Gestion de la mémoire

Allocation dynamique

- On a souvent besoin d'utiliser de la mémoire qu'on ne peut pas réserver à l'avance parce qu'on ne connaît pas la taille à la compilation
  - Données lues à partir d'un fichier, du clavier, etc
- Nécessité de pouvoir réserver dynamiquement de la mémoire dans le tas à l'exécution
- Le tas contient de la mémoire
  - Réservée et initialisée à la compilation
  - Réservée à la compilation mais non initialisée
  - Non réservée (tout le reste, jusqu'à la pile)

La fonction malloc (1)

- Permet de réserver un certain nombre d'octets dans le tas
  - on passe en paramètre le nombre d'octets souhaités (utiliser l'opérateur sizeof pour calculer la taille à partir du nombre d'objets)
  - La fonction renvoie un pointeur sur le début de la zone allouée, ou NULL si plus de mémoire

```
int tabnbr;                 /* Nombre d'éléments dans le tableau    */
int *tabptr;                /* Pointeur sur le tableau des éléments */
printf ("Entrez le nombre d'éléments :");
scanf ("% d",&tabnbr);      /* Lecture du nombre d'éléments         */
tabptr = malloc (tabnbr * sizeof (int));/* Allocation               */
for (i= 0;i<tabnbr;i++)     /* Chargement des éléments              */
scanf ("% d",&tabptr[i]);
```


La fonction malloc (2)

- Il est important de tester la valeur de retour de la fonction malloc pour détecter au plus tôt les problèmes
  - Test avec traitement d'erreur dans le cadre du déroulement du programme

```
if ((tabptr = m alloc (tabnbr * sizeof (int))== NU LL){
        ... /* Gestion de l'erreur */
}
```

  - Assertion avec la macro assert

```
tabptr = m alloc (tabnbr * sizeof (int));
assert (tabptr != NU LL); /* Termine le programme si pas vrai*/
```

La fonction free

- Permet de libérer une zone mémoire allouée avec malloc
  - Ne pas utiliser avec les zones non allouées par malloc
  - Ne pas appeler free plusieurs fois sur la même zone

```
free(tabptr);
```

- A fin d'éviter les fuites mémoire, il est préférable d'appeler free dès que la zone mémoire considérée n'est plus utilisée.

Exercice

- Soit le bout de code suivant :

```
…
int main(){
    int *tab1;
    int *tab2;
    int *tab3;
    printf("Donner le nombre d'éléments du premier tableau :");
    …
    printf("Donner le nombre d'éléments du second tableau :");
    …
}
```

complétez ce code de manière à ce que tab1 et tab2 soient deux tableaux d'entiers, les éléments seront saisis par l'utilisateur
Le tableau tab3 doit lui contenir l'ensemble d'éléments contenus dans tab1 ou dans tab2.