Système dôExploitation- TP1
CERI / L2

A déposer dans lôENT à la fin du TP

Micro projet, organisation, librairies, etc...

Lôobjectif du TP est de réaliser un mini-projet en C, affichant la somme puis la multiplication de 2 matrices dôentiers 3X3 lues au clavier (int Mat[3][3]).

Soient 2 matrices A, B et Plus = A+B et Mult = A*B

$$Plus[i][j] = A[i][j] + B[i][j]$$
$$Mult[i][j] = Somme_k\ (A[i][k] * B[k][j])$$

Pour cela, je vous demande dôutiliser la console de commandes et les commandes suivantes :

- **Lôéditeur vi :** pour écrire les fichiers.

- **Le compilateur gcc :**

`gcc –c Main.c` : va générer le fichier Main.o
`gcc –o MutlMat MatIO.o MatArith.o Main.o` : va générer le programme MutlMat avec les objets MatIO.o MatArith.o Main.o

Exercice 1 : Première Méthode

Le projet se composera de 5 fichiers contenant du code :

- MatIO.c : contient les définitions des fonctions pour lire une matrice, afficher une matrice.
- MatArith.c : contient les définitions des fonctions pour effectuer lôaddition et la multiplication de 2 matrices.
- MatIO.h et MatArith.h : contiennent les déclarations des fonctions. Aucune définition ne doit y figurer !!!
- Main.c : contient le programme principal « main » qui lit 2 matrices au clavier et affiche leur somme puis leur produit.

Le projet sera organisé de la façon suivante :
Un répertoire TP0 contiendra 4 sous-répertoires : src, include, obj, bin.

- Le répertoire **src** contiendra les sources du projet (.c)
- Le répertoire **include** contiendra le fichier .h
- Le répertoire **obj** contiendra les fichiers .o
- Le répertoire **bin** contiendra lôexécutable MultMat

Question 1 : Donner la liste des commandes permettant ces traitements (cp, mkdir, gcc, mv, etcé) :

Question 2 : Donner les sources des 5 fichiers :

Question 3 : Afin de ne pas taper les matrices à chaque exécution, créer un fichier texte contenant les 2 matrices et rediriger lœentrée du programme exécutable. Donner la commande.

Exercice 2 : Librairie Matrice.a

Nous souhaitons à présent faire une librairie nommée libMatrice.a contenant toutes les opérations que nous avons développées sur les matrices.
Pour créer une librairie, il suffit de regrouper à lœaide de la commande **ar** les fichiers objets qui doivent y figurer (dans notre cas, uniquement MatIO.o et MatArith.o).

Dans ce cas, le projet sera réorganisé :

- Un repertoire libMatrice contiendra 4 sous répertoires :
  - src qui contient MatArith.c et MatIO.c
  - obj qui contient MatArith.o et MatIO.o
  - include qui contient Matrice.h (regroupant les déclarations des fonctions figurant dans la librairie)
  - lib qui contient libMatrice.a
- Un repertoire TP0 contiendra 3 sous répertoires :
  - src qui contient Main.c
  - obj qui contient Main.o
  - bin qui contient MultMat

Question 4 : Donner la liste des commandes permettant ces traitements (cp, mkdir, gcc, mv, etcé) :

Question 5 : Donner les sources des fichiers modifiés pour lœoccasion :

Exercice 3 : Lœutilitaire Make

Lœutilitaire Make permet de définir des dépendances entre des fichiers ainsi que les commandes à lancer lorsque les dates ne sont pas cohérentes.
Un fichier, nommé par défaut « makefile » permet de définir ces dépendances ainsi que les commandes à lancer. Ce fichier se compose de lignes du type :

```
Main.o : Main.c Mat.h
<tab>gcc –o Main.o –c Main.c

Executale : Main.o
<tab>gcc –o Executable Main.o
```

Dans cet exemple, si un des fichiers Main.c ou Mat.h est plus récent que le fichier Main.o, la commande « gcc ï o Main.o ï c Main.c » sera exécutée, etcé .
Cet outil permet de nœexécuter que les commandes strictement nécessaires, sans se soucier de savoir quel fichier a été modifié ou pas, etc..
La commande : « make Executable » lancera les commandes nécessaires.

Question 6 : Faire un fichier makefile pour la construction de la librairie et un autre pour la construction du projet complet. Donner les contenus des 2 fichiers makefile.