Cours d'introduction à la programmation (en C++)

`typedef`

Jean-Cédric Chappelier
Jamila Sam
Vincent Lepetit

Faculté I&C

Alias de types

Pour des types composés complexes, dont l'*utilisation directe est difficile*, on peut utiliser la commande `typedef` pour

donner un autre nom (alias) à ce type

Syntaxe : `typedef type alias;`

où *alias* est le nouveau nom de type et *type* un type élémentaire ou composé.

Exemples :

```
typedef vector<double> Vecteur;

typedef vector<Vecteur> Matrice;

Matrice rotation(3, Vecteur(3, 1.0));
```

Alias de types

De telles définitions de nouveaux noms de types sont particulièrement utiles, pour :

- *bien définir les types* des objets que l'on manipule
  Exemple :

```
typedef int Distance;

Distance ma_longueur(0);
```

  - meilleure identification des « concepts »
    Si tout est `int`, on ne distingue plus les `distance`s des `volume`s, des `couleur`s, etc.
  - changements ultérieurs de types plus faciles
    par exemple, les distances deviennent des `double`
- les *paramètres de fonctions*
- les déclarations de *tableaux*

Alias de types

De telles définitions de nouveaux noms de types sont particulièrement utiles, pour :

- *bien définir les types* des objets que l'on manipule
- les *paramètres de fonctions*

  Écriture plus claire, plus compacte et plus systématique

  Exemple :

```
typedef vector<double> Vecteur;

Vecteur produit_vectoriel(Vecteur, Vecteur);
```

- les déclarations de *tableaux*

Alias de types

De telles définitions de nouveaux noms de types sont particulièrement utiles, pour :

- *bien définir les types* des objets que l'on manipule
- les *paramètres de fonctions*
- les déclarations de *tableaux*

idem (améliore la lecture, l'écriture et la manipulation)

Exemples :

```
typedef vector<double> Vecteur;

typedef vector<Vecteur> Matrice;

Matrice rotation(3, Vecteur(3, 1.0));
```

```
typedef vector<Personne> Peuple;
```