

Programmer en Langage C++

Claude Delannoy

CHIHAB-EYROLLES



TABLE DES MATIERES

AVANT-PROPOS	V
I. PRESENTATION GENERALE DE C + +	1
1. LA PROGRAMMATION ORIENTEE 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 INCOMPATIBILITES DE C + + AVEC LE C ANSI.....	6
4. LES SPECIFICITES DE C + +	6
5. C + + ET LA PROGRAMMATION ORIENTEE OBJET.....	7
II. LES INCOMPATIBILITES DE C + + AVEC LE C ANSI	11
1. LES DEFINITIONS DE FONCTIONS EN C + +	12
2. LES PROTOTYPES EN C + +	13
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	18
4.1 Portée	18
4.2 Utilisation dans une expression	19
5. COMPATIBILITE ENTRE LE TYPE VOID * ET LES AUTRES POINTEURS	20

III. LES NOUVELLES POSSIBILITES D'ENTREES-SORTIES CONVERSATIONNELLES DE C + +	21
1. LES NOUVELLES ENTREES-SORTIES EN C + +	21
2. ECRITURE SUR LA SORTIE STANDARD.....	22
2.1 Quelques exemples	22
2.2 Les possibilités d'écriture sur cout	25
3. LECTURE SUR L'ENTREE STANDARD	26
3.1 Introduction	26
3.2 Les possibilités de lecture sur cin	27
3.3 Exemples	28
IV. LES SPECIFICITES DU C + +	31
1. LE COMMENTAIRE DE FIN DE LIGNE	32
2. DECLARATIONS ET INITIALISATIONS	32
3. LA NOTION DE REFERENCE	34
3.1 Transmission des arguments en C	34
3.2 Transmission par référence en C + +	36
3.3 La référence d'une manière générale.....	38
a) Transmission par référence de la valeur de retour d'une fonction.....	38
b) La notion de référence est plus générale que celle d'argument	38
c) Initialisation de référence.....	39
d) La transmission d'un argument ou d'une valeur de retour est une initialisation	40
4. LES ARGUMENTS PAR DEFAULT	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. SURDEFINITION DE FONCTIONS.....	46
5.1 Mise en oeuvre de la surdéfinition de fonctions.....	47
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.....	51
b) Cas des fonctions à plusieurs arguments.....	51
5.4 Le mécanisme de la surdéfinition de fonctions	52
6. LES OPERATEURS NEW ET DELETE	53
6.1 Exemples d'utilisation de new	54
6.2 Syntaxe et rôle de new	55
6.3 Exemples d'utilisation de l'opérateur delete	56
6.4 Syntaxe et rôle de l'opérateur delete.....	57
6.5 Pour gérer les débordements de mémoire : set new handler.....	57
7. LA SPECIFICATION INLINE	59
7.1 Rappels concernant les macros et les fonctions	59

7.2 Utilisation de fonctions "en ligne"	60
---	----

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 DONNEE STATIQUES	84
6. EXPLOITATION D'UNE CLASSE	87
6.1 La classe, comme "composant logiciel"	87
6.2 En cas de modification d'une classe.....	89
a) La déclaration des membres publics n'a pas changé.....	89
b) La déclaration des membres publics a changé.....	90
7. LES CLASSES EN GENERAL	90
7.1 Les autres sortes de classes en C + +	90
7.2 Ce qu'on peut trouver dans la déclaration d'une classe	90
7.3 Déclaration d'une classe	91
EXERCICES.....	91

VI. LES PROPRIETES DES FONCTIONS MEMBRE.....	93
---	-----------

1. SURDEFINITION DES FONCTIONS MEMBRE	94
2. ARGUMENTS PAR DEFAULT	95
3. LES FONCTIONS MEMBRE "EN LIGNE"	97
4. CAS DES OBJETS TRANSMIS EN ARGUMENT D'UNE FONCTION MEMBRE.....	99
5. MODE DE TRANSMISSION DES OBJETS EN ARGUMENT	102
5.1 Transmission de l'adresse d'un objet	102
5.2 Transmission par référence	104
5.3 Les problèmes posés par la transmission par valeur.....	105
6. LORSQUE LA VALEUR DE RETOUR D'UNE FONCTION EST ELLE-MEME UN OBJET.....	105
7. AUTO-REFERENCE : LE MOT CLE THIS	106
8. LES FONCTIONS MEMBRE STATIQUES.....	108

9. LES FONCTIONS MEMBRE CONSTANTES	110
10 LES POINTEURS SUR DES FONCTIONS MEMBRE.....	112
EXERCICES.....	114
VII. CONSTRUCTION, DESTRUCTION ET INITIALISATION DES OBJETS	117
1. LES OBJETS AUTOMATIQUES ET STATIQUES.....	118
1.1 Leur durée de vie	119
1.2 Appel des constructeurs et des destructeurs	119
1.3 Exemple.....	120
2. LES OBJETS TEMPORAIRES	122
3. LES OBJETS DYNAMIQUES	124
3.1 Les structures dynamiques	125
3.2 Les objets dynamiques	125
a) Points communs avec les structures dynamiques	125
b) Les nouvelles possibilités des opérateurs new et delete	126
c) Exemple	127
4. INITIALISATION D'UN OBJET LORS DE SA DECLARATION.....	128
4.1 Un premier exemple.....	129
4.2 D'une manière générale.....	130
4.3 Présentation du "constructeur par recopie".....	131
a) Il n'existe pas de constructeur approprié.....	131
b) Il existe un constructeur approprié.....	132
4.4 Exemples d'utilisation de "constructeurs par recopie"	133
a) Emploi du constructeur par recopie par défaut.....	133
b) Définition d'un constructeur par recopie	135
4.5 Une exception.....	137
5. CAS DES OBJETS TRANSMIS EN ARGUMENT OU EN VALEUR DE RETOUR D'UNE FONCTION.....	138
6. LES TABLEAUX D'OBJETS.....	140
6.1 Notations	140
6.2 Constructeurs et initialiseurs.....	141
6.3 Cas des tableaux dynamiques d'objets	142
7. OBJETS MEMBRE.....	143
7.1 Introduction	143
7.2 Mise en oeuvre des constructeurs et des destructeurs	144
7.3 Cas du constructeur par recopie.....	147
EXERCICES.....	148
VIII. LES FONCTIONS AMIES	151
1. EXEMPLE DE FONCTION INDEPENDANTE AMIE D'UNE CLASSE.....	152
2. LES DIFFERENTES SITUATIONS D'AMITIE	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 SURDEFINITION D'OPERATEURS	163
1. LE MECANISME DE LA SURDEFINITION D'OPERATEUR.....	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 POSSIBILITES ET LES LIMITES DE LA SURDEFINITION D'OPERATEUR.....	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 SURDEFINITION DE L'OPERATEUR =	178
4. NOTION DE FORME CANONIQUE D'UNE CLASSE	184
5. EXEMPLE DE SURDEFINITION DE L'OPERATEUR []	185
6. SURDEFINITION DES OPERATEURS NEW ET DELETE.....	188
6.1 Surdéfinition de new et delete.....	188
6.2 Exemple.....	189
EXERCICES.....	192
X. LES CONVERSIONS DE TYPE DEFINIES PAR L'UTILISATEUR	195
1. LES DIFFERENTES SORTES DE CONVERSIONS DEFINIES PAR L'UTILISATEUR	196
2. L'OPERATEUR 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
4. LES CONVERSIONS D'UN TYPE CLASSE EN UN AUTRE TYPE CLASSE.....	215
4.1 Exemple simple d'opérateur de "cast".....	215
4.2 Exemple simple de conversion par un constructeur.....	216
4.3 Pour donner, dans une classe, une signification à un opérateur défini dans une autre classe.....	218
XI. LES PATRONS DE FONCTIONS.....	223
1. EXEMPLE DE CREATION 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 PARAMETRES DE TYPE D'UN PATRON DE FONCTION.....	229
2.1 Utilisation des paramètres de type dans la définition d'un patron.....	230
2.2 Algorithme d'instanciation d'une fonction patron.....	231
2.3 Pour pallier l'absence de conversions dans l'instanciation de fonctions patron.....	232
2.4 Nouvelle syntaxe d'initialisation des variables des types standards.....	233
2.5 Limitations des patrons de fonctions.....	234
3. LES PARAMETRES EXPRESSION D'UN PATRON DE FONCTIONS.....	235
4. SURDEFINITION DE PATRONS.....	236
4.1 Exemples de surdéfinition de patron de fonctions ne comportant que des paramètres de type.....	237
4.2 Exemples de surdéfinition de patron de fonctions comportant des paramètres expression.....	239
5. SPECIALISATION DE FONCTIONS DE PATRON.....	240
6. LES PATRONS DE FONCTIONS D'UNE MANIERE GENERALE.....	241
XII. LES PATRONS DE CLASSES.....	243
1. EXEMPLE DE CREATION ET D'UTILISATION 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 PARAMETRES DE TYPE D'UN PATRON DE CLASSES	249
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 PARAMETRES EXPRESSION D'UN PATRON DE CLASSES.....	251
3.1 Exemple.....	251
3.2 D'une manière générale	253
4. SPECIALISATION D'UN PATRON DE CLASSES	254
4.1 Exemple de spécialisation d'une fonction membre	254
4.2 D'une manière générale	256
a) On peut spécialiser pour les valeurs de tous les paramètres.....	256
b) On peut spécialiser une fonction membre ou une classe.....	256
5. IDENTITE DE CLASSES PATRON.....	257
6. CLASSES PATRON ET DECLARATIONS D'AMITIES	258
a) Déclaration de classes ou fonctions "ordinaires" amies	258
b) Déclaration d'instances particulières de classes patron ou de fonctions patron.....	258
c) Déclaration d'un autre patron de fonctions ou d'un autre patron de classes	259
7. UN EXEMPLE DE CLASSE TABLEAU A DEUX INDICES	260
XIII. LA TECHNIQUE DE L'HERITAGE	265
1. MISE EN OEUVRE DE L'HERITAGE EN C++	266
2. UTILISATION, DANS UNE CLASSE DERIVEE, DES MEMBRES DE LA CLASSE DE BASE	269
3. REDEFINITION DES FONCTIONS MEMBRE.....	272
4. APPEL DES CONSTRUCTEURS ET DES DESTRUCTEURS.....	274
4.1 Rappels.....	274
4.2 La hiérarchisation des appels.....	274
4.3 Transmission d'informations entre constructeurs	275
4.4 Exemple.....	276
4.5 D'une manière générale	278
4.6 Cas du constructeur par recopie.....	279
a) La classe dérivée (B) n'a pas défini de constructeur par recopie	279
b) La classe dérivée (B) a défini un constructeur par recopie	280
5. CONTROLE DES ACCES	283
5.1 Les membres protégés.....	284
5.2 Exemple.....	285
5.3 Intérêt du statut protégé	285
5.4 Action sur le	286
a) Rappels concernant la dérivation publique	286
b) Dérivation privée.....	287
5.5 Les possibilités de dérivation protégée (version 3)	289
5.6 Récapitulation	289

