
Claude Delannoy
Conforme ANSI/ISO
Programmer en langage C++
Eyrolles

Programmer en langage C++

Claude DELANNOY

QUATRIÈME ÉDITION
revue et augmentée
1998

deuxième tirage 1998

Eyrolles

III. LES NOUVELLES POSSIBILITÉS D'ENTRÉES-SORTIES CONVERSATIONNELLES DE C++ 21

1. LES NOUVELLES ENTRÉES-SORTIES EN C++ 21
2. ÉCRITURE SUR LA SORTIE STANDARD 22
   2.1 Quelques exemples 22
   2.2 Les possibilités d'écriture sur cout 25
3. LECTURE SUR L'ENTRÉE STANDARD 26
   3.1 Introduction 26
   3.2 Les possibilités de lecture sur cin 27
   3.3 Exemples 28

IV. LES SPÉCIFICITÉS DU C++ 31

1. LE COMMENTAIRE DE FIN DE LIGNE 32
2. DÉCLARATIONS ET INITIALISATIONS 33
3. LA NOTION DE RÉFÉRENCE 34
   3.1 Transmission des arguments en C 34
   3.2 Transmission par référence en C++ 36
      a) Transmission d'arguments par référence 36
      b) Transmission par référence de la valeur de retour d'une fonction 38
   3.3 La référence d'une manière générale 38
      a) La notion de référence est plus générale que celle d'argument 39
      b) Initialisation de référence 39
      c) La transmission d'un argument ou d'une valeur de retour est une initialisation 40
4. LES ARGUMENTS PAR DÉFAUT 41
   4.1 Exemples 41
   4.2 D'une manière générale 43
   4.3 Lorsque l'on conjugue argument par défaut et transmission par référence 44
5. SURDÉFINITION DE FONCTIONS 46
   5.1 Mise en œuvre de la surdéfinition de fonctions 46
   5.2 Exemples de choix d'une fonction surdéfinie 48
   5.3 Règles de recherche d'une fonction surdéfinie 50
      a) Cas des fonctions à un argument 50
      b) Cas des fonctions à plusieurs arguments 51
   5.4 Le mécanisme de la surdéfinition de fonctions 52
6. LES OPÉRATEURS NEW ET DELETE 53
   6.1 Exemples d'utilisation de new 54
   6.2 Syntaxe et rôle de new 54
   6.3 Exemples d'utilisation de l'opérateur delete 55
   6.4 Syntaxe et rôle de l'opérateur delete 56
   6.5 Pour gérer les débordements de mémoire : set_new_handler 56

7. LA SPÉCIFICATION INLINE....58
7.1 Rappels concernant les macros et les fonctions....58
7.2 Utilisation de fonctions "en ligne"....59

V. CLASSES ET OBJETS....63

1. LES STRUCTURES EN C++....64
1.1 Rappel : les structures en C....64
1.2 Déclaration d'une structure comportant des fonctions membre....65
1.3 Définition des fonctions membre....66
1.4 Utilisation d'une structure comportant des fonctions membre....67
1.5 Exemple récapitulatif....69
2. NOTION DE CLASSE....70
3. AFFECTATION D'OBJETS....74
4. NOTION DE CONSTRUCTEUR ET DE DESTRUCTEUR....76
4.1 Introduction....76
4.2 Exemple de classe comportant un constructeur....77
4.3 Construction et destruction des objets....79
4.4 Rôles du constructeur et du destructeur....80
4.5 Quelques règles....83
5. LES MEMBRES DONNÉE STATIQUES....84
5.1 Le qualificatif static pour un membre donnée....84
5.2 Initialisation des membres donnée statiques....85
5.3 Exemple....86
6. EXPLOITATION D'UNE CLASSE....87
6.1 La classe, comme "composant logiciel"....87
6.3 Cas des membres donnée statiques....90
6.4 En cas de modification d'une classe....90
a) La déclaration des membres publics n'a pas changé 91
b) La déclaration des membres publics a changé 91
7. LES CLASSES EN GÉNÉRAL....91
7.1 Les autres sortes de classes en C++....91
7.2 Ce qu'on peut trouver dans la déclaration d'une classe....92
7.3 Déclaration d'une classe....93
EXERCICES....93

