Langage C

Allocation dynamique de mémoire

François Cuvelier

Laboratoire d'Analyse Géