Fiche annexe sur le debug d'erreur mémoire

Une erreur mémoire consiste à accéder à une zone mémoire non prévue initialement. Une fuite mémoire consiste à laisser des espacements mémoires alloués sans les libérer.

Le noyau peut détecter certains type de dépassement de segments mémoires et interrompt alors le programme fautif en indiquant: *SegFault*, ou *Erreur de Segmentation*.

Note: Si le programme n'indique pas de SegFault, cela n'implique pas forcément qu'il n'y a pas d'erreur mémoire. Seul un programme externe de validation peut l'indiquer.

Les erreurs mémoires sont difficiles à debuger sans l'aide d'un programme externe car l'origine de l'erreur peut passer inaperçu et le comportement du programme peut être altéré.
Il existe pour cela deux programmes permettant de débuger les erreurs mémoires en indiquant l'origine de l'erreur: **gdb** et **valgrind**.

GDB

En cas d'erreur de segmentation, **gdb** permet d'indiquer la ligne à l'origine du problème et la trace d'execution associée.

Principe.

Soit un fichier exécutable du nom de [EXEC]. Le fichier exécutable doit impérativement avoir été compilé avec les options de debug (-g).

Lancer gdb (depuis le même répertoire que l'exécutable) avec
```
$ gdb ./[EXEC]
```

Démarrez ensuite l'exécutable avec
```
(gdb) run
```

La ligne à l'origine de l'erreur mémoire s'affiche alors (voir exemple en bas d'annexe).

Si l'executable attend des **arguments** [ARGS] de la ligne de commande, la syntaxe est la suivante
```
(gdb) run [ARGS]
```

L'affichage de la **trace d'exécution** s'obtient avec la commande
```
(gdb) bt
```

Pour **quitter gdb**:
```
(gdb) quit
```

Valgrind

Valgrind est un programme permettant de vérifier si des zones mémoires sont mal utilisés ou si il y a des fuites mémoires. Il détecte d'avantages d'erreurs que **gdb** et indiquent tous les comportements étranges, même si ceux-ci n'aboutissent pas à d'erreur de segmentation.
Avant de rendre un programme, il est recommandé de vérifier avec **Valgrind** si celui-ci ne détecte pas de comportement suspects.

Lancement de valgrind en ligne de commande (depuis le répertoire de l'executable):
`$ valgrind ./[EXEC]`
ou
`$ valgrind ./[EXEC] [ARGS]` *(dans le cas où il y a des arguments).*

Exemples de sorties:

Soit le programme incorrecte suivant:
```
int main()
{
    int *a;
    *a=8;

    return 0;
}
```

Cas de gdb:

```
(gdb) run
Starting program: [EXEC]

Program received signal SIGSEGV, Segmentation fault.
0x0000000000401af8 in main ()
    at [FICHIER]:LIGNE
LIGNE     *a=8;
```

Cas de valgrind:

```
$ valgrind ./[EXEC]

[...]
==5438== Invalid write of size 4 (indique écriture non permise à la ligne 41 du fichier main.c)
==5438==    at 0x401AF8: main (main.c:41)
==5438==  Address 0x0 is not stack'd, malloc'd or (recently) free'd
[...]
==5438== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 2 from 2)
(La dernière ligne résume le nombre d'erreurs: il faut toujours atteindre 0 erreurs au moment de rendre un programme)
```