I2010 : langage C (TP3 - suite)

Exercices de programmation avec pointeurs

2. Quiz moodle

Avant de résoudre les exercices suivants de cette fiche, faites le petit test CodeRunner « **TP3 Les pointeurs - Quiz** » sur Moodle.

3. Allocation dynamique de tableaux à une dimension

a) Écrivez un programme qui :

- lit sur *stdin* un entier `n` qui représente le nombre de données à encoder
- alloue dynamiquement un tableau de n entiers
- lit sur *stdin* n entiers qui sont enregistrés dans le tableau ; ces entiers peuvent être soit positifs, nuls ou négatifs
- après avoir lu les n données, le programme crée, remplit et affiche deux tableaux :
  - le premier contient la liste des entiers >=0 ;
  - le second la liste des entiers <0

Vous devez allouer dynamiquement les 3 tableaux de sorte que tous les éléments soient assignés (i.e. taille logique = taille physique) et libérer la place qu'ils occupent après leur utilisation.

Exemple d'exécution :

```
Entrez le nombre de donnees: 5
Entrez les donnees:
5 -2 56 12 -3
Résultats:
        tableau de valeurs positives: 5  56  12
        tableau de valeurs négatives: -2  -3
```

b) Après avoir traité les entrées et affiché le résultat, le programme redemandera un nouveau jeu de données à traiter tant que `n>0` .

4. Arithmétique des pointeurs

Pour rappel, lorsque vous avez un pointeur dans un tableau, vous pouvez passer à l'élément suivant en incrémentant simplement ce pointeur (via l'opérateur ++).

Passez d'une version indicée à une version pointeurs du programme 3 : modifiez-le afin de ne plus utiliser l'opérateur d'indexation **[ ]**.

Exercices d'observation et debugging

5. Utilisation du debugger *gdb*[1]

Prenez sur Moodle les programmes sources suivant :

1. *to_debug_stack.c*
2. *to_debug_stack_smashing_1.c*
3. *to_debug_segmentation_fault_1.c*
4. *to_debug_segmentation_fault_2.c*
5. *to_debug_stack_smashing_2.c*
6. *to_debug_doublette.c*

Certains de ces programmes comportent des fautes de style et/ou des erreurs de bonne gestion de la mémoire.

Compilez et exécutez-les dans cet ordre. Pour chaque programme, trouvez quel est le problème, et si c'est pertinent, comment le corriger. Mais vous ne devez **pas corriger ces programmes** !

Pour les 2 programmes avec *segmentation fault*, utilisez **gdb**.

[1] Pour les possesseurs de Mac, utilisez le débugger *lldb*.