Mettre au point votre code avec gdb ou ddd

gdb *file* ou **ddd** *file*	appel de gdb (ou de l'interface graphique ddd) sur l'exécutable *file*.
quit	Pour quitter le metteur au point.
help *cde*	Obtention d'aide sur la commande *cde.*
br *etiq* ou **br** *fichier*:*ligne*	Pose de point d'arrêt à l'entrée de *etiq* (ou a la ligne *ligne* du fichier *fichier*) de votre programme. Accessible via un double-clic dans la marge dans ddd.
info br	Liste des points d'arrêt courants.
delete *i*	Suppression du point d'arrêt numéro *i*.
run	Lancement de l'exécution du programme chargé depuis le début jusqu'au premier point d'arrêt s'il y en a de posé.
si	Exécution du programme pas à pas (instruction par instruction).
ni	Exécution du programme pas à pas mais en sautant les appels de fonctions.
cont	Poursuite de l'exécution jusqu'au prochain point d'arrêt.
RET	Retour chariot : exécute à nouveau la commande précédente.
CTRL-p	Prompte la commande précédente.
print $eax	Visualisation de registres particuliers, par exemple *eax.*
i reg	Visualisation de tous les registres. Accessible via Menu Status -> Registers dans ddd.
frame	Visualisation du bloc de pile associé à la fonction courante.
frame *i*	Visualisation du bloc de pile n° i (ième niveau d'appel) ou à l'adresse i.
bt	Liste des blocs empilés.
x/nfu *addr*	Visualisation de la mémoire : • *n* nombre d'unités à visualiser • *f* format d'affichage prend ses valeurs dans : **x** hexadécimal/ **d** décimal signé/ **u** décimal non signé / **t** binaire/ **i** instruction/ **s** chaîne terminée par 0 • *u* taille d'une unité qui peut être : **b** octet/ **h** demi-mot / **w** mot / **g** mot de 64 bits • *addr* est l'adresse, donnée sous forme hexadécimale ou symbolique, du début de la zone à visualiser. *Exemples :* *x/15i main # affiche 15 instructions à partir de l'adresse main.* *x/10i $eip # affiche 10 instructions à partir de l'instruction courante* *x/4xb &str # affiche les 4 premiers octets de str*

display	Surveillance de la valeur d'une variable, d'un registre ou du contenu d'une zone-mémoire à chaque pas d'exécution. Sans paramètre : liste des points de surveillance courants. Paramètre(s) : • *[/nfu] x* # valeur de la variable x. # Exemples : *display n* *# display /4xb &variable* 4 premiers octets de variable • *$reg* # contenu du registre x. # Exemples : *display $eax, display /x $eax* • /nfu *$reg* /* contenu de la zone mémoire pointée par le registre $reg. *Exemples :* o *display /4wx $ebx # valeur hexadécimale de 4 doubles mots à partir de l'adresse contenue dans EBX.* o *display /s $ebx # chaîne de caractères pointée par EBX. */*
set {type}addr = value	Modification d'un emplacement mémoire quelconque : • *type* le type de la variable. Le champ type prend ses valeurs dans *{short, int, char}* (en fait tout type de base du langage C). • *addr* son adresse. • *value* la valeur à écrire.
set $reg = value	Modification du contenu d'un registre : • *reg* est le nom du registre que l'on veut modifier • *value* est la valeur décimale, octale ou hexadécimale que l'on veut donner au registre. *Exemples :* *set $eax = 0xf3* *affecte la valeur hexadécimale 0xf3 au registre eax* *set $eax = ($eax &0xffff0000) | 0xfedc* *affecte la valeur hexadécimale 0xfedc au mot de poids faible de eax sans modifier la valeur du mot de poids fort.*