VI. LES PROPRIÉTÉS DES FONCTIONS MEMBRE....95

1. SURDÉFINITION DES FONCTIONS MEMBRE....96
2. ARGUMENTS PAR DÉFAUT....97
3. LES FONCTIONS MEMBRE "EN LIGNE"....99
4. CAS DES OBJETS TRANSMIS EN ARGUMENT D'UNE FONCTION MEMBRE....101

5. MODE DE TRANSMISSION DES OBJETS EN ARGUMENT 104
5.1 Transmission de l'adresse d'un objet 104
5.2 Transmission par référence 106
5.3 Les problèmes posés par la transmission par valeur 107
6. LORSQUE LA VALEUR DE RETOUR D'UNE FONCTION EST ELLE-MÊME UN OBJET 107
7. AUTO RÉFÉRENCE : LE MOT CLÉ THIS 108
8. LES FONCTIONS MEMBRE STATIQUES 110
9. LES FONCTIONS MEMBRE CONSTANTES 112
10. LES POINTEURS SUR DES FONCTIONS MEMBRE 114
EXERCICES 115

VII. CONSTRUCTION, DESTRUCTION, RECOPIE ET INITIALISATION DES OBJETS 117

1. LES OBJETS AUTOMATIQUES ET STATIQUES 118
1.1 Leur durée de vie 118
1.2 Appel des constructeurs et des destructeurs 119
1.3 Exemple 120
2. LES OBJETS DYNAMIQUES 122
2.1 Les structures dynamiques 122
2.2 Les objets dynamiques 123
a) Points communs avec les structures dynamiques 123
b) Les nouvelles possibilités des opérateurs new et delete 124
c) Exemple 125
3. LE CONSTRUCTEUR DE RECOPIE 126
3.1 Présentation du constructeur de recopie 126
3.2 Premier exemple d'utilisation du constructeur par recopie : objet transmis par valeur 128
a) Emploi du constructeur par recopie par défaut 129
b) Définition d'un constructeur par recopie 130
3.3 Deuxième exemple d'utilisation du constructeur par recopie : objet transmis en valeur de retour d'une fonction 133
4. INITIALISATION D'UN OBJET LORS DE SA DÉCLARATION 134
5. OBJETS MEMBRE 137
5.1 Introduction 137
5.2 Mise en œuvre des constructeurs et des destructeurs 138
5.3 Cas du constructeur par recopie 141
6. LES TABLEAUX D'OBJETS 141
6.1 Notations 142
6.2 Constructeurs et initialiseurs 143
6.3 Cas des tableaux dynamiques d'objets 144

7. LES OBJETS TEMPORAIRES 145
EXERCICES 148

VIII. LES FONCTIONS AMIES 151
1. EXEMPLE DE FONCTION INDÉPENDANTE AMIE D'UNE CLASSE 152
2. LES DIFFÉRENTES SITUATIONS D'AMITIÉ 155
2.1 Fonction membre d'une classe, amie d'une autre classe 155
2.2 Fonction amie de plusieurs classes 157
2.3 Toutes les fonctions d'une classe sont amies d'une autre classe 158
3. EXEMPLE 159
3.1 Fonction amie indépendante 159
3.2 Fonction amie, membre d'une classe 161
4. EXPLOITATION DE CLASSES DISPOSANT DE FONCTIONS AMIES 162

IX. LA SURDÉFINITION D'OPÉRATEURS 163
1. LE MÉCANISME DE LA SURDÉFINITION D'OPÉRATEUR 165
1.1 Surdéfinition d'opérateur avec une fonction amie 165
1.2 Surdéfinition d'opérateur avec une fonction membre 167
1.3 Opérateurs et transmission par référence 169
2. LES POSSIBILITÉS ET LES LIMITES DE LA SURDÉFINITION D'OPÉRATEUR 170
2.1 Il faut se limiter aux opérateurs existants 171
2.2 ... au contexte de classe 173
2.3 ... et ne pas faire d'hypothèse sur la "signification" d'un opérateur 173
2.4 Cas des opérateurs ++ et -- 175
2.5 Les opérateurs = et & ont une signification prédéfinie 176
2.6 Les conversions 177
2.7 Choix entre fonction membre et fonction amie 177
3. EXEMPLE DE SURDÉFINITION DE L'OPÉRATEUR = ... 178
4. NOTION DE FORME CANONIQUE D'UNE CLASSE 183
5. EXEMPLE DE SURDÉFINITION DE L'OPÉRATEUR [ ] 185
6. SURDÉFINITION DE L'OPÉRATEUR () 188
7. SURDÉFINITION DES OPÉRATEURS NEW ET DELETE 189
7.1 Surdéfinition de new et delete 189
7.2 Exemple 190
EXERCICES 192