6. L'HERITAGE EN GENERAL.....	290
7. EXPLOITATION D'UNE CLASSE DERIVEE.....	291
8. COMPATIBILITE ENTRE OBJETS D'UNE CLASSE DE BASE ET OBJETS D'UNE CLASSE DERIVEE.....	293
8.1 Conversions d'un objet dérivé dans un objet d'un type de base.....	294
8.2 Conversions d'un pointeur sur une classe dérivée en un pointeur sur une classe de base	295
8.3 Limitations liées au "typage statique" des objets	296
8.4 Les risques de violation des protections de la classe de base.....	299
9. OPERATEUR D'AFECTATION ET HERITAGE	300
9.1 La classe dérivée (B) n'a pas surdéfini l'opérateur =	301
9.2 La classe dérivée (B) a surdéfini l'opérateur =	301
10. RETOUR SUR LES POINTEURS SUR DES FONCTIONS MEMBRE	305
11. POUR ELARGIR LA NOTION D'HERITAGE	306
11.1 La situation d'héritage.....	307
a) Le type du résultat de l'appel	308
b) Le type des arguments de f.....	308
11.2 Exemples	309
a) Héritage dans pointcol d'un opérateur + défini dans point.....	309
b) Héritage dans pointcol de la fonction coincide de point... ..	310
12. EXEMPLE DE CLASSE DERIVEE.....	311
13. PATRONS DE CLASSES ET HERITAGE	315
XIV. L'HERITAGE MULTIPLE	319
1. MISE EN OEUVRE DE L'HERITAGE MULTIPLE	320
2. POUR REGLER LES EVENTUELS CONFLITS : LES CLASSES VIRTUELLES.....	324
3. APPELS DES CONSTRUCTEURS ET DES DESTRUCTEURS - CAS DES CLASSES VIRTUELLES	326
4. EXEMPLE D'UTILISATION DE L'HERITAGE MULTIPLE : LISTE CHAINEE DE POINTS	328
4.1 Une classe abstraite : liste chaînée	329
4.2 Création par héritage multiple, d'une classe liste_points.....	333
XV. LES FONCTIONS VIRTUELLES ET LE TYPAGE DYNAMIQUE	337
1. RAPPEL D'UNE SITUATION OU LE TYPAGE DYNAMIQUE EST NECESSAIRE	338
2. LE MECANISME DES FONCTIONS VIRTUELLES	339
3. UNE AUTRE SITUATION OU LA LIGATURE DYNAMIQUE EST INDISPENSABLE	341
4. LE MECANISME D'IDENTIFICATION DYNAMIQUE DES OBJETS.....	344
5. LES FONCTIONS VIRTUELLES EN GENERAL.....	347
5.1 Leurs limitations sont celles de l'héritage.....	347

