TP 13 : Foncteurs et Lambdas

Imad Kissami

29 Avril 2025

Instructions

— Tous les fichiers doivent être regroupés dans un dossier `TP12_Nom_Prénom`.
— Créer un `Makefile` pour compiler tous les fichiers `.cpp`.
— Tous les fichiers doivent être regroupés dans un dossier

Objectif

— Comprendre et utiliser les foncteurs personnalisés en C++.
— Maîtriser les expressions lambda simples et capturant par valeur ou par référence.
— Passer des lambdas comme arguments de fonctions avec `std::function`.
— Utiliser les algorithmes de la STL (`for_each`, `transform`, `sort`).

Exercice 1 : Création d'un foncteur simple

Créez un foncteur `Additionneur` qui ajoute une valeur fixe à chaque nombre fourni.

Instructions :

— Le foncteur possède un attribut privé `valeur` donné par son constructeur.
— La surcharge de l'opérateur `()` doit afficher la somme.
— Testez avec un `std::vector<int>` et `std::for_each`.

Résultat obtenu :

```
11 12 13 14 15
```

Exercice 2 : Lambda pour filtrer un vecteur

Implémentez une fonction `filtrer` qui prend un `std::vector<int>` et un prédicat sous forme de lambda.

Instructions :

— Le prédicat est passé en argument en utilisant `std::function<bool(int)>`.
— Le prédicat doit être une lambda capturant par valeur.
— Affichez uniquement les éléments validés par le prédicat.

Résultat obtenu :

Pour un vecteur {10,15,20,25,30,50,75} :
- Prédicat : 'x > 20'

```
25 30 50 75
```

- Prédicat : 'x % 2 == 0'

```
10 20 30 50
```

Exercice 3 : Modifier un vecteur avec std ::transform et lambda

Appliquez une transformation sur un vecteur d'entiers en utilisant `std::transform`.

Instructions :

— La transformation ajoute 5 à chaque élément du vecteur.
— Utilisez une lambda qui capture les variables externes par référence si nécessaire.
— Affichez le contenu final du vecteur.

Résultat obtenu :

Pour un vecteur {10,12,15,17} :

```
15 17 20 22
```

Exercice 4 : Trier des objets avec lambdas

Créez une classe `Livre` ayant un titre et un nombre de pages.

Instructions :

- Définissez `operator<<` pour afficher un `Livre`.
- Triez un `std::vector<Livre>` selon le titre (ordre alphabétique), puis selon le nombre de pages (ordre décroissant).
- Utilisez `std::sort` avec des lambdas.

Résultat obtenu :

À partir des livres :

“‘ C++ Moderne - 350 pages Apprendre Python - 500 pages Algorithmique - 450 pages “‘

- Tri par titre :

```
Algorithmique - 450 pages
Apprendre Python - 500 pages
C++ Moderne - 350 pages
```

- Tri par nombre de pages décroissant :

```
Apprendre Python - 500 pages
Algorithmique - 450 pages
C++ Moderne - 350 pages
```