列: valarray	1167
40.5.1	コンストラクタと代入	1168
40.5.2	添字演算	1170
40.5.3	演算	1171
40.5.4	slice	1173
40.5.5	slice_array	1176
40.5.6	汎用のスライス	1176
40.6	汎用数値アルゴリズム	1178
40.6.1	accumulate()	1178
40.6.2	inner_product()	1179
40.6.3	partial_sum() と adjacent_difference()	1180
40.6.4	iota()	1181
40.7	乱数	1181
40.7.1	乱数エンジン	1184
40.7.2	乱数デバイス	1186
40.7.3	分布	1186
40.7.4	C 言語スタイルの乱数	1190
40.8	アドバイス	1190

第 41 章 並行処理 1193

41.1	導入	1193
41.2	メモリモデル	1195
41.2.1	メモリロケーション	1196
41.2.2	命令の順序の変更	1197
41.2.3	メモリオーダ	1198

41.2.4	データ競合	1199
41.3	アトミック性	1201
41.3.1	atomic 型	1203
41.3.2	フラグとフェンス	1208
41.4	volatile	1210
41.5	アドバイス	1210

第 42 章 スレッドとタスク 1213

42.1	はじめに	1213
42.2	スレッド	1213
42.2.1	識別	1215
42.2.2	構築	1216
42.2.3	解体	1217
42.2.4	join()	1218
42.2.5	detach()	1219
42.2.6	this_thread 名前空間	1221
42.2.7	thread の強制終了	1222
42.2.8	thread_local データ	1222
42.3	データ競合の回避	1224
42.3.1	mutex	1224
42.3.2	複数のロック	1233
42.3.3	call_once()	1234
42.3.4	条件変数	1235
42.4	タスクベースの並行処理	1241
42.4.1	future と promise	1241
42.4.2	promise	1242
42.4.3	packaged_task	1243
42.4.4	future	1247
42.4.5	shared_future	1249
42.4.6	async()	1250
42.4.7	並列 find() の具体例	1253
42.5	アドバイス	1256

第 43 章 標準 C ライブラリ 1259

43.1	はじめに	1259
43.2	ファイル	1259
43.3	printf() ファミリ	1260
43.4	C 言語スタイルの文字列	1265
43.5	メモリ	1266
43.6	日付と時刻	1267
43.7	その他	1270
43.8	アドバイス	1271

第 44 章 互換性 1273

44.1	はじめに	1273
44.2	C++11 の新機能	1274
44.2.1	言語機能	1274
44.2.2	標準ライブラリコンポーネント	1275

44.2.3 非推奨とされた機能	1276
44.2.4 以前の C++ 処理系の利用	1277
44.3 CとC++の互換性	1278
44.3.1 C言語とC++は兄弟	1278
44.3.2 “無言の”違い	1280
44.3.3 C++ではないC言語コード	1280
44.3.4 C言語ではないC++コード	1283
44.4 アドバイス	1285
索引	1287
翻訳者後書き	1319