Travaux Dirigés Programmation C avancée

Informatique 1ère année.

—*Julien Allali* - `allali@enseirb-matmeca.fr` —

1 Gdb : premiers pas

Dans le programme `gdb`, la commande `help` permet de consulter la documentation des commandes internes à `gdb`.

▸**Exercice 1.** *Breakpoints*

Télécharger le fichier `gdb1.c.gz` *et décompresser le avec* `gunzip`*. Compiler ce fichier avec les options* `-g` *et* `-O0` *(moins la lettre "o" suivie de zéro).*

Lancer la commande `gdb ./a.out` *et vérifier que les symboles de debug sont bien trouvés dans votre binaire.*

Pour exécuter votre programme taper la commande `run`*.*

A l'aide de la commande `list`*, afficher le code du programme. A l'aide de la commande* `break` *positionner un point d'arrêt sur la fonction* `main` *: donner deux façons de faire cela.*

Exécuter à nouveau votre programme. Pour avancer d'une ligne, utiliser la commande `next`*. Si une ligne comporte un appel de fonction, la commande* `next` *ne vous fera pas renter dans cette fonction, la commande* `step` *si.*

▸**Exercice 2.** *Affichage Pour afficher la valeur d'une variable, on dispose de deux commandes :* `print` *ou bien* `display`*. Quelle est la différence entre ces deux commandes ?*

Exécuter votre programme en vous arrêtant avant chaque appel à la fonction `printf` *et en controlant : la valeur de* `i`*, la valeur et l'adresse de* `t[i]`*.*

▸**Exercice 3.** *Arrêt conditionnel Consulter la documentation de la commande* `cond`*. Utiliser cette commande pour suspendre l'exécution avant l'appel à* `printf` *lorsque la variable* `i` *vaut* 5*.*

▸**Exercice 4.** *Surveillance Consulter la documentation de la commande* `watch`*. Utiliser cette commande sur la variable* `i` *et observer ce qu'il se passe.*

2 Tables de hachage

La bibliothèque de table de hachage fournie comporte certains problèmes, nous allons les corriger ici à l'aide de gdb.

▸**Exercice 5.** *Ajout d'un nouveau test.*

Ajouter un nouvelle exemple qui procède à plusieurs insertions et recherches dans une table de hachage.

Ajouter la compilation de cet exemple dans votre script shell avec une liaison sur la bibliothèque statique.

Mettre en avant un bug au niveau de la recherche d'élément.

▸**Exercice 6.** *Compilation.*

Modifier votre script shell afin d'ajouter les options `-g` *et* `-O0` *lors de la compilation des fichiers objets.*

Recompiler l'ensemble du projet (bibliothèques et exemples).

▸**Exercice 7.** *gdb : se déplacer dans la pile*

Consulter la documentation des commandes `bt`*,* `up` *et* `down`*.*

Placer un point d'arrêt sur l'appel à la fonction `hash_init`*. Exécuter votre programme et utiliser les commandes ci-dessus une fois l'exécution stoppée.*

▸**Exercice 8.** *trouver le bug.*
Charger votre binaire d'exemple dans gdb, afficher le code source et positionner un point d'arrêt sur l'appel de fonction qui pose un problème.

Lancer votre programme puis tracer l'exécution afin de trouver l'origine du bug.

▸**Exercice 9.** *gdb training.*
Récupérer le fichier `list.c`. *Ce fichier comporte une implémentation partielle de listes doublement chainées avec plusieurs bugs.*

Compiler et debugger ce programme à l'aide de gdb.

3 GDB Contest !

Il est important de ne pas faire circuler les énoncés à l'extérieur de votre groupe dans l'intérêt de tous !

Vous allez former deux équipes qui vont s'affronter dans un combat de gdb.

Chaque "round" consiste à être le plus rapide pour trouver un correctif à un programme.

Au début de chaque "round", les membre d'une équipe tirent une carte au hasard dans le jeu de carte fourni. Celui qui a la plus grand carte (l'as vaut 1) est désigné opérateur : c'est lui et lui seul qui pourra entrer des commandes dans la session gdb. Les autres membres ont un rang donné par l'ordre croissant des cartes.

Un round se divise en set. La durée d'un set est propre à chaque problème et donné ci-dessous. Dans un set, la commande run ne peut être utilisée qu'une seule fois. Les autres commandes gdb n'ont pas cette limite.

Une fois les sessions gdb prête pour chaque équipe l'arbitre (l'enseignant) va signaler le départ d'un set. Les équipes commencent alors à jouer : chaque membre donne une commande à entrer en suivant leur ordre. Il est autorisé de discuter entre les membres mais il est interdit de proposer des commandes gdb explicitement (on a le droit de dire : "il faudrait connaître la valeur de i", on ne peut pas dire : "fait un print i"). Lorsque l'arbitre signale la fin du set, chaque équipe doit quitter gdb et le relancer (chaque set commence dans une session vierge).

Entre deux set, les équipes disposent d'un inter-set pendant lequel elles peuvent modifier le code source. Attention, à la fin de l'inter-set, il n'est plus possible de modifier et/ou de compiler le source. Il peut être judicieux de faire une copie du binaire original au cas où.

A tout moment une équipe peut mettre fin au round en criant "solution". L'équipe propose alors une solution, celle-ci est implémentée et testée. Si l'équipe a vu juste, alors elle marque des points, sinon c'est l'équipe adverse qui marque les points. Le nombre de point initial est donné par exercice, celui-ci est décrémenté par 2 au début de chaque set.

Entre deux rounds, les équipes disposent de 3 minutes pour debrifer et s'organiser pour le prochain round.

Toute faute observée par l'arbitre est sanctionné par -2 : commandes entrées après la fin d'un set, non respect de l'ordre des joueurs...

Votre chargé de TD vous donnera l'archive des épreuves une fois que les équipes seront prêtes à se battre.

▸**Exercice 10.** *Round 0 : Dho*
La durée d'un set est de 45 secondes.
La durée de l'inter-set est de 45 secondes.
Nombre initial de points : 6

** inter-round : 3 minutes **

▸**Exercice 11.** *Round 1 : Xtrem*
La durée d'un set est de 60 secondes.
La durée de l'inter-set est de 80 secondes.
Nombre initial de points : 12

** inter-round : 3 minutes **

▸**Exercice 12.** *Round 2 : Space*
La durée d'un set est de 60 secondes.
La durée de l'inter-set est de 80 secondes.
Nombre initial de points : 12

** inter-round : 3 minutes **

▸**Exercice 13.** *Round 3 : Cafard*
La durée d'un set est de 80 secondes.
La durée de l'inter-set est de 80 secondes.
Nombre initial de points : 14

** inter-round : 3 minutes **

▸**Exercice 14.** *Round 4 : Geek*
La durée d'un set est de 120 secondes.
La durée de l'inter-set est de 90 secondes.
Nombre initial de points : 18

** inter-round : 3 minutes **

▸**Exercice 15.** *Round 5 : Hack*
La durée d'un set est de 150 secondes.
La durée de l'inter-set est de 120 secondes.
Nombre initial de points : 20