X. LES CONVERSIONS DE TYPE DÉFINIES PAR L'UTILISATEUR 195
1. LES DIFFÉRENTES SORTES DE CONVERSIONS DÉFINIES PAR L'UTILISATEUR 196
2. L'OPÉRATEUR DE "CAST" POUR LA CONVERSION D'UN TYPE CLASSE DANS UN TYPE DE BASE 198

2.1 Définition de l'opérateur de "cast" 198
2.2 Exemple simple d'utilisation 199
2.3 Appel implicite de l'opérateur de "cast" lors d'un appel de fonction 200
2.4 Appel implicite de l'opérateur de "cast" dans l'évaluation d'une expression 202
2.5 Conversions en chaîne 204
2.6 En cas d'ambiguïté 206
3. LE CONSTRUCTEUR POUR LA CONVERSION D'UN TYPE DE BASE EN UN TYPE CLASSE 207
3.1 Un premier exemple 207
3.2 Le constructeur dans une chaîne de conversions 210
3.3 Choix entre constructeur ou opérateur d'affectation 210
3.4 Emploi d'un constructeur pour élargir la signification d'un opérateur 212
3.5 Pour interdire l'utilisation du constructeur dans des conversions implicites : le mot clé *explicit* 215
4. LES CONVERSIONS D'UN TYPE CLASSE EN UN AUTRE TYPE CLASSE 215
4.1 Exemple simple d'opérateur de "cast" 216
4.2 Exemple simple de conversion par un constructeur 217
4.3 Pour donner, dans une classe, une signification à un opérateur défini dans une autre classe 218
5. QUELQUES CONSEILS 221

XI. LES PATRONS DE FONCTIONS 223

1. EXEMPLE DE CRÉATION ET D'UTILISATION D'UN PATRON DE FONCTION 224
1.1 Création d'un patron de fonction 224
1.2 Premières utilisations de notre patron de fonction 225
1.3 D'autres utilisations de notre patron 227
a) Application au type char * 227
b) Application à un type classe 228
2. LES PARAMÈTRES DE TYPE D'UN PATRON DE FONCTIONS 229
2.1 Utilisation des paramètres de type dans la définition d'un patron 229
2.2 Identification des paramètres de type d'une fonction patron 230
2.3 Nouvelle syntaxe d'initialisation des variables des types standard 232
2.4 Limitations des patrons de fonctions 233
3. LES PARAMÈTRES EXPRESSION D'UN PATRON DE FONCTIONS 234
4. SURDÉFINITION DE PATRONS 235
4.1 Exemples de surdéfinition de patron de fonctions ne comportant que des paramètres de type 235
4.2 Exemples de surdéfinition de patron de fonctions comportant des paramètres expression 238

5. SPÉCIALISATION DE FONCTIONS DE PATRON 239
6. LES PATRONS DE FONCTIONS D'UNE MANIÈRE GÉNÉRALE 240

XII. LES PATRONS DE CLASSES 243

1. EXEMPLE DE CRÉATION ET D'UTILISATIN D'UN PATRON DE CLASSES 244
1.1 Création d'un patron de classes 244
1.2 Utilisation d'un patron de classes 246
1.3 Exemple récapitulatif 247
2. LES PARAMÈTRES DE TYPE D'UN PATRON DE CLASSES 248
2.1 Les paramètres de type dans la création d'un patron de classes 249
2.2 Instanciation d'une classe patron 249
3. LES PARAMÈTRES EXPRESSION D'UN PATRON DE CLASSES 250
3.1 Exemple 251
3.2 D'une manière générale 253
4. SPÉCIALISATION D'UN PATRON DE CLASSES 254
4.1 Exemple de spécialisation d'une fonction membre 254
4.2 D'une manière générale 255
a) On peut spécialiser pour les valeurs de tous les paramètres 255
b) On peut spécialiser une fonction membre ou une classe 256
c) Spécialisation partielle de patrons de classes 256
5. PARAMÈTRES PAR DÉFAUT 257
6. PATRONS DE FONCTIONS MEMBRE 257
7. IDENTITÉ DE CLASSES PATRON 258
8. CLASSES PATRON ET DÉCLARATIONS D'AMITIÉS 259
8.1 Déclaration de classes ou fonctions "ordinaires" amies 259
8.2 Déclaration d'instances particulières de classes patron ou de fonctions patron 260
8.3 Déclaration d'un autre patron de fonctions ou de classes 261
9. UN EXEMPLE DE CLASSE TABLEAU À DEUX INDICES 261

