M1103 Amphi 1

Thomas Nowak

Université Paris-Sud

M1103 - Structures de données et algorithmes fondamentaux

Organisation du cours

Organisation du cours

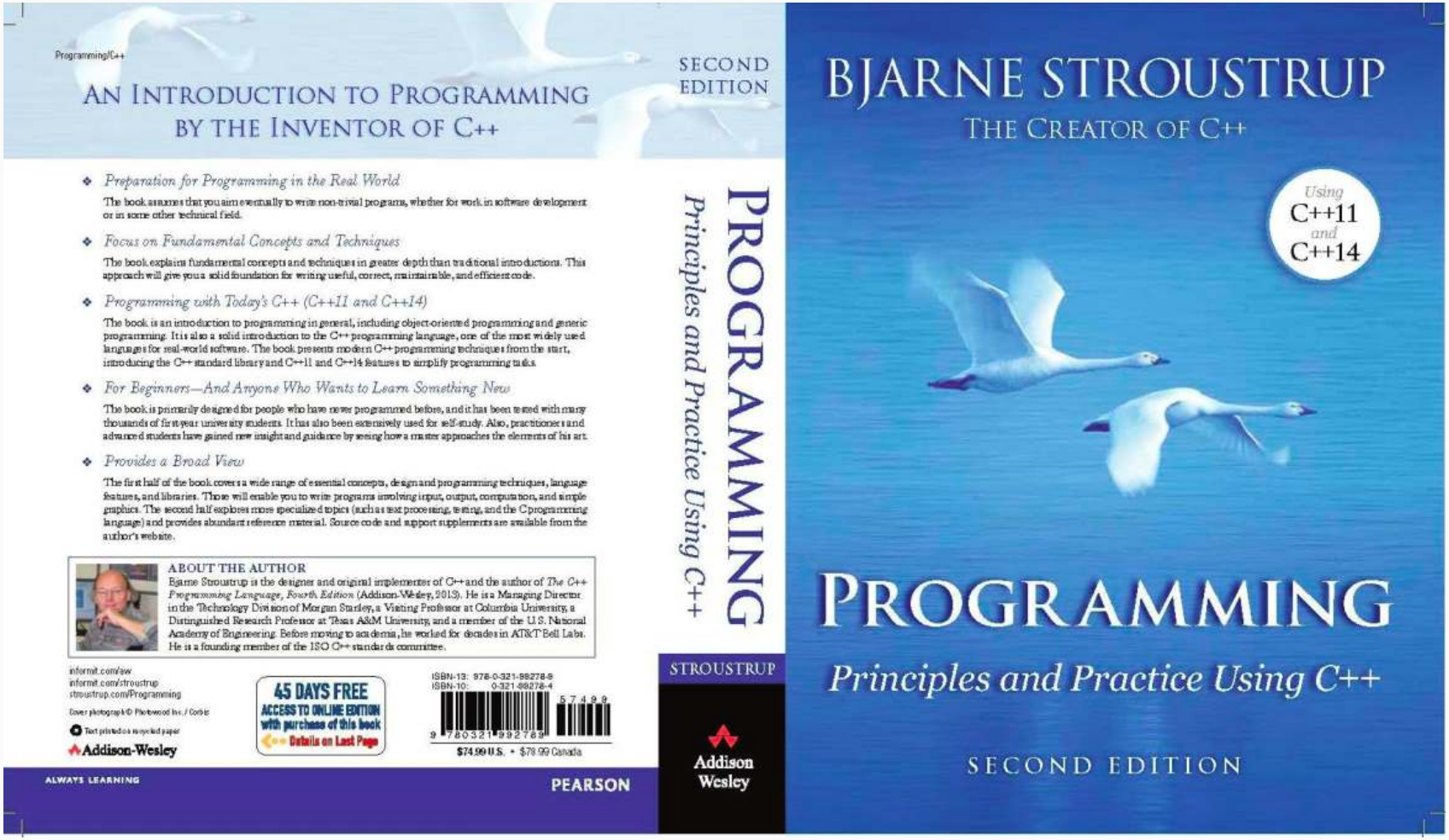
Site web du cours

- https://www.lri.fr/~nowak/teaching/algoprog

Cours sous Moodle

- https://cours.iut-orsay.fr/course/view.php?id=301
- Clé d'inscription : cléGroupe-1A, cléGroupe-1B, ...
- Rendus d'exercices, forum de questions, feedback

