
2e année – Spécialité Informatique

Année 2024 – 2025

# Programmation orientée objet en C++

Notes de cours

Sébastien Fourey

**Chapitre 8.** Épilogue, Bibliographie, Index

Version du 3 septembre 2024

# Chapitre 8

# Épilogue

Fig. 5. — Haricot qui a germé *en pleine lumière*. Il est vert, son poids a augmenté : il a vécu sur le carbone qu'il a pris à l'acide carbonique de l'air.

Nous terminons ce document par la présentation du moyen d'obtenir, en C++ , des informations propres à une implémentation particulière. Une deuxième courte section est consacrée à des mots-clés qui existent pour des raisons essentiellement historiques et que nous qualifions ici d'exotiques.

## 8.1 Données propres à l'implémentation

En C++ des données concernant les types de base sont fournies par l'en-tête `<limits>`. Pour ce faire, chaque implémentation définit ses propres *spécialisations* d'un modèle de structure, le modèle `numeric_limits<>`. Celui-ci définit finalement les *traits*[1] des types de base du langage (voir listing 8.1).  donne un aperçu du modèle général.

### Exemples

Avec le compilateur GNU g++ (GCC) 3.2.1 sur une architecture SPARC™ :

L'instruction

```
cout << numeric_limits<unsigned long long>::max();
```

affiche :

```
18446744073709551615
```

D'autre part, l'instruction

```
cout << numeric_limits<unsigned long long>::digits;
```

affiche

```
64
```

**Exercice 8.1** *Quelle est la relation entre les deux entiers qui viennent d'être « affichés » ?*

1. La notion de *trait* (c.-à-d. caractéristique), basée sur la spécialisation de modèle de classe ou de structure, est normalement abordée par l'exemple en cours, dans le cadre de la programmation générique (§ 3.3).

```
1  template <typename T>
2  struct numeric_limits {
3     static const bool is_specialized;
4     static const int digits = ? ? ? ;
5     static const bool is_signed = ? ? ? ;
6     static const bool is_integer = ? ? ? ;
7     static const bool is_exact = ? ? ? ;
8     static const int radix = ? ? ? ;
9     static const int digits = ? ? ? ;
10    static const int digits10 = ? ? ? ;
11    static const bool is_modulo = ? ? ? ;
12    static const bool has_infinity;
13    inline static T min_exponant = ? ? ? ;
14    inline static T max_exponant = ? ? ? ;
15    inline static T min() throw();
16    inline static T max() throw();
17    inline static T epsilon() throw();
18    inline static T infinity() throw();
19    // ...
20 };
```

Listing 8.1 – Extrait de la définition du modèle `numeric_limits<>`.

Notez que le type « long long » est apparu avec la norme de 2011 du langage, même s'il était disponible auparavant avec certains compilateurs comme g++.

## 8.2 Exotisme, les synonymes d'opérateurs

Plusieurs opérateurs du langage ont des synonymes sous la forme de mots-clés (qui sont donc des noms réservés).

| Opér. | Mot-clé | Opér. | Mot-clé | Opér. | Mot-clé |
|---|---|---|---|---|---|
| && | and | & | bitand | &= | and_eq |
| \|\| | or | \| | bitor | \|= | or_eq |
| ! | not | ~ | compl | ~= | not_eq |
| | | ^ | xor | ^= | xor_eq |

Table 8.1 – Synonymes d'opérateurs.

Le listing 8.2 montre que ces mots-clés, qui restent d'un usage marginal, mériteraient peut être d'être utilisés plus souvent puisqu'ils apportent, pour un lecteur peu familier du langage, un certain gain d'« intelligibilité ». Il existe aussi des équivalents pour certaines unités lexicales (cf. tableau 8.2). Il ne s'agit pas d'opérateurs, et à la différence des synonymes précédents leur utilisation a tendance à obscurcir le code pour le programmeur non averti.

| Symbole | Équiv. | Symbole | Équiv. |
|---|---|---|---|
| { | <% | } | %> |
| [ | <: | ] | :> |
| # | %: | ## | %:%: |

TABLE 8.2 – Synonymes d'unités lexicales.

```
1  struct Knowledge {};
2  struct Teacher {
3    void increases(Knowledge & k);
4  };
5  struct Language {
6    bool isGoodToKnow() {
7      return true;
8    }
9    Language & operator++();
10   Language operator++(int);
11 };
12 struct Reading {
13   bool isGood() {
14     return true;
15   }
16 };
17 Reading theLecture() {
18   return Reading();
19 }
20 bool youKnow(Language language);
21 namespace Your
22 {
23 int Help;
24 }
25
26 int main() {
27   Knowledge yourKnowledge;
28   Teacher sebastienFourey;
29   Language c;
30
31   if ((youKnow(c++) or theLecture().isGood())
32       and (c++).isGoodToKnow()) {
33     using Your::Help;
34     sebastienFourey.increases(yourKnowledge);
35   }
36 }
```

Listing 8.2 – Un code correct et très lisible, mais plutôt vers la fin.

# Remerciements

Les personnes citées ci-après ont contribué à l'amélioration de la qualité de ce support par leurs suggestions de corrections ou simples modifications.

— Nicolas Signolle, attaché temporaire à l'enseignement et à la recherche à l'Ensicaen en 2007-2008 ;
— Christine Porquet, maître de conférences à l'Ensicaen ;
— Loïc Simon, maître de conférences à l'Ensicaen ;
— Yoran Le Bagousse, étudiant en spécialité informatique à l'Ensicaen (promotion 2014) ;
— Charles Chaudet, étudiant en spécialité informatique à l'Ensicaen (promotion 2019).
— Bastien Hubert, étudiant en spécialité informatique à l'Ensicaen (promotion 2021).
— Julien Zaïdi, étudiant en spécialité informatique à l'Ensicaen (promotion 2022).

Tout élève est invité à mettre en défaut le contenu de ce document pour voir apparaître son nom dans les versions futures. En effet, l'auteur ne peut prétendre qu'appocher (modestement) de très loin le niveau d'exactitude du créateur du langage. D'ailleurs, Bjarne Stroustrup lui-même commet des erreurs comme le confirme la page d'errata de la 3e édition du livre de référence [8] dont l'URL est donnée ci-dessous.

cf. `http://www.informit.com/content/images/0201889544/errata%5C833.pdf`

# Conclusion

Beaucoup de sujets n'ont pas été abordés dans ce document. Certains sont traités lors des cours magistraux, d'autres sont rencontrés à l'occasion des travaux pratiques. Les exercices proposés dans les différents chapitres devraient aussi avoir initié quelques recherches personnelles. À titre de document de référence complet sur le langage, et en dehors de la norme elle-même [4], le livre de Bjarne Stroustrup [11] est sans doute le plus complet.

Le support s'achèvera par une citation de M. Clouard, enseignant-chercheur en informatique à l'ENSICAEN depuis 1999.

*« Si tu veux savoir programmer en C++, il faut que tu pratiques en permanence ; y compris les week-ends car sinon le lundi, tu ne sais plus rien faire avec. »*

Régis Clouard [2]

2. Entretien avec l'auteur, Été 2006.

# Bibliographie


[1] Paul Bert. *La première année d'enseignement scientifique (Sciences naturelles et physiques)*. Librairie classique Armand Colin et C^ie, Paris, 1882. 550 gravures.

[2] Michel Desvignes. *Programmation objet en C++, notes de cours*. ENSICAEN, M^me Mullois Édition, 2002.

[3] International Organization for Standardization. *Programming languages – C++*. International standard ISO/IEC 14882:2003, Octobre 2003. Remplace 14882:1998.

[4] International Organization for Standardization. *Information technology – Programming languages – C++*. International standard ISO/IEC 14882:2011, Mars 2011. Remplace 14882:2003.

[5] Ray Lischner. *STL – Précis & concis*. O'Reilly, 2004.

[6] Scott Meyers. *Programmer efficacement en C++*. Dunod, 2016.

[7] Standard C++ Foundation. C++ FAQ, Exceptions and Error Handling [Page Web]. `https://isocpp.org/wiki/faq/exceptions#ctors-can-throw`, 2016. [visité en octobre 2022].

[8] Bjarne Stroustrup. *The C++ programming language, third edition*. Addison Wesley Publishing Company, 1997.

[9] Bjarne Stroustrup. *C++*. CampusPress France, 1999. Troisième édition du livre de Bjarne Stroustrup, traduit de l'américain par Christine Eberhardt.

[10] Bjarne Stroustrup. Sibling Rivalry: C and C++. AT&T Labs – Research Technical Report TD-54MQZY, `http://www.stroustrup.com/sibling_rivalry.pdf`, Janvier 2002. [visité en octobre 2022].

[11] Bjarne Stroustrup. *The C++ Programming Language, fourth edition*. Pearson Education, 2013.

[12] Bjarne Stroustrup. Bjarne Stroustrup's C++11 FAQ [Page Web]. `http://www.stroustrup.com/C++11FAQ.html`, 2014. [visité en octobre 2022].

[13] Bjarne Stroustrup. Bjarne Stroustrup's FAQ [Page Web]. `http://www.stroustrup.com/bs_faq.html#really-say-that`, 2014. [visité en octobre 2022].

# Index