Travaux Pratiques d'algorithmique n°6

Cours d'Informatique de Deuxième Année

—DEUG Sciences S3-MI—

Manipulation de pointeurs

Dans cette séance, nous abordons la gestion des pointeurs sous divers aspects :

- quelques rappels de syntaxe ;
- un peu d'expérimentation ;
- l'allocation de la mémoire.

▸ **Exercice 1.** *On considère le programme suivant :*

```
#include <stdio.h>
int main(void)
{
  int *ad1, *ad2, *ad3;
  int n = 10, p = 20;
  ad1 = &n;
  ad2 = &p;
  *ad1 = 3;
  *ad2 = *ad1 + *ad2 + p;
  printf("n = %d, p =  %d\n",n,p);
  printf("*ad1 = %d, *ad2 =  %d\n",*ad1,*ad2);
  printf("adresse de n=%p\n",&n);
  printf("valeur de ad1=%p\n",ad1);
  printf("adresse de ad1=%p\n",&ad1);
  *ad3 = *ad2;
  printf("*ad2 = %d, *ad3 =  %d\n",*ad2,*ad3);
  return 0;
}
```

Ce programme est syntaxiquement correct mais ne marche pas à l'exécution. Pourquoi ?

▸ **Exercice 2.** ***(Allocation dynamique de tableaux)***
Le but de cet exercice est de saisir un tableau de réels (alloué dynamiquement) puis d'effectuer des opérations sur ce tableau.

1. *Écrire une fonction permettant de saisir une suite d'au plus `MAX` nombre réels positifs. Les nombres ainsi saisis seront rangés dans un tableau alloué dynamiquement. La taille effective du tableau sera calculée par la fonction. La fonction retournera l'adresse du premier élément du tableau si l'allocation s'est bien passée et `NULL` sinon.*

2. *Réécrire la fonction précédente de telle sorte que la valeur de retour soit 1 si tout s'est bien passé et 0 en cas de problème (l'allocation a échoué ou bien la taille effective du tableau est supérieure à `MAX`).*

3. *Écrire une fonction qui effectue une permutation circulaire à gauche des éléments d'un tableau de réels. Par exemple, le tableau contenant $(3.2, 1.3, 5.6, 2.8)$ doit être transformé en $(1.3, 5.6, 2.8, 3.2)$.*
   Écrire une autre fonction qui permet d'afficher un tableau de réels et, bien évidemment, écrire aussi la fonction `main`.

▸ **Exercice 3.** *Dans l'exercice précédent, le nombre de valeurs à saisir doit obligatoirement être inférieur ou égal à la valeur prédéfinie `MAX`. Dans la pratique, aucune valeur de `MAX` ne peut convenir. On peut résoudre ce problème en considérant des allocations (et réallocations) "par blocs". On alloue d'abord un tableau de taille `MAX`, qu'on remplit. Si cela ne suffit pas, on réalloue le tableau pour qu'il ait une taille $2\times$`MAX` et on continue à le remplir. Si cela ne suffit pas, on le réalloue pour qu'il ait une taille $3\times$`MAX` et on continue à le remplir, etc...*

- *Écrire une autre fonction permettant de lire une suite quelconque de nombres réels positifs en utilisant cette méthode.*

▸ **Exercice 4.** ***(Triangle de Pascal)***

- *Construire à la main le triangle de Pascal de dimension 4 à l'aide des relations suivantes :*

$$\begin{cases} M_{1,1} &= 1 \ , \\ M_{i,j} &= M_{i-1,j-1} + M_{i-1,j} \quad \text{pour } 0 < j \leq i \ , \\ &= 0 \quad \text{sinon.} \end{cases}$$

- *Écrire les fonctions `AllocMat`, `TrianglePascal`, `AffichageTriangle` et `FreeMat` pour respectivement allouer une matrice triangulaire, remplir le triangle, afficher le triangle et libérer l'espace mémoire précédemment alloué.*