Livre du cours



Organisation du cours

Séances présentielles

- amphis : 1h/semaine
- TP : $2 \times$ 2h/semaine
- récitations : 1h30/semaine

Exercices

- 1 mini-feuille d'exercices chaque semaine ($\leq$ 20 minutes)
- 4 feuilles d'exercices
- tests automatiques fournis
- rendus sous Moodle

Organisation du cours

Notation

- 10% mini-feuilles
- 30% feuilles
- 60% exam final

Organisation du cours

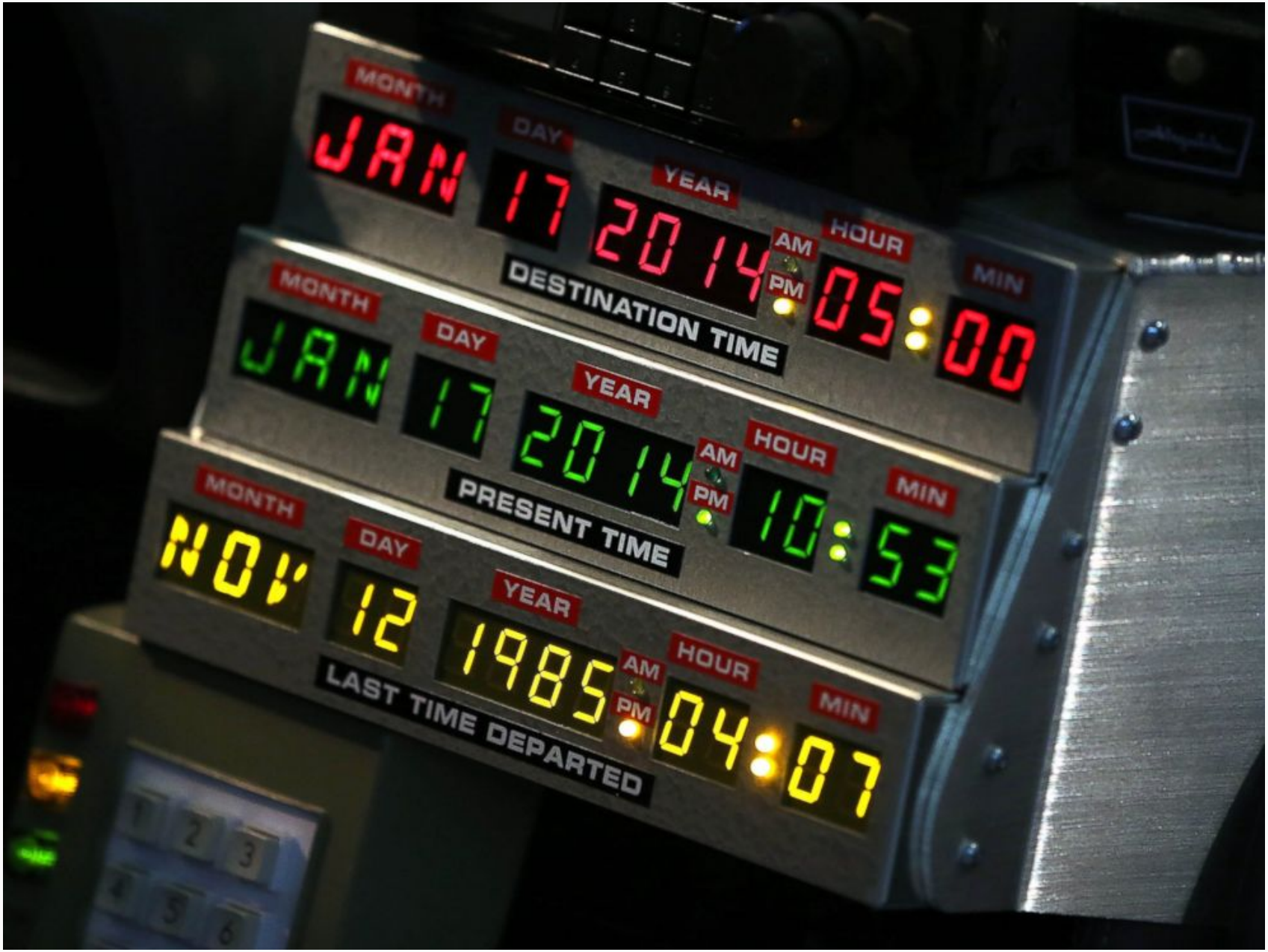
Collaboration

- collaboration en groupes de 2 autorisée
  - écrire le nom du binôme dans *tout* fichier rendu
- discussion de votre code autorisée
  - ne jamais copier le code d'un.e autre étudiant.e
  - ne jamais écrire du code pour un.e autre étudiant.e