XIII. LA TECHNIQUE DE L'HÉRITAGE 267

1. MISE EN ŒUVRE DE L'HÉRITAGE EN C++ 268
2. UTILISATION, DANS UNE CLASSE DÉRIVÉE, DES MEMBRES DE LA CLASSE DE BASE 271
3. REDÉFINITION DES FONCTIONS MEMBRE 274
4. APPEL DES CONSTRUCTEURS ET DES DESTRUCTEURS 276
4.1 Rappels 276
4.2 La hiérarchisation des appels 276
4.3 Transmission d'informations entre constructeurs 277
4.4 Exemple 278
4.5 D'une manière générale 280

5. CONTRÔLE DES ACCÈS 281
5.1 Les membres protégés 281
5.2 Exemple 282
5.3 Intérêt du statut protégé 283
5.4 Action sur le statut des membres d'une classe dérivée : dérivation publique ou privée 284
a) Rappels concernant la dérivation publique 284
b) Dérivation privée 285
5.5 Les possibilités de dérivation protégée (version 3) 286
5.6 Récapitulation 286
6. COMPATIBILITÉ ENTRE OBJETS D'UNE CLASSE DE BASE ET OBJETS D'UNE CLASSE DÉRIVÉE 287
6.1 Conversions d'un objet dérivé dans un objet d'un type de base 288
6.2 Conversions d'un pointeur sur une classe dérivée en un pointeur sur une classe de base 289
6.3 Limitations liées au "typage statique" des objets 290
6.4 Les risques de violation des protections de la classe de base 293
7. CAS DU CONSTRUCTEUR PAR RECOPIE 294
7.1 La classe dérivée (B) n'a pas défini de constructeur par recopie 295
7.2 La classe dérivée (B) a défini un constructeur par recopie 295
8. OPÉRATEUR D'AFFECTATION ET HÉRITAGE 297
8.1 La classe dérivée (B) n'a pas surdéfini l'opérateur = ... 298
8.2 La classe dérivée (B) a surdéfini l'opérateur = ... 298
9. HÉRITAGE ET FORME CANONIQUE D'UNE CLASSE 302
10. CE QU'EST L'HÉRITAGE ET CE QU'IL N'EST PAS 303
10.1 La situation d'héritage 303
a) Le type du résultat de l'appel 304
b) Le type des arguments de f 304
10.2 Exemples 305
a) Héritage dans pointcol d'un opérateur + défini dans point 305
b) Héritage dans pointcol de la fonction coincide de point 306
11. EXEMPLE DE CLASSE DÉRIVÉE 307
12. PATRONS DE CLASSES ET HÉRITAGE 311
13. RETOUR SUR LES POINTEURS SUR DES FONCTIONS MEMBRE 315
14. L'HÉRITAGE EN GÉNÉRAL 316
15. EXPLOITATION D'UNE CLASSE DÉRIVÉE 318

XIV. L'HÉRITAGE MULTIPLE 321

1. MISE EN ŒUVRE DE L'HÉRITAGE MULTIPLE 322
2. POUR RÉGLER LES ÉVENTUELS CONFLITS : LES CLASSES VIRTUELLES 326

3. APPELS DES CONSTRUCTEURS ET DES DESTRUCTEURS - CAS DES CLASSES VIRTUELLES 328
4. EXEMPLE D'UTILISATION DE L'HÉRITAGE MULTIPLE : LISTE CHAÎNÉE DE POINTS 330
4.1 Une classe abstraite : liste chaînée 330
4.2 Création par héritage multiple d'une classe liste_points 335

