POLYTECH LILLE

Université Lille1 Sciences et Technologies

Département IMA / 3A (S5) **Programmation Structurée 2011/2012**

http://laure.gonnord.org/pro/teaching/

TP5 - Tris de Tableaux, 1/2

Objectifs

Coder et comparer les différents tris vus en cours. Début de la compilation séparée.

IMPORTANT : à chaque fin de section, faites vérifier votre travail par un enseignant !

1 Unix - Wget / Tar

1. Ouvrez un Terminal et placez-vous dans *votre répertoire de travail pour la programmation structurée*, répertoire TP5.
2. Récupérez sur la page web du cours l'archive `tp5.tgz` *à la ligne de commande...*[1].
3. Désarchivez avec la commande :

   ```
   tar xvf tp5.tgz
   ```

   (x pour extraire, v pour verbose, f pour file). La création du répertoire `TP5` est normalement automatique.

2 Préparation : compilation de plusieurs fichiers

Au TP 4, nous avons créé un fichier contenant des fonctions de base sur les vecteurs d'entiers (initialisation avec des valeurs aléatoires, impression sur le terminal, ...).

L'archive fournit une correction des fonctions les plus importantes : le fichier `vecteurs.c` contient ces fonctions, et le fichier `vecteurs.h` fournit les déclarations de ces fonctions. Nous allons utiliser ses fonctions dans un troisième fichier (`tris.c`). Pour compiler tout cela, nous allons utiliser la méthode de *compilation séparée*.

1. Observer les fichiers `vecteurs.c` et `vecteurs.h`. Remarquer que le .c ne contient pas de fonction principale (*main*), et que le .h ne contient pas de code des fonctions, juste leurs signatures (entêtes).
2. Compiler `vecteurs.c` avec la commande :

   ```
   gcc -Wall -c vecteurs.c
   ```

   Cette commande crée un fichier `vecteurs.o` (vérifier), qui est un fichier objet (voir la feuille annexe sur la compilation)

1. Dans un navigateur, un clic droit sur le lien vers le fichier voulu enregistre l'adresse, qu'il suffit de coller dans un terminal après `wget` (clic du milieu avec la souris).

3. Créer et éditer le fichier `tris.c` (avec emacs), inclure le fichier d'entête (`#include "vecteurs.h"`), puis dans le programme principal(*main*) faites des appels aux fonctions d'initialisation et d'impression.
4. Compiler avec :

   ```
   gcc -Wall -c tris.c
   ```

   Et enfin, si la commande précédente s'est passée sans problème (avec génération de `tris.o`), compiler le binaire final avec :

   ```
   gcc -Wall -o tris vecteurs.o tris.o
   ```
5. Exécuter. Tester.
6. Modifier le fichier `tris.c` en ajoutant un `printf` par exemple. Quelle(s) commande(s) de compilation est-il suffisant de lancer ?
7. Compiler avec le Makefile fourni : taper `make` et observer. [2]
8. À ce stade, faites une archive compressée avec les fichiers .c et .h, Makefile et pas le reste :

   ```
   tar cvzf tp5_amoi.tgz Makefile vecteurs.h vecteurs.c tris.c
   ```

   puis vérifiez (dans un autre répertoire), que vous savez le décompresser.

3 Tris non graphiques

Dans le fichier `tris.c`, écrire et tester les tris du cours suivants (utiliser `make` pour compiler) :

1. Tri insertion.
2. Tri sélection.
3. Tri fusion.

On écrira une procédure par tri, et on testera dans le main, sur des tableaux de taille 10 initialisés au hasard.

2. Voir http://fr.wikipedia.org/wiki/Make