



Peer-class Containers

Ou comment faire containers sans se perdre dans
une boucle de l'enfer



De quoi ca parle?

- 1 – Comment commencer containers?
- 2 – Coder un container pas a pas
- 3 – Conseils divers et varies
- 4 – Questions/reponses



- 1 - Comment commencer containers?

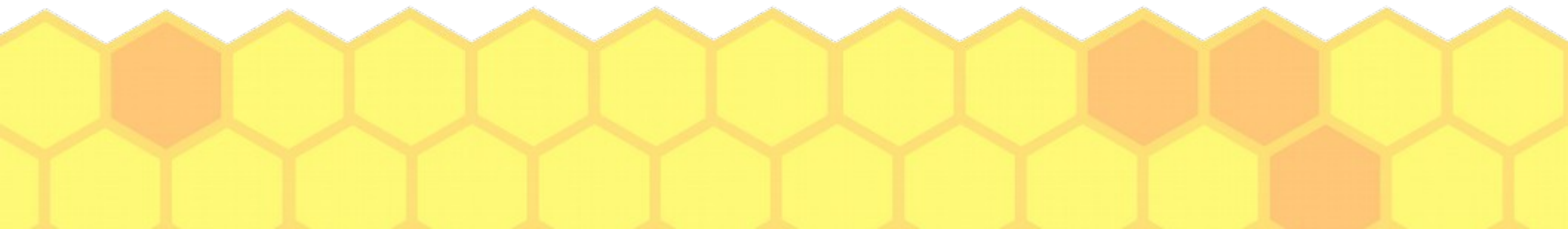
A- Stack, iterator_traits, reverse_iterators et les classes supplementaires a implementer (sujet page 5)

- Coder stack avec `std::vector` (a remplacer par `ft::` quand vector est termine)

- `iterator_traits` et `iterators` : chapitre 24 de la norme ISO



- tests possibles pour certaines classes (ex : `is_integral`), impossibles pour d'autres (ex : `enable_if`) > a verifier a l'usage pendant le projet
- Doc : norme ISO / cppreference et cplusplus (liste non exhaustive)
- Temps estime 5 jours



B- Vector

Temps estime : 3 semaines

C- Map

Temps estime : 3 semaines

D- Set (si red and black tree)

Temps estime : 1 journee




2- Coder pas a pas un container

Materiel :

- la norme ISO (chap. 23.2 pour stack et vector, chap. 23.3 pour map et set)
- un editeur (vscode, vim, ...)
- le sujet (on l'a tout.e.s fait de pas relire et de zapper des points importants!)

de la perseverance, de la motivation et un brin de bonne humeur (on va en avoir besoin, on cherche tous les petits trucs positifs et on lache rien)



Vector/map/set

- 0- Ne pas essayer de coder tout mon container en une fois
- 1- Prendre le chapitre dans la norme et copier le synopsis
- 2- Commenter toutes les fonctions et ne garder que le constructeur par défaut, le destructeur et la structure. Finaliser la structure (ex : implementation des itérateurs)



- 3- Prendre un main de base (merci cpp reference) et compiler (enfin essayer...)
- 4 Quand ca compile , commencer a implementer les autres fonctions en les testant au fur et a mesure.



Map

- Necessite un AVL ou un RBT (implementer un arbre binaire pour commencer puis l'équilibrer quand il fonctionne)
- Avantage RBT : le bonus (set) sera presque fait
- Coder les `bidirectional_iterators`
- Doc : Introduction to algorithm, 3rd edition chap. III-13 (regles du rbt, fonctionnement et pseudo-code insert et erase)



Set

- Reprendre le RBT et les iterators de map, adapter si nécessaire (on passe d'une paire d'elements a un seul)
- Copier le synopsis de set et reprendre l'implementation de map avec quelques differences : template et fonctions en moins dans set



3- Conseils divers et varies

- Faire les testeurs en meme temps qu'on fait ses fonctions (attention : prevoir une sortie ft / une std)
- Comment configurer les testeurs mli et mazoises
- Demander de l'aide !
- Et surtout, on garde le moral! :sun_with_face:



Doc

- Norme ISO/IEC 14882-1998 (norme C++)
- Introduction to algorithms, 3rd edition
- Sujet ft_containers
- Sites en vrac : cplusplus, cppreference, geekforgeeks, ...
- Testeurs : mli, mazoises, jgiron, ...
- Notion :

<https://juzain.notion.site/containers-298ba19c71614fc58045ca47f25da0e8>