5.2 La redéfinition d'une méthode virtuelle n'est pas obligatoire	348
5.3 Fonctions virtuelles et surdéfinition	349
5.4 On peut déclarer une fonction virtuelle dans n'importe quelle classe	349
5.5 Quelques restrictions.....	350
6. LES FONCTIONS VIRTUELLES PURES : UN OUTIL POUR LA CREATION DE CLASSES ABSTRAITES	350
7. EXEMPLE D'UTILISATION DE FONCTIONS VIRTUELLES : LISTE HETEROGENE.....	352
XVI. LES FLOTS.....	357
1. PRESENTATION GENERALE DE LA CLASSE OSTREAM.....	359
1.1 Surdéfinition de l'opérateur < <	359
1.2 Les flots prédéfinis.....	360
1.3 La fonction put	361
1.4 La fonction write	361
1.5 Quelques possibilités de formatage.....	362
a) Action sur la base de numération	362
b) Action sur le gabarit de l'information écrite	363
2. PRESENTATION GENERALE DE LA CLASSE ISTREAM	364
2.1 Surdéfinition de l'opérateur > >	365
2.2 La fonction get	366
2.3 Les fonctions getline et gcount.....	367
2.4 La fonction read.....	368
2.5 Quelques autres fonctions	369
3. STATUT D'ERREUR D'UN FLOT	369
3.1 Les bits d'erreur	370
3.2 Action concernant les bits d'erreur	370
a) Accès aux bits d'erreur	371
b) Modification du statut d'erreur	371
3.3 Surdéfinition des opérateurs () et !	372
4. SURDEFINITION DES OPERATEURS < < ET > > POUR LES TYPES DEFINIS PAR L'UTILISATEUR.....	373
4.1 La démarche.....	373
4.2 Exemple.....	374
5. GESTION DU FORMATAGE	377
5.1 Le statut de formatage d'un flot.....	378
5.2 Description du mot d'état du statut de formatage	378
5.3 Action sur le statut de formatage	380
a) Les manipulateurs non paramétriques	380
b) Les manipulateurs paramétriques.....	381
c) Les fonctions membre.....	382
6. CONNEXION D'UN FLOT A UN FICHIER.....	385
6.1 Connexion d'un flot de sortie à un fichier	385

6.2 Connexion d'un flot d'entrée à un fichier	387
6.3 Les possibilités d'accès direct	389
6.4 Les différents modes d'ouverture d'un fichier	391
7. LES POSSIBILITES DE FORMATAGE EN MEMOIRE	392
7.1 La classe ostrstream	392
7.2 La classe istrstream	393
ANNEXE A : REGLES DE "MISE EN CORRESPONDANCE D'ARGUMENTS"	397
1. CAS DES FONCTIONS A UN ARGUMENT	397
1.1 Recherche d'une correspondance exacte	397
1.2 Promotions numériques	398
1.3 Conversions standard	399
1.4 Conversions définies par l'utilisateur	399
1.5 Fonctions à arguments variables	400
1.6 Exception	400
2. CAS DES FONCTIONS A PLUSIEURS ARGUMENTS	400
3 CAS DES FONCTIONS MEMBRE	401
ANNEXE B : LES INCOMPATIBILITES ENTRE C ET C + +	403
ANNEXE C : DIFFERENCES ENTRE LES VERSIONS 2.0 ET 3.0	409
ANNEXE D : LES DIFFERENTES SORTES DE FONCTIONS EN C + +	411
ANNEXE E : COMPTAGE DE REFERENCES	413
CORRECTION DES EXERCICES	417
INDEX	433