M1102 - Amphi C++ - Complément 1

Alain Casali Marc Laporte

Aix Marseille Univ

Pré-requis

```
#include <vector>
using namespace std;
```

Identificateur

Modèle général algorithmique :

```
tableau_de UnType;
```

Modèle général C++ :

```
vector <Type>
```

Exemple

```
vector <int> VInt;
vector <float> VFloat;
```

La taille du vecteur est nulle lors de sa création

Taille d'un vecteur

Pour connaître la taille d'un vector (), on appelle la méthode () size () size () sur l'objet (la variable) de type vector (même nom que pour connaître la taille d'une string ()).

Exemple

```
cout << VInt.size ();   // 0
```

Insertion en fin de vecteur

Pour insérer des éléments compatibles en fin d'un vector, on appelle la méthode () push_back () sur l'objet (la variable) de type vector (cette méthode existe aussi pour les string). Le vecteur est redimensionné automatiquement.

Exemple

```
VInt.push_back (0);
VInt.push_back (1);
...
VInt.push_back (9);
cout << VInt.size ();   //10
```

- Même exemple que précédemment en utilisant une boucle `for` :

Exemple

```
for (int InsVal (0); InsVal < 10; InsVal = InsVal + 1)
{
    VInt.push_back (InsVal);
}
cout << VInt.size ();  //10
```

- Insertion de 10 valeurs saisies au clavier :

Exemple

```
int InsVal;
for (unsigned NbVal = 0; NbVal < 10; NbVal = NbVal + 1)
{
    cin >> InsVal;
    VInt.push_back (InsVal);
}
cout << VInt.size ();  //10
```

Redimensionnement d'un vecteur

Pour redimensionner un `vector`, on appelle la `méthode () resize () sur l'objet` (la variable) de type `vector`.

Exemple

```
VInt.resize (42);
```

Si la taille de l'ancien vecteur $<$ taille du nouveau (i.e. on ne fait pas de troncature), alors les nouvelles valeurs sont :

Type	Valeur par défaut
Entier	0
Réel	0.0
`string`	chaine vide
`char`	'\0'

Accès à un élément en lecture / écriture

On accède au $(i+1)^{ème}$ élément d'un `vector` en utilisant la notation `[i]`.

- La première case de lu `vector` a pour indice 0 ;
- La dernière case du `vector` a pour indice `VarIdent.size () - 1` ;
- Aucun contrôle quant à la validité de l'indice n'est effectué lors d'un accès à une case, que ce soit en lecture ou en écriture.

Exemple : Affichage du contenu d'un vecteur

```
for (unsigned i = 0; i < VInt.size (); i = i + 1)
{
    cout << VInt [i];
}
```

Et surtout pas `cout « VInt;`

La boulette du débutant

Exemple

```
vector <int> VInt;
VInt [0] = 0;
```

Compilation : pas d'erreur
Exécution : Segmentation fault : 11 (anciennement écran bleu)

A l'aide

`VInt` est déclaré, mais non dimensionné

$\Rightarrow$ La case d'indice 0 n'existe pas!!

Recopie d'un tableau dans un autre

On souhaite recopier le contenu de `VInt` dans un second tableau (`VInt2`)

```
vector <int> VInt2;
```

- Solution 1 :

```
for (unsigned i = 0; i < VInt.size (); i = i + 1)
{
    VInt2.push_back (VInt[i]);
}
```

- Solution 2 :

```
vector <int> VInt2;
VInt2 = VInt;
// vector <int> VInt2 = VInt;
```

- Solution 3 (uniquement lors de la déclaration) :

```
vector <int> VInt2 (VInt);
```

Initialisation d'un tableau (1)

Fixer la dimension du tableau lors de la déclaration :

```
vector <Type> VarIdent (Size);
```

Exemple

```
vector <int> VInt (42);
```

- `VInt` est un vecteur de 42 entiers ;
- Chaque case du vecteur est initialisée avec les valeurs par défaut liées au type (voir transparent précédent).

Cette déclaration est équivalente à :

```
vector <int> VInt;
VInt.resize (42);
```

Initialisation d'un tableau (2)

Fixer la dimension du tableau lors de la déclaration ET l'initialiser à des valeurs différentes que celles par défaut lors de la déclaration :

```
vector <Type> VarIdent (Size, DefaultValue);
```

Exemple

```
vector <int> VInt (42, -1);
```

- `VInt` est un vecteur de 42 entiers ;
- Chaque case a pour valeur -1.

Cette déclaration est équivalente à :

```
vector <int> VInt;
VInt.resize (42);
for (unsigned i (0); i < VInt.size (); i = i + 1)
{
    VInt[i] = -1;
}
```

Initialisation d'un tableau (3)

Initialiser le tableau à l'aide d'un agrégat

```
vector <Type> VarIdent {Val1, Val2, ... ,ValN};
```

- '{' marque du début d'agrégat ;
- '}' marque du fin d'agrégat.
- Le tableau est automatiquement dimensionné au nombre de valeurs présentes dans l'agrégat ;
- La $i^{ème}$ valeur de l'agrégat est stockée dans la $i-1^{ème}$ case du tableau.

Exemple

```
vector <int> VInt {0, -1, 2, -3};
```

Cette déclaration est équivalente à :

```
vector <int> VInt (4);
VInt[0] = 0; VInt[1] = -1; VInt[2] = 2; VInt[3] = -3;
```

Boucle `for` pour les collections

En algo. on aimerait avoir une boucle de parcours de collection du type :

```
pour_chaque (Valeur dans IdTableau)
faire
    ...
ffaire
```

Où `Valeur` prend les valeurs successives des cases de `IdTableau`.
Cette boucle existe en C++ :

```
for ([const] Type & VarIdent : CollectionVarIdent)
{
    ...
}
```

- `const` **DOIT** être présent si aucune modification d'une des cases de la collection ;
- `VarIdent` et `CollectionVarIdent` doivent être de type compatible.

Exemples

Afficher le contenu d'un vecteur

Exemple

```
for (const int & Val : VInt)
{
    cout << Val << '\t';
}
```

Saisir le contenu d'un vecteur

Exemple

```
for (int & Val : VInt)
{
    cin >> Val;
}
```