XV. LES FONCTIONS VIRTUELLES ET LE TYPAGE DYNAMIQUE 339

1. RAPPEL D'UNE SITUATION OÙ LE TYPAGE DYNAMIQUE EST NÉCESSAIRE 340
2. LE MÉCANISME DES FONCTIONS VIRTUELLES 341
3. UNE AUTRE SITUATION OÙ LA LIGATURE DYNAMIQUE EST INDISPENSABLE 343
4. LES FONCTIONS VIRTUELLES EN GÉNÉRAL 346
4.1 Leurs limitations sont celles de l'héritage 346
4.2 La redéfinition d'une méthode virtuelle n'est pas obligatoire 348
4.3 Fonctions virtuelles et surdéfinition 348
4.4 On peut déclarer une fonction virtuelle dans n'importe quelle classe 349
4.5 Quelques restrictions 349
5. LES FONCTIONS VIRTUELLES PURES : UN OUTIL POUR LA CRÉATION DE CLASSES ABSTRAITES 350
6. EXEMPLE D'UTILISATION DE FONCTIONS VIRTUELLES : LISTE HÉTÉROGÈNE 351
7. LE MÉCANISME D'IDENTIFICATION DYNAMIQUE DES OBJETS 355

XVI. LES FLOTS 359

1. PRÉSENTATION GÉNÉRALE DE LA CLASSE OSTREAM 361
1.1 Surdéfinition de l'opérateur << 361
1.2 Les flots prédéfinis 362
1.3 La fonction put 362
1.4 La fonction write 363
1.5 Quelques possibilités de formatage 364
a) Action sur la base de numération 364
b) Action sur le gabarit de l'information écrite 365
2. PRÉSENTATION GÉNÉRALE DE LA CLASSE ISTREAM 366
2.1 Surdéfinition de l'opérateur >> 367
2.2 La fonction get 368
2.3 Les fonctions getline et gcount 369
2.4 La fonction read 370
2.5 Quelques autres fonctions 371
3. STATUT D'ERREUR D'UN FLOT 371
3.1 Les bits d'erreur 372

3.2 Action concernant les bits d'erreur 372
a) Accès aux bits d'erreur 373
b) Modification du statut d'erreur 373
3.3 Surdéfinition des opérateurs () et ! 374
4. SURDÉFINITION DES OPÉRATEURS << ET >> POUR LES TYPES DÉFINIS PAR L'UTILISATEUR 375
4.1 La démarche 375
4.2 Exemple 376
5. GESTION DU FORMATAGE 379
5.1 Le statut de formatage d'un flot 379
5.2 Description du mot d'état du statut de formatage 380
5.3 Action sur le statut de formatage 382
a) Les manipulateurs non paramétriques 382
b) Les manipulateurs paramétriques 383
c) Les fonctions membre 384
6. CONNEXION D'UN FLOT À UN FICHIER 387
6.1 Connexion d'un flot de sortie à un fichier 387
6.2 Connexion d'un flot d'entrée à un fichier 390
6.3 Les possibilités d'accès direct 391
6.4 Les différents modes d'ouverture d'un fichier 393
7. LES POSSIBILITÉS DE FORMATAGE EN MÉMOIRE 395
7.1 La classe ostrstream 395
7.2 La classe istrstream 397

XVII. LA GESTION DES EXCEPTIONS 399

1. UN PREMIER EXEMPLE D'EXCEPTION 399
1.1 Comment lancer une exception : l'instruction throw 400
1.2 Utilisation d'un gestionnaire d'exception 401
1.3 Récapitulatif 401
1.4 Commentaires 403
2. UN SECOND EXEMPLE 403
3. LES EXCEPTIONS D'UNE MANIÈRE GÉNÉRALE 406
3.1 Après l'exécution du gestionnaire d'exception 406
3.2 Algorithme de choix du gestionnaire d'interruption 407
3.3 Le cheminement des exceptions 408
3.4 Spécification d'interface 409
4. LES EXCEPTIONS STANDARD 410

XVIII. GÉNÉRALITÉS CONCERNANT LA BIBLIOTHÈQUE STANDARD 411

1. NOTIONS DE CONTENEUR, D'ITÉRATEUR ET D'ALGORITHME 412
1.1 Notion de conteneur 412
1.2 Notion d'itérateur 413

1.3 Parcours d'un conteneur avec un itérateur 413
1.4 Intervalle d'itérateur 415
1.5 Notion d'algorithme 415
1.6 Itérateurs et pointeurs 417
2. LES DIFFÉRENTES SORTES DE CONTENEURS 417
2.1 Conteneurs et structures de données classiques 417
2.2 Les différentes catégories de conteneurs 418
3. LES GÉNÉRATEURS D'OPÉRATEURS 419
4. LES CONTENEURS DONT LES ÉLÉMENTS SONT DES OBJETS 420
4.1 Construction et copie d'objets 421
4.2 Autres opérations 422
5. EFFICACITÉ DES OPÉRATIONS SUR DES CONTENEURS 422
6. FONCTIONS, PRÉDICATS ET CLASSES FONCTION 423
6.1 Fonction unaire 423
6.2 Prédicats 424
6.3 Classes et objets fonction 424
7. CONTENEURS, ALGORITHMES ET RELATION D'ORDRE 426
7.1 Introduction 426
7.2 Propriétés à respecter 427

XIX. LES CONTENEURS SÉQUENTIELS 429

1. FONCTIONNALITÉS COMMUNES AUX CONTENEURS VECTOR, LIST ET DEQUE 430
1.1 Construction 430
1.2 Modifications globales 432
1.3 Comparaison de conteneurs 434
1.4 Insertion ou suppression d'éléments 435
2. LE CONTENEUR VECTOR 438
2.1 Accès aux élément existants 438
2.2 Insertions et suppressions 440
2.3 Gestion de l'emplacement mémoire 440
2.4 Exemple 442
2.5 Cas particulier des vecteurs de booléns 444
3. LE CONTENEUR DEQUE 444
3.1 Présentation générale 444
3.2 Exemple 445
4. LE CONTENEUR LIST 446
4.1 Accès aux éléments existants 446
4.2 Insertions et suppressions 447
4.3 Opérations globales 448
4.4 Gestion de l'emplacement mémoire 452
4.5 Exemple 452

5. LES ADAPTATEURS DE CONTENEUR : QUEUE, STACK ET PRIORITY_QUEUE 453
5.1 L'adaptateur stack 454
5.2 L'adaptateur queue 455
5.3 L'adaptateur priority_queue 456

XX. LES CONTENEURS ASSOCIATIFS 459

1. LE CONTENEUR MAP 460
1.1 Exemple introductif 460
1.2 Le patron de classes pair 463
1.3 Construction d'un conteneur de type map 463
1.4 Accès aux éléments 467
1.5 Insertions et suppressions 468
1.6 Gestion mémoire 471
1.7 Autres possibilités 471
1.8 Exemple 471
2. LE CONTENEUR MULTIMAP 472
2.1 Présentation générale 472
2.2 Exemple 474
3. LE CONTENEUR SET 475
3.1 Présentation générale 475
3.2 Exemple 476
3.3 Le conteneur set et l'ensemble mathématique 477
4. LE CONTENEUR MULTISET 477
5. CONTENEURS ASSOCIATIFS ET ALGORITHMES 479

XXI. LES ALGORITHMES STANDARD 481

1. NOTIONS GÉNÉRALES 481
1.1 Algorithmes et itérateurs 481
1.2 Les catégories d'itérateurs 482
1.3 Algorithme et séquences 484
1.4 Itérateur d'insertion 484
1.5 Itérateur de flot 487
2. ALGORITHMES D'INITIALISATION DE SÉQUENCES EXISTANTES 489
2.1 Copie d'une séquence dans une autre 489
2.2 Génération de valeurs par une fonction 491
3. ALGORITHMES DE RECHERCHE 493
3.1 Algorithmes fondés sur une égalité ou un prédicat unaire 493
3.2 Algorithmes de recherche de maximum ou de minimum 495
4. ALGORITHMES DE TRANSFORMATION D'UNE SÉQUENCE 496
4.1 Remplacement de valeurs 496
4.2 Permutations de valeurs 497

4.3 Partitions 500
5. ALGORITHMES DITS "DE SUPPRESSION" 501
6. ALGORITHMES DE TRIS 503
7. ALGORITHMES DE RECHERCHE ET DE FUSION SUR DES SÉQUENCES ORDONNÉES 505
7.1 Algorithmes de recherche binaire 505
7.2 Algorithmes de fusion 505
8. ALGORITHMES À CARACTÈRE NUMÉRIQUE 507
9. ALGORITHMES À CARACTÈRE ENSEMBLISTE 508

