

TP POO C++ n°2b - La STL

© 2013 tv <tvaira@free.fr> - v.1.0

Sommaire

Rappels	1
Travail demandé	1
Les tableaux dynamiques (vector)	1
Les listes (list)	2
Les tables associatives (map)	2

Les objectifs de ce tp sont de découvrir la STL en C++.

Rappels

La STL est la librairie standard du C++ qui implémente de nombreux types de données de manière efficace et générique.

Dans un programme C++, on privilégiera toujours l'utilisation de la STL par rapport à une implémentation manuelle : gain en efficacité, robustesse, facilité, lisibilité car standard.



Lire le document de cours : [conteneurs-stl.pdf](#)

Travail demandé

Les tableaux dynamiques (**vector**)

Dans cet exercice, on va réaliser un programme `vecteur.cpp` qui permet de stocker des `std::string` et manipuler ceux-ci de manière simple.

Question 1. Vous devez déclarer un vecteur de nom `monVecteur` stockant des `std::string`. Ajoutez 5 `std::string` dans le vecteur qui vaudront respectivement "bonjour", "comment", "allez", "vous", "?".

Question 2. Ajoutez 5 `std::string` dans le vecteur qui vaudront respectivement "bonjour", "comment", "allez", "vous", "?".

Question 3. Affichez la taille de votre vecteur. Afficher sa capacité (`capacity`). Quel est la différence avec sa taille ?

Question 4. Afficher le contenu du vecteur en utilisant la notation indexée de tableau en accédant à la valeur avec une syntaxe du type : `monVecteur[k]`.

Question 5. Afficher le contenu du vecteur en utilisant les itérateurs (`iterator`) sur votre vecteur.

Question 6. Réaliser un échange entre le contenu de la case d'indice 1 et le contenu de la case d'indice 3 de votre vecteur (vérifiez votre résultat en affichant le vecteur). Notez l'existence de `std::swap`.

Question 7. Trier le vecteur en utilisant un algorithme de la STL (inclure l'en-tête `algorithm`). L'ordre de tri par défaut est celui de la comparaison sur des `std::string`. Afficher le résultat obtenu.

Question 8. Créer une fonction `affiche` qui affiche le contenu du vecteur passé en paramètre. Chaque élément sera espacé d'un 'espace' à l'affichage. Notez qu'ici, on passera le vecteur sous forme de référence constante car il n'a pas à être modifié, ni copié.

Question 9. Créer une fonction `concatene` qui concatène l'ensemble des éléments du vecteur dans une seule variable de type `std::string`. Chaque élément sera espacé d'un 'espace' dans la `std::string`. Réfléchir au prototype de votre fonction, sous quelle forme vous passez le paramètre d'entrée, sous quelle forme retournez-vous le `std::string` ?

Question 10. Insérer la valeur "a tous" après le premier élément dans votre vecteur. Vérifier votre résultat.

Question 11. Changer le point d'interrogation "?" par "!". Vérifier votre résultat.

Les listes (`list`)

Éditer un fichier source `liste.cpp` qui permettra de :

Question 12. Créer une liste de 30 entiers (les 30 premiers entiers impairs).

Question 13. Afficher votre liste.

Question 14. Supprimer le troisième élément.

Question 15. Afficher à nouveau votre liste.

Question 16. Insérer les 20 entiers impairs suivants.

Question 17. Afficher à nouveau votre liste.

Les tables associatives (`map`)

Éditer un fichier source `map.cpp` qui permettra de construire une `std::map` dont la clé est une chaîne de caractère, et la valeur est un nombre flottant.

Question 18. Remplir cette `map` avec un exemple de type "facture de restaurant" (nom du plat et prix du plat). Afficher le contenu de la `map`.

Question 19. Afficher le contenu de la `map`.

Question 20. Calculer et afficher le prix total des plats présents dans la `map`.