TP3

Compiler et plus

L'objectif de ce TP est de se familiariser avec les outils de développement du langage C : compilation, édition de liens, mise au point.

Remarque : la documentation des fonctions de la bibliothèque standard C peut se consulter avec la commande man.

1 Compilation, édition de liens

Programme mono-fichier

1. Copiez le fichier /share/esir1/sys/insert_sort.c dans votre répertoire de tp.
2. Regardez la documentation du compilateur C (*man gcc*) : lisez entièrement les sections *Synopsis* et *Description*.
3. Trouvez la signification des options -c, -o et -Wall ; vous pouvez aussi utiliser gcc --help.
4. Compilez le fichier source copié avec la commande : gcc -c -Wall fichier.c -o fichier.o.
5. Corrigez les erreurs dans le fichier source, y compris les avertissements (*warning*).
6. Recompilez et vérifiez quel est le fichier produit par la compilation.
7. Exécutez la commande gcc fichier.o -o fichier : que représente cette opération ?
8. Exécutez le programme généré à l'étape précédente
   — quelle commande permet de démarrer ce programme ?
   — comment faut-il lui fournir ses données ?
   — Si les résultats sont faux ou si le programme « plante », n'en tenez pas compte pour l'instant.
9. Supprimez tous les fichiers sauf le fichier source.
10. Dans le répertoire de ce tp, créez les répertoires *src*, *obj* et *bin* puis déplacez le fichier source dans *src*.
11. Les opérations suivantes doivent se réaliser *depuis le répertoire de ce tp* :
    — Compilez le fichier source ; la commande gcc doit placer le fichier objet dans le répertoire *obj* ;
    — Fabriquez le programme exécutable ; la commande gcc doit placer le fichier résultat dans le répertoire *bin* ;
    — Exécutez le programme

Programme multi-fichiers

On veut dans ce deuxième exercice conserver la même organisation de fichiers et répertoires que dans l'exercice précédent.

— Copiez *en une seule commande* le répertoire /share/esir1/sys/tpc/pile dans le répertoire de ce tp ; déplacez-vous dans le répertoire copié.
— Compilez chacun des fichiers sources puis fabriquez le programme exécutable (*pile*) de sorte que les fichiers .o soient dans le répertoire *obj* et l'exécutable dans le répertoire *bin*. Si la compilation échoue, c'est probablement qu'il faut dire au compilateur où trouver le fichier d'entête : regardez dans la documentation l'utilisation de l'option -I.
— Exécutez votre programme en lui donnant sur l'entrée standard le contenu du fichier *donnees.pile*.
— Modifiez la fonction d'affichage du programme client : quelles opérations devez-vous refaire pour obtenir un programme exécutable ?

Automatiser

Chaque fois qu'on modifie un fichier, il faut reconstruire tous les fichiers qui en dépendent (compilations puis édition de liens).

make est un outil standard Unix qui permet (entre autres) d'automatiser les opérations de compilation et d'édition de liens.

Vous pouvez consulter ce petit tutoriel : `https://perso.univ-rennes1.fr/jean-christophe.engel/make`.

1. Supprimez les fichiers .o et l'exécutable créés à l'étape précédente.
2. Complétez le *Makefile* fourni puis vérifiez son fonctionnement en exécutant la commande *make* : s'il est correct, vous devriez obtenir le même résultat qu'à l'étape précédente.
3. Tapez la commande *make test* : le programme devrait s'exécuter et lire ses données dans le fichiers *donnees.pile*.
4. Faites une modification dans l'un des fichiers sources, puis exécutez à nouveau la commande *make test* : si tout est correct, seul le fichier modifié doit être recompilé puis l'édition de liens sera effectuée et enfin le programme devrait s'exécuter.
5. Supprimez tous les fichiers générés (*make clean*) ; vérifiez.
6. Faites *make test* : si tout est correct, ceci devrait compiler tous les fichiers, faire l'édition de liens puis exécuter le programme. Comment pouvez-vous garantir que le programme que vous testez est à jour ?
7. Mettez dans la variable CFLAGS les options de compilation communes et remplacez ces options dans les commandes de compilations par $(CFLAGS) : ceci permet de modifier globalement les options de compilation de l'ensemble des commandes de compilation.
8. Complétez l'entrée de test pour faire vérifier la gestion mémoire du programme avec *valgrind*. Testez, puis corrigez l'erreur qui provoque la fuite ; reconstruisez puis testez à nouveau.

2 Mise au point avec gdb

gdb est l'outil standard de mise au point en ligne de commande.

Opérations élémentaires

gdb possède une documentation interne accessible avec la commande help. Vous pouvez aussi jeter un coup d'oeil sur le memento fourni (*/share/esir1/sys/tpunix/memento_gdb.pdf*).

- afficher la valeur d'une variable : print *v*
- afficher une expression C/C++ : print *expr*
- afficher les *n* premiers éléments d' un tableau à l'aide d'un pointeur : print **p@n*
- mettre un point d'arrêt sur une instruction :
  - break *numéro ligne* (dans fichier courant)
  - break *fonction*
- rendre conditionnel un point d'arrêt : cond N *condition* ; N est le numéro du point d'arrêt
- afficher la liste des points d'arrêt : info break
- afficher une expression à chaque arrêt : display *expr*
- arrête l'exécution dès qu'une expression change : watch *expr*

Regardez aussi la signification des commandes step, next, cont.

Exemple

Regardez la signification de l'option de compilation -ggdb puis utilisez-la pour recompiler le programme de tri par insertion. Faites l'édition de liens puis :

- démarrez gdb : *gdb bin/insert_sort*
- exécutez le programme avec une liste d'arguments : *run 45 89 3 64 777 -20*
- notez les informations affichées par gdb lors du plantage.
- affichez la pile des appels de fonction : backtrace
- affichez les variables locales : info locals
- si besoin, frame *n* permet de sélectionner l'une des fonctions dans la pile d'appel pour étudier ses variables locales

Vous pouvez placer un point d'arrêt à l'endroit qui vous parait judicieux, puis redémarrer le programme : la commande run sans argument réutilise les arguments précédents.

Quand vous aurez corrigé l'instruction qui provoque le plantage, exécutez à nouveau le programme et notez quelles sont les valeurs correctements placées dans le tableau et celles qui ne le sont pas : que peut-on constater ?

Utilisez les outils ci-dessus pour corriger le programme afin d'obtenir les résultats corrects.