I6 - TP Complexité et efficacité expérimentale
- Utilisation d'un Makefile -

Licence Math / Info S4 - 2007

Présentation

Le présent document ne constitue qu'une introduction brève au makefile (plus précisément un rappel de ce que vous avez vu en I2 lors de la première année), pour de plus amples détails vous pouvez vous reporter au manuel de GNU-make à l'adresse :

`http://www.gnu.org/software/make/manual/html_node/index.html`.

Les fichiers makefiles permettent d'automatiser des tâches récurrentes. Un fichier makefile se nomme en général "makefile" et contient un ensemble de lignes de la forme :

```
cible: dépendances
    commandes
```

où `cible` est le résultat de l'éxécution des `commandes`[1]. `dépendances` représente l'ensemble des dépendances nécessaires pour réaliser les commandes. Si une dépendance est une cible du fichier makefile, la ligne correspondante est alors évaluée. Une fois toutes les dépendances résolues la commande est exécutée (si il existe au moins une dépendance plus récente).

Exemple d'un programme en C

Soit le programme présentée sur la figure 1. Ce programme permet de remplir un tableau avec des en-

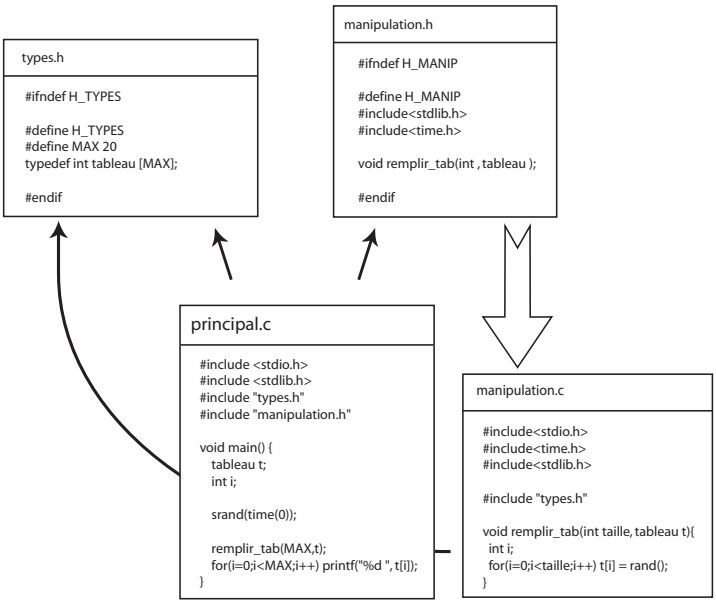

FIG. 1 – Programme permettant de remplir un tableau avec des entiers générés aléatoirement

tiers aléatoires. Le programme s'articule autour du fichier *principal.c* qui utilise les fonctions définies dans *manipulation.c* et les types de données définis dans *types.h*. Le fichier *manipulations.c* utilise aussi les types définis dans *types.h*, il doit donc inclure ce fichier. Dans le fichier *types.h* les lignes `#ifndef` et `#endif` garantissent de ne pas exécuter plusieurs fois les instructions comprises entre ces deux instructions (i.e. redéfinir plusieurs fois les même types), ce qui créerait des conflits.

[1] ATTENTION : les lignes de commandes sont précédées d'une tabulation

Dans le cadre de cet exemple nous considérons tous les fichiers dans le même répertoire. Un fichier makefile possible à associer à une telle structure est le suivant :

```
projet: principal.o manipulation.o
    gcc -o projet manipulation.o principal.o

manipulation.o: manipulation.c types.h
    gcc -o manipulation.o -c manipulation.c

principal.o: principal.c types.h manipulation.h
    gcc -o principal.o -c principal.c

clean:
    rm -rf *.o
```

Ce makefile doit se trouver dans le même répertoire que les sources auxquelles il fait référence. La première ligne de ce fichier est celle qui sera exécutée lorsque l'on tape "make" dans la console (penser à être dans le même répertoire que le fichier makefile). Cette ligne précise que pour générer le fichier projet, le compilateur a besoin de *principal.o* et *manipulation.o*, la commande associée est la commande de compilation classique. La deuxième (respectivement troisième) règle définit comment générer le fichier *manipulation.o* (respectivement *principal.o*) nécessaire à la génération de l'éxécutable projet.

Compiler

- Taper `make` lance la première instruction du fichier makefile (même effet que `make projet`).
- Taper `make clean` dans la console permet d'effacer les .o du dossier dans lequel se trouve le makefile.

1 Compléments

Comme ennoncé en début de ce document, ce dernier ne représente qu'une introduction sommaire au concept du makefile, avec les éléments présentés vous êtes capable de réaliser votre premier makefile. Cependant, il peut être interessant de regarder en détail la documentation de make à l'adresse `http://www.gnu.org/software/make/manual/html_node/index.html`.
Bien entendu d'autres ressources concernant les makefiles sont disponibles sur internet, vous n'aurez aucun problème à trouver des éléments pour étendre vos connaissances.