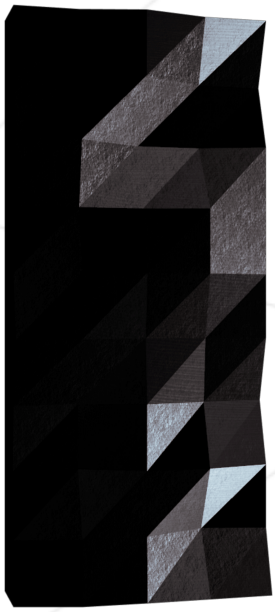

ft_containers

Les containers c'est facile !

Résumé: Ce projet est une introduction à la STL et aux containers

Table des matières

Chapitre I

Règles Générales

- Toute fonction déclarée dans une header (sauf pour les templates) ou tout header non-protégé, signifie 0 à l'exercice.
- Tout output doit être affiché sur stdout et terminé par une newline, sauf autre chose est précisé.
- Les noms de fichiers imposés doivent être suivis à la lettre, tout comme les noms de classe, les noms de fonction, et les noms de méthodes.
- Rappel : vous codez maintenant en `C++`, et plus en `C`. C'est pourquoi :
  - Les fonctions suivantes sont **INTERDITES**, et leur usage se soldera par un 0 : `*alloc`, `*printf` et `free`
  - Vous avez l'autorisation d'utiliser à peu près toute la librairie standard. CEPENDANT, il serait intelligent d'essayer d'utiliser la version C++ de ce à quoi vous êtes habitués en C, plutôt que de vous reposer sur vos acquis.
  - L'objectif de ce projet est de vous faire recoder les containers de la STL. Il vous est évidemment interdit de les utiliser.
- L'utilisation d'une fonction ou mécanique explicitement interdite sera sanctionnée par un `0`
- Notez également que sauf si la consigne l'autorise, les mot-clés `using namespace` et `friend` sont interdits. Leur utilisation sera punie d'un `0`.
- Les fichiers associés à une classe seront toujours nommés `ClassName.cpp` et `ClassName.hpp`, sauf si la consigne demande autre chose.
- Vous devez lire les exemples minutieusement. Ils peuvent contenir des prérequis qui ne sont pas précisés dans les consignes.
- Vous n'êtes pas autorisés à utiliser des librairies externes, incluant `C++11`, `Boost`, et tous les autres outils que votre ami super fort vous a recommandé
- Vous allez surement devoir rendre beaucoup de fichiers de classe, ce qui peut paraitre répétitif jusqu'à ce que vous appreniez a scripter ca dans votre éditeur de code préferé.

- Lisez complètement chaque exercice avant de le commencer.
- Le compilateur est `clang++`
- Votre code sera compilé avec les flags `-Wall -Wextra -Werror`
- Chaque include doit pouvoir être incluse indépendamment des autres includes. Un include doit donc inclure toutes ses dépendances.
- Il n'y a pas de norme à respecter en `C++`. Vous pouvez utiliser le style que vous préferez. Cependant, un code illisible est un code que l'on ne peut pas noter.
- Important : vous ne serez pas noté par un programme (sauf si précisé dans le sujet). Cela signifie que vous avez un degré de liberté dans votre méthode de résolution des exercices.
- Faites attention aux contraintes, et ne soyez pas fainéant, vous pourriez manquer beaucoup de ce que les exercices ont à offrir
- Ce n'est pas un problème si vous avez des fichiers additionnels. Vous pouvez choisir de séparer votre code dans plus de fichiers que ce qui est demandé, tant qu'il n'y a pas de moulinette.
- Même si un sujet est court, cela vaut la peine de passer un peu de temps dessus afin d'être sûr que vous comprenez bien ce qui est attendu de vous, et que vous l'avez bien fait de la meilleure manière possible.

Chapitre II

Introduction

Dans ce projet, vous allez devoir ré-implémenter les différents containers de la STL (standard template library).
Pour chaque container, vous devez rendre des fichiers de classe nommés correctement.
Vous utiliserez le namespace `ft`, et vos containers seront appellés à l'aide de ft : :<container>.
Vous devez respecter la structure de votre container de reference. N'implementez que les élements présents.
Nous vous rapellons que vous codez en C++98, donc toutes les nouvelles fonctions des containers **NE DOIVENT PAS** être implémentées. Vous devez par contre implémenter les anciennes (même deprecated).

Chapitre III

Mandatory part

- Implémentez les containers suivants, et rendez les fichiers `<container>.hpp` nécessaires.
- Vous devez également rendre un `main.cpp` qui teste votre rendu en vue des évaluations.
- Vous n'avez pas a implémenter `get_allocator`. Le reste est demandé, incluant les overloads non-membres.
- Si votre fonction utilise un système d'itérateur, vous devez le ré-implémenter.
- Vous pouvez utiliser http ://www.cplusplus.com/ en reférence de quoi implémenter.
- Vous ne devez pas avoir plus de fonctions/variables publiques que celles proposées dans les containers standard. Le reste doit être protected/private.
- Pour les overloads non-membres, le mot-clé `friend` est autorisé. Chaque utilisation de `friend` doit être justifié et sera vérifié pendant les évaluations

Vous devez rendre les containers suivants et leur fonctions associés :

- List
- Vector
- Map
- Stack
- Queue

Bien entendu, la STL est interdite. Vous pouvez par contre utiliser la STD.

Chapitre IV

Partie bonus

Si vous avez complètement fini la partie obligatoire, vous pouvez essayer de rendre la partie bonus. Réimplémentez les containers suivants :

- Deque
- Set
- Multiset
- Multimap