XXII. LA CLASSE STRING 511

1. GÉNÉRALITÉS 512
2. CONSTRUCTION 512
3. OPÉRATIONS GLOBALES 513
4. CONCATÉNATION 513
5. RECHERCHE DANS UNE CHAÎNE 514
5.1 Recherche d'une chaîne ou d'un caractère 514
5.2 Recherche d'un caractère présent ou absent d'une suite 515
6. INSERTIONS, SUPPRESSIONS ET REMPLACEMENTS 516
6.1 Insertions 516
6.2 Suppressions 517
6.3 Remplacements 518

XXIII. LES OUTILS NUMÉRIQUES 521

1. LA CLASSE COMPLEX 521
2. LA CLASSE VALARRAY 523

ANNEXE A : RÈGLES DE MISE EN CORRESPONDANCE D'ARGUMENTS 525

1. CAS DES FONCTIONS À UN ARGUMENT 525
1.1 Recherche d'une correspondance exacte 525
1.2 Promotions numériques 526
1.3 Conversions standard 527
1.4 Conversions définies par l'utilisateur 527
1.5 Fonctions à arguments variables 528
1.6 Exception 528
2. CAS DES FONCTIONS À PLUSIEURS ARGUMENTS 528
3. CAS DES FONCTIONS MEMBRE 529

ANNEXE B : LES INCOMPATIBILITÉS ENTRE C ET C++ 531

ANNEXE C : OPÉRATEURS DE CAST ET IDENTIFICATION DE TYPE À L'ÉXÉCUTION 537

1. IDENTIFICATION DE TYPE À L'ÉXÉCUTION 537
   1.1 Utilisation du champ name de Type_info 538
   1.2 Utilisation des opérateurs de comparaison de Type_info 540
   1.3 Exemple avec des références 541
2. LES NOUVEAUX OPÉRATEURS DE "CAST" 542
3. LES "CAST" DYNAMIQUES 544
   3.1 Introduction 544
   3.2 D'une manière générale 544
   3.3 Exemple 544

ANNEXE D : LES DIFFÉRENTES SORTES DE FONCTIONS EN C++ 547

ANNEXE E : COMPTAGE DE RÉFÉRENCES 549

ANNEXE F : LES ALGORITHMES STANDARD 553

1. ALGORITHMES D'INITIALISATION DE SÉQUENCES EXISTANTES 554
2. ALGORITHMES DE RECHERCHE 555
3. ALGORITHMES DE TRANSFORMATION D'UNE SÉQUENCE 558
4. ALGORITHMES DE SUPPRESSION 561
5. ALGORITHMES DE TRI 562
6. ALGORITHMES DE RECHERCHE ET DE FUSIONS SUR DES SÉQUENCES ORDONNÉES 564
7. ALGORITHMES À CARACTÈRE NUMÉRIQUE 567
8. ALGORITHMES À CARACTÈRE ENSEMBLISTE 568
9. ALGORITHMES DIVERS 571

CORRECTION DES EXERCICES 573

INDEX 589

TABLE DES MATIÈRES

AVANT-PROPOS V

I. PRÉSENTATION GÉNÉRALE DE C++ 1
1. LA PROGRAMMATION ORIENTÉE OBJET 1
1.1 La problématique de la programmation 1
1.2 La programmation structurée 2
1.3 La Programmation Orientée Objet 3
1.4 P.O.O. et langages 4
2. C++, C ANSI ET P.O.O. 5
3. LES INCOMPATIBILITÉS DE C++ AVEC LE C ANSI 6
4. LES SPÉCIFICITÉS DE C++ 6
5. C++ ET LA PROGRAMMATION ORIENTÉE OBJET 7

II. LES INCOMPATIBILITÉS DE C++ AVEC LE C ANSI 11
1. LES DÉFINITIONS DE FONCTIONS EN C++ 12
2. LES PROTOTYPES EN C++ 12
3. ARGUMENTS ET VALEUR DE RETOUR D'UNE FONCTION 16
3.1 Points communs à C et C++ 16
3.2 Différences entre C et C++ 17
4. LE QUALIFICATIF CONST 17
4.1 Portée 18
4.2 Utilisation dans une expression 19
5. COMPATIBILITÉ ENTRE LE TYPE VOID * ET LES AUTRES POINTEURS 20