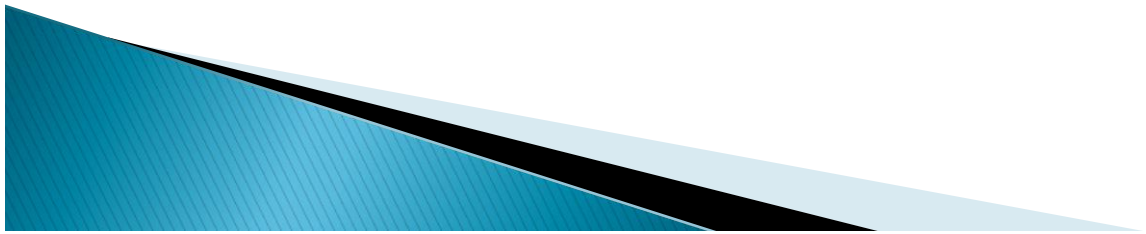
Gestion dynamique de la mémoire

- L'attribution d'une adresse valide à un pointeur par allocation d'un nouvel emplacement mémoire

- Fonctions de la bibliothèque **stdlib**

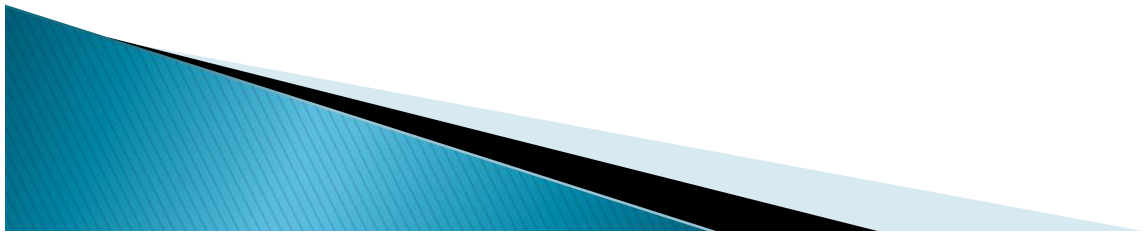
- **void * malloc (size_t nb_octets) ;**
- **void * calloc (size_t nb, size_t taille) ;**
- **void * realloc (void * pointeur, size_t nb_octets) ;**
- **void free (void * pointeur) ;**



Malloc

- **void * malloc (size_t nb_octets) ;**
- **malloc (nb_octets)** renvoie un pointeur sur une zone de la mémoire de taille **nb_octets** ou **NULL** en cas d'échec

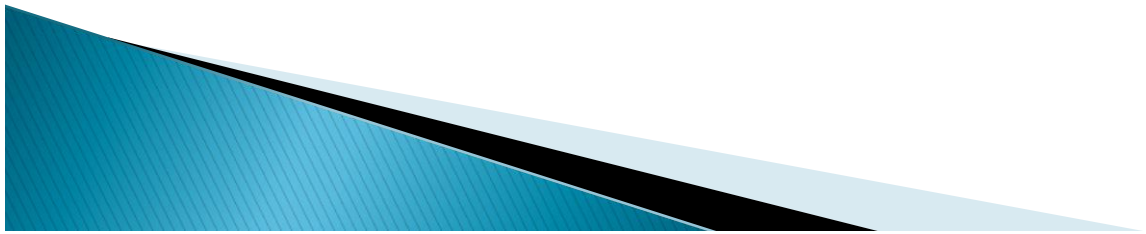
```
1 #include < stdio.h >
2 #include < stdlib.h >
3 int main( )
4 {
5       int * pt = NULL;
6       pt = malloc(sizeof(int)) ;
7       if(pt == NULL) printf("Echec allocation dans malloc \n") ;
8       return 0;
9 }
```



Calloc

- **void * calloc (size_t nb, size_t taille) ;**
- **calloc(nb, taille)** renvoie un pointeur sur une zone de la mémoire permettant de ranger **nb** objets de grandeur **taille** ou **NULL** en cas d'échec

```
1 #include < stdio.h >
2 #include < stdlib.h >
3 int main( )
4 {
5       int * pt = NULL;
6       pt = calloc(5, sizeof(int)) ;
7       if(pt == NULL) printf("Echec allocation dans calloc \n") ;
8       return 0;
9 }
```



