Cours de programmation séquentielle

Réusinage et makefiles

1 Buts

- Introduction à la compilation séparée.
- Introduction à `make`.
- Création de librairies.
- Réusinage.

2 Énoncé

Reprendre le travail pratique des fractions. Résusiner le codes en le rendant aussi modulaire que possible. Séparez le code en fichiers `.h` et `.c`. Il faudra également séparer les fichiers en "librairies" contenant uniquement les fonctions publiques et en programme ne contenant que l'exécutable de votre projet (et donc la fonction `main()`). Inspirez vous de la structure proposée ci-dessous pour faire ce travail.

2.1 Exemple de travail à réaliser

Diviser le code des fractions en fichiers `.h` et `.c`. Mettre le programme dans un fichier `.c` séparé contenant uniquement la fonction `main()`. Ainsi par exemple pour les fractions, votre code doit être séparé en au moins quatre fichiers:

- `frations.h`
- `fractions.c`
- `calcul_de_pi.c`
- `calculatrice.c`

Les fichiers `fractions.h/c` contiennent la structure `fraction`, et toutes les fonctions de création, addition, multiplication, etc. agissant sur les fractions. Les prototypes et la structure se trouvent dans le `.h`, les implémentations dans le `.c`. Le fichier `calculal_de_pi.c` contient les fonctions de calcul de `pi` et compare les valeurs obtenues pour chaque méthode avec la valeur `M_PI` ou avec la valeur `4*atan(1)`. Le fichier `calculatrice.c` doit contenir le code de la partie bonus du travail pratique sur les fractions à savoir la lecture de la ligne de commande et l'appel aux fonctions d'addition. Ces deux derniers fichiers `.c` contiennent un `main()` à eux.

Dans votre `Makefile` créez des cibles `fractions.o`, `calcul_de_pi.o`, et `calculatrice.o` avec les dépendances nécessaires. Puis deux cibles exécutables `calcul_de_pi` et `calculatrice`. Pensez également à faire une cible `clean` sans

dépendances qui effacera les produits de la compilation (les fichiers `.o` et les exécutables)