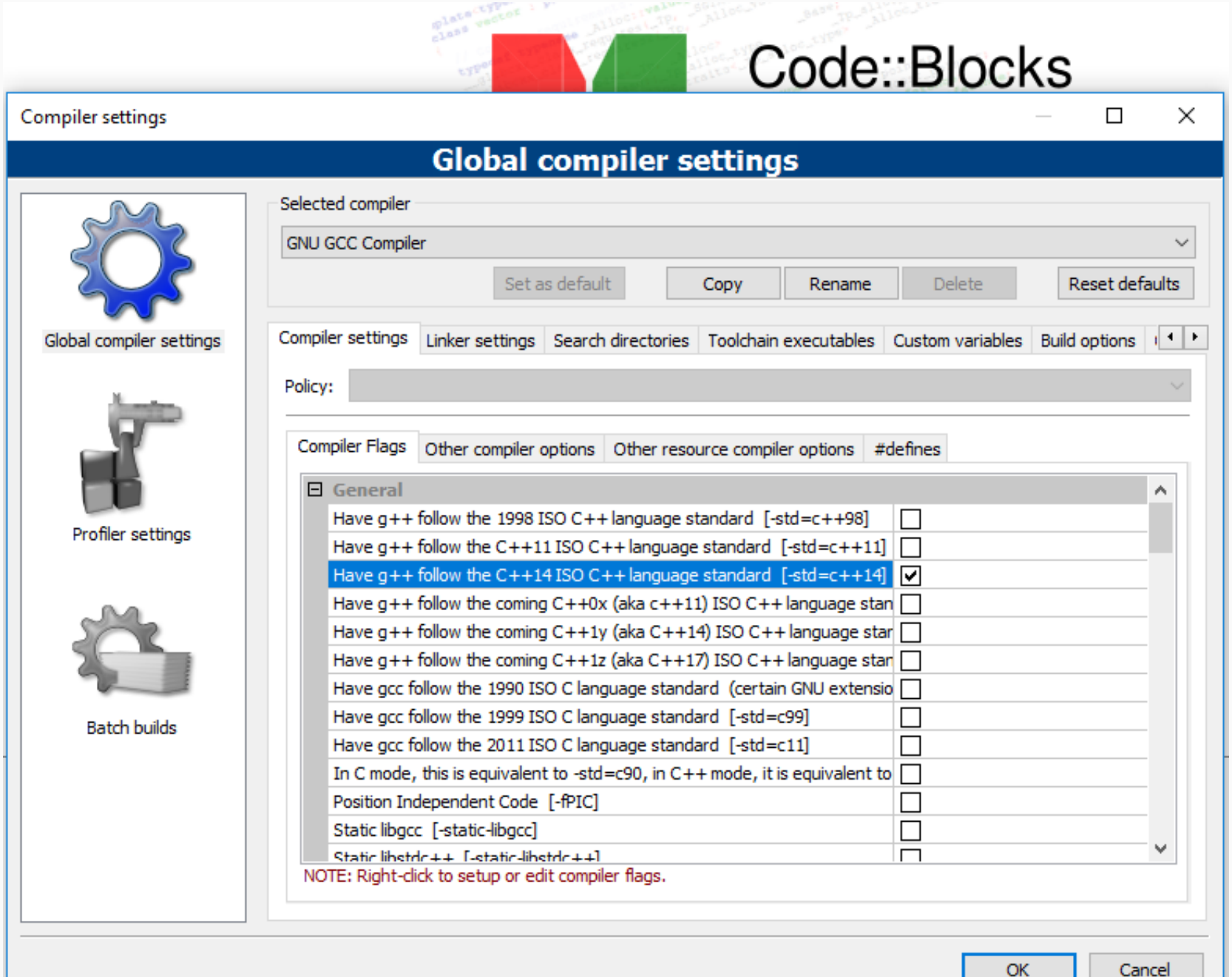
Au programme

1. vecteurs
2. objets, classes
3. temps d'exécution
4. recherche, tri

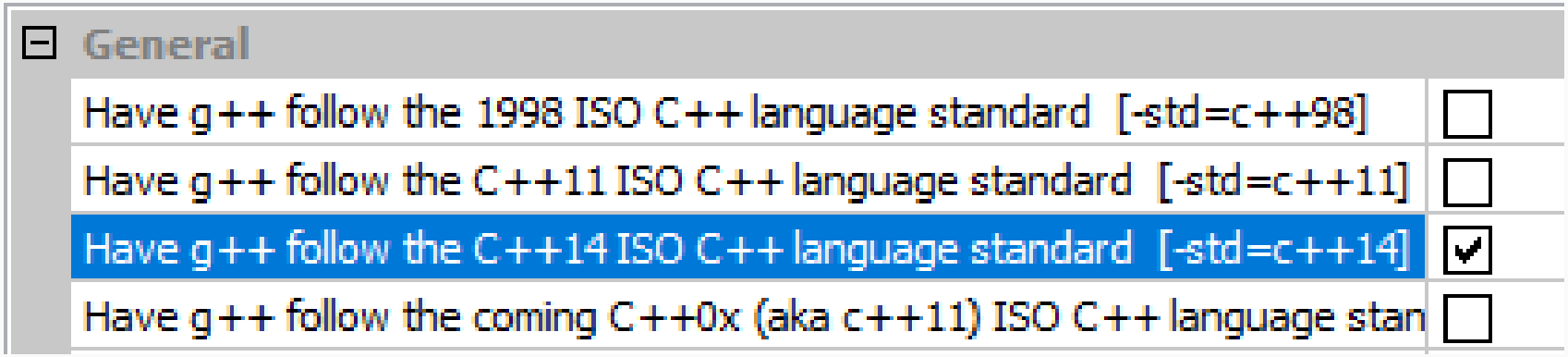
1985 → 2014


MONTH
DAY
YEAR
HOUR
MIN
JAN
17
2014
AM
PM
05:00
DESTINATION TIME
MONTH
DAY
YEAR
HOUR
MIN
JAN
17
2014
AM
PM
10:53
PRESENT TIME
MONTH
DAY
YEAR
HOUR
MIN
NOV
12
1985
AM
PM
04:07
LAST TIME DEPARTED

1985 → 2014


Code::Blocks
Compiler settings
Global compiler settings
Global compiler settings
Profiler settings
Batch builds
Selected compiler
GNU GCC Compiler
Set as default
Copy
Rename
Delete
Reset defaults
Compiler settings
Linker settings
Search directories
Toolchain executables
Custom variables
Build options
Policy:
Compiler Flags
Other compiler options
Other resource compiler options
#defines
General
Have g++ follow the 1998 ISO C++ language standard [-std=c++98]
Have g++ follow the C++11 ISO C++ language standard [-std=c++11]
Have g++ follow the C++14 ISO C++ language standard [-std=c++14]
Have g++ follow the coming C++0x (aka c++11) ISO C++ language stan
Have g++ follow the coming C++1y (aka C++14) ISO C++ language star
Have g++ follow the coming C++1z (aka C++17) ISO C++ language star
Have gcc follow the 1990 ISO C language standard (certain GNU extensio
Have gcc follow the 1999 ISO C language standard [-std=c99]
Have gcc follow the 2011 ISO C language standard [-std=c11]
In C mode, this is equivalent to -std=c90, in C++ mode, it is equivalent to
Position Independent Code [-fPIC]
Static libgcc [-static-libgcc]
NOTE: Right-click to setup or edit compiler flags.
OK
Cancel

1985 $\rightarrow$ 2014



Vecteurs

Tableau + taille

```
const int MAX = 100;

int tab[MAX] = {1, 2, 3, 4};
int nbE = 4;
```

Vecteurs

- vecteur = tableau redimensionnable
- “connaît” sa propre taille
- possible de rajouter et de supprimer des éléments
- utilisé comme un tableau
- pas de nombre maximal d’éléments

Vecteurs

```
vector<int> vec = {1, 2, 3, 4};
```

Vecteurs

```cpp
#include <iostream>
#include <vector>

using namespace std;

int main()
{
    vector<int> vec = {1, 2, 3, 4};

    cout << "premier element : " << vec[0] << endl;
    cout << "taille de vec : " << vec.size() << endl;
}
```

Vecteurs

Rajout d'un élément

- `vec` avant rajout :

1	2	3	4

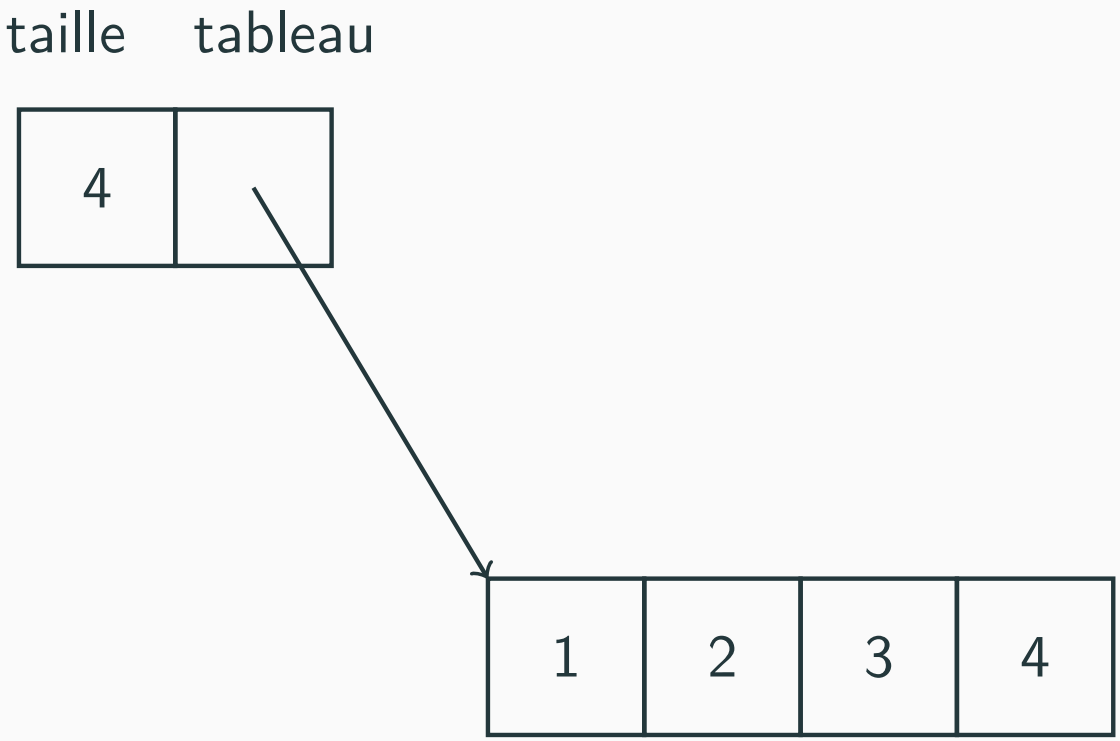
- rajout :

```
vec.push_back(5);
```

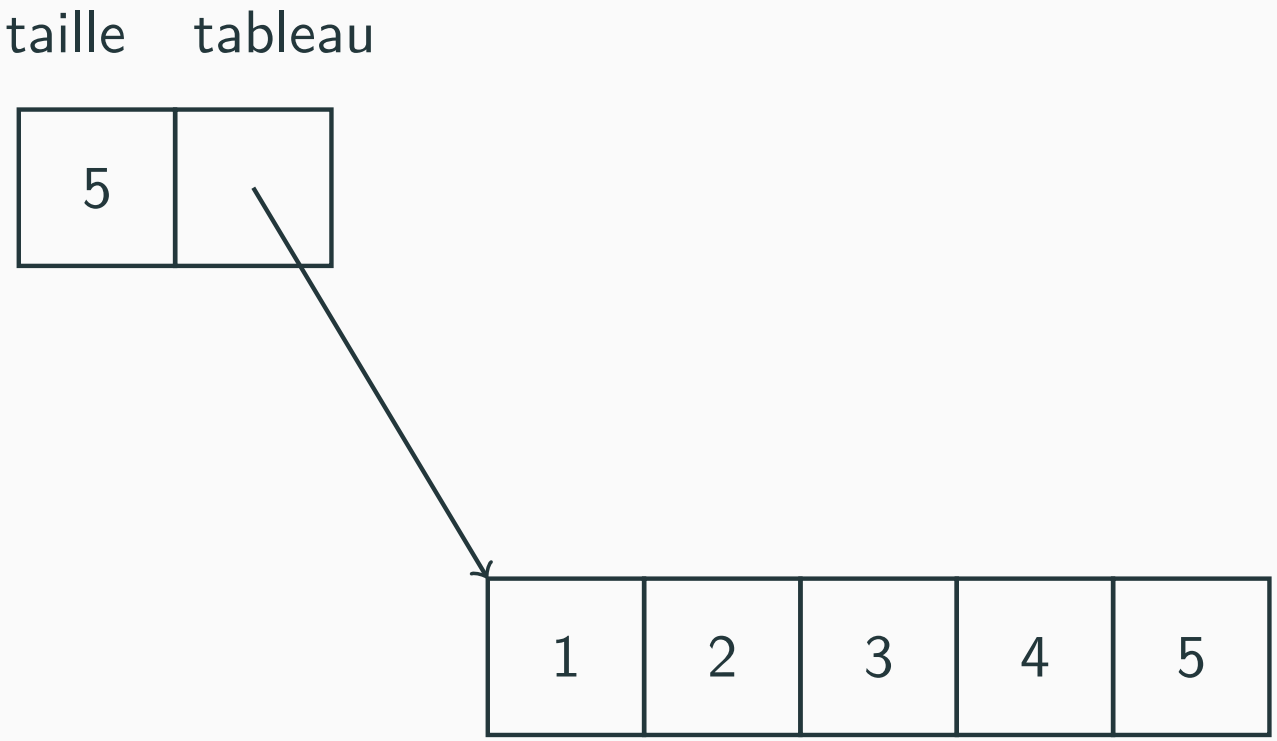
- `vec` après rajout :

1	2	3	4	5

Vecteurs



Vecteurs



Vecteurs

Accès à un élément

```
vector<int> vec = {1, 2, 3, 4};
vec[0];
vec[4];
```

Vecteurs


Pro Evolution Soccer 2016
Pro Evolution Soccer 2016 a cessé de fonctionner
Un problème a fait que le programme a cessé de fonctionner correctement. Windows va fermer ce programme et vous indiquer si une solution est disponible.
Fermer le programme

Vecteurs

Accès vérifié à un élément

- “`vec.at(i)`” = “`vec[i]`” si `i` est un indice valide
- si `i` n’est pas valide, alors `vec.at(i)` lance une exception
- le comportement de `vec[i]` n’est pas défini si `i` n’est pas valide
- `vec.at(i)` contient une vérification d’indice, `vec[i]` n’en contient pas

Vecteurs

Suppression d'un élément

- suppression du dernier élément : `vec.pop_back()`

Insertion et suppression à un indice arbitraire

- insertion de `val` à l'indice `i` :

```
vec.insert(vec.begin() + i, val);
```

- suppression de l'élément à l'indice `i` :

```
vec.erase(vec.begin() + i);
```

- `vec.begin()` est une *position* (ou un *itérateur*) qui pointe vers le premier élément du vecteur `vec`

Vecteurs

- `vec` avant suppression :

1	2	3	4

- suppression :

```
vec.erase(vec.begin() + 2);
```

- `vec` après suppression :

1	2	4

Vecteurs et fonctions

- peuvent être retournés d'une fonction
- peuvent être passés à une fonction en paramètre
- pour éviter de copier des vecteurs grands : passer en *référence*
- ou en référence *constante* si on ne veut pas changer le vecteur

Vecteurs

- passage par copie :

```
void f(vector<int> vec)
```

- passage par référence :

```
void f(vector<int> &vec)
```

- passage par référence constante :

```
void f(const vector<int> &vec)
```

Premières feuilles d'exercices

Premières feuilles d'exercices

- Mini-feuille 1, date limite de rendu sous Moodle : 16 novembre 2018
- Feuille 1, date limite de rendu sous Moodle : 23 novembre 2018
- traduction anglaise à partir de la semaine prochaine

Questions