Realloc

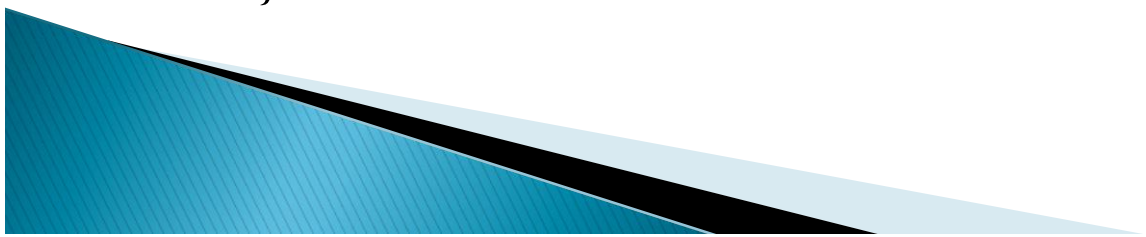
- **void * realloc (void * pointeur, size_t nb_octets) ;**
- **realloc(pointeur, nb_octets)** agrandit une zone précédemment allouée et repérée par **pointeur** à la taille **nb_octets**, ou, en cas d'échec, retourne le pointeur **NULL** et le bloc pointé par pointeur peut être détruit.

```
1 #include < stdio.h >
2 #include < stdlib.h >
3 int main( )
4 {
5       int * pt1 = NULL, * pt2 = NULL;
6       pt1 = calloc(5, sizeof(int)) ;
7       if(pt1 == NULL) printf("Echec allocation dans calloc \n") ;
8       else {
9                   pt2 = realloc(pt1, 10*sizeof(int)) ;
10                  if(pt2 == NULL) printf("Echec allocation dans realloc \n") ;}
11      return 0;
12}
```

Free

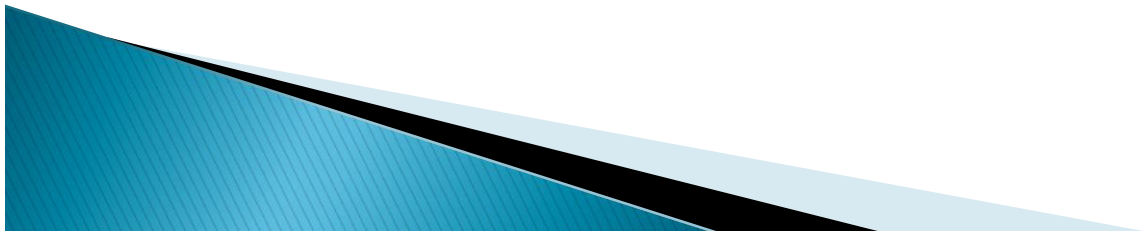
- **void free (void * pointeur) ;**
- **free(pointeur)** libère l'emplacement mémoire précédemment alloué et repéré par **pointeur**

```
1 #include < stdio.h >
2 #include < stdlib.h >
3 int main( )
4 {
5       int * pt1 = NULL;
6       pt1 = calloc(5, sizeof(int)) ;
7       if(pt1 == NULL) printf("Echec allocation dans calloc \n") ;
8       else  free(pt1);
9       return 0;
10 }
```



Les tableaux dynamiques

```
1 #include < stdio.h >
2 #include < stdlib.h >
3 int main( )
4 {
5       int n, *tab = NULL;
6       printf("Donner la taille du tableau \n") ;
7       scanf("%d", &n) ;
8       tab =calloc(n,sizeof(int)) ;
9       initialiser(tab,n) ;
10      afficher(tab,n) ;
11      return 0;
12 }
```



Tableaux multidimensionnels dynamiques

```
1 #include < stdio.h >
2 #include < stdlib.h >
3 int main( )
4 {
5       int n, m, **matrice = NULL;
6       printf("Donner le nombre de lignes \n") ;
7       scanf("%d", &n) ;
8       printf("Donner le nombre de colonnes \n") ;
9       scanf("%d", &m) ;
10      matrice =calloc(n,sizeof(int*)) ;
11      for(i = 0; i < n; i++) matrice[i] =calloc(m, sizeof(int)) ;
12      initialiser(matrice, n, m) ;
13      afficher(matrice, n, m) ;
14 }
```