TP5 : Tableau dynamique

Dans ce TP, vous allez implémenter en C++ un module TableauDynamique qui devra être utilisable par d'autres. Dans cette implémentation, les cases du tableau seront numérotées à partir de 0. Vous allez créer votre classe sous forme de type de données abstrait, c'est-à-dire avec l'interface (fichier .h) séparée de l'implémentation (fichier .cpp). Ce sera aussi l'occasion pour vous d'écrire votre premier Makefile et de vous entraîner à bien valider vos procédures et fonctions membres au fur et à mesure de leur implémentation.

Exercice 1 : Création et destruction, première compilation multi-fichiers

a. Le type TableauDynamique doit permettre de stocker des éléments de type ElementTD. Vous trouverez les fichiers ElementTD.h, ElementTD.cpp et TableauDynamique.h sur le site de l'UE. Dans votre répertoire LIFAPSD, créez un sous-répertoire TP5 et enregistrez ces fichiers dedans.

b. Examinez le contenu du fichier TableauDynamique.h : il contient les prototypes (= déclarations) des fonctions et procédures membres promis par le module.

c. Créez un nouveau fichier « TableauDynamique.cpp », destiné à contenir l'implémentation de votre module. Dans ce fichier, commencez par indiquer que vous allez utiliser les types et sous-programmes déclarés dans les fichiers TableauDynamique.h et ElementTD.h, grâce à la directive de précompilation `#include`. Ecrivez ensuite la **définition** du constructeur par défaut (initialisation à une case vide), et celle du destructeur (libération de la mémoire allouée sur le tas) en respectant bien les prototypes déclarés dans le .h. Enregistrez votre fichier mais ne le fermez pas.

d. Créez un nouveau fichier « main.cpp » et écrivez un programme principal qui appelle les procédures que vous venez d'écrire, afin de tester si elles fonctionnent bien. Enregistrez votre fichier mais ne le fermez pas.

e. Compilez votre programme en tapant successivement les commandes suivantes (ne passez pas à la commande 2 tant que vous avez des erreurs sur la commande 1, etc) :

```
g++ -Wall -c ElementTD.cpp
g++ -Wall -c TableauDynamique.cpp
g++ -Wall -c main.cpp
g++ TableauDynamique.o main.o ElementTD.o -o monprog.out
```

Notez bien l'option –c dans les 3 premières commandes : elle indique à gcc de s'arrêter à l'étape de traduction de votre code en langage binaire, sans essayer de créer un exécutable. C'est la 4ème commande qui crée l'exécutable en éditant les liens entre les 3 fichiers .o et avec les bibliothèques (ex. iostream).

Exercice 2 : Création d'un fichier Makefile et utilisation de la commande make

Créez un nouveau fichier Makefile (sans extension), qui permettra de compiler automatiquement le programme, en tapant simplement `make` au lieu des 4 commandes précédentes. Inspirez vous de l'exemple vu en cours magistral.

Exercice 3 : Implémentation des fonctionnalités de base du module

Définissez dans TableauDynamique.cpp les fonctionnalités ci-dessous, en respectant les prototypes déclarés dans le .h. Remarque importante : Testez *au fur et à mesure* que vous ajoutez une fonctionnalité : appelez-la dans le main, enregistrez tous vos fichiers, recompilez en utilisant la commande `make` et exécutez le programme.

a. vider le tableau,

b. ajouter un élément en fin de tableau,

c. accéder à l'élément de la case i,

d. modifier l'élément de la case i,

e. afficher le tableau,

f. supprimer un élément.

Exercice 4 : Expérimenter la complexité du remplissage du tableau

a. Le but de cet exercice est de vérifier que le coût amorti d'un ajout dans un TableauDynamique est de l'ordre de 3 affectations. Pour cela, vous allez comparer le temps T_1 (en secondes) nécessaire pour faire n ajouts dans un TableauDynamique, avec le temps T_0 nécessaire pour faire n ajouts dans un tableau « simple », qui ne change pas de taille au fur et à mesure des ajouts (ce tableau statique simple doit donc être déjà de taille n au départ). Prenez $n = 500000$ pour cet exercice.

   Rappel : Pour mesurer le temps d'exécution, référez-vous à l'annexe C.

b. Mesurez les deux temps d'exécution pour n allant de 100 000 à 20 000 000 par pas de 100 000. En utilisant un tableur, tracez les deux courbes $T_0(n)$ et $T_1(n)$. Retrouvez-vous bien un coût 3 fois plus important pour le remplissage du tableau extensible comparé au tableau simple ?

c. Listez les avantages et inconvénients d'un TableauDynamique (extensible en fonction des besoins) comparé à un tableau simple (dont la taille ne change pas).

Exercice 5 : Fonctionnalités plus avancées

a. Testez dans votre main ce qu'il se passe lorsqu'on fait a = b quand a et b sont deux tableaux dynamiques, et une libération de la mémoire de b ensuite. Expliquez. Implémentez alors le constructeur par copie.

b. Implémentez la procédure membre d'insertion d'un élément en i-ème position.

c. Implémentez la procédure de recherche d'un élément dans le tableau trié. Vous pouvez commencer par une recherche linéaire, puis si vous avez le temps, optez pour une recherche dichotomique.

d. Implémentez la procédure membre de tri du tableau dynamique. Vous pouvez commencer par un tri par insertion ou sélection, puis si vous avez le temps, optez pour un tri par fusion interne ou externe.