

Langage C

Institut d'Optique 1A/S5

TD 7 - Allouer dynamiquement la mémoire

A retenir

Prérequis : tableaux statiques, création et manipulation de pointeurs.

Notions étudiées : allocation de mémoire avec `malloc()`, allocation dans les fonctions.

Bonnes pratiques de programmation : arguments d'entrée/de sortie des fonctions, libération de la mémoire lorsqu'un tableau n'est plus utile.

Diapos à lire :

- LangC-Allocation_dynamique

Exercice 1. Remplissage d'un tableau à la console

Dans la fonction `main()`, écrivez les lignes de codes qui permettent à l'utilisateur de choisir la taille d'un tableau d'entiers et d'allouer un tableau d'entiers de cette taille.

Faites ensuite une fonction permettant à l'utilisateur de saisir au clavier les valeurs de ce tableau. Cette fonction a déjà été réalisée lors du TD sur la modularité (`saisieTabInt`). Vous pouvez soit la reprogrammer soit utiliser la modularité interfichiers pour y faire appel directement.

Exercice 2. Carré des composantes d'un tableau

Écrivez une fonction calculant le carré des éléments d'un tableau d'entiers passé en argument d'entrée de la fonction, et les stockant dans un autre tableau. Ce second tableau sera alloué dynamiquement dans la fonction et devra pouvoir être récupéré dans le `main()` après appel à la fonction (méthode 1 du cours).

Faites une seconde fonction où le tableau sera alloué dans le `main()` (méthode 2 du cours).

Exercice 3. En-tête d'un fichier texte et allocation dynamique

On dispose d'un fichier écrit au format texte *exo_allocation.txt* (à télécharger sur eCampus). Ce fichier contient les données suivantes : la première ligne est le nombre de valeurs contenues dans le fichier *dim*, les *dim* lignes suivantes contiennent des nombres entiers. Adaptez la fonction de lecture dans un fichier au format texte vue lors du TD sur la lecture/écriture dans des fichiers au format texte (*lecture_fichier*) permettant de récupérer la valeur de *dim*, d'allouer dynamiquement la mémoire nécessaire pour stocker *dim* nombres entiers et d'y récupérer les *dim* valeurs contenues dans le fichier. Vous afficherez les valeurs récupérées dans le fichier sur la console pour vérifier votre fonction.

Pour vous aider, le prototype de la fonction pourra être :

```
int* lecture_fichier_entete(char nomfichier[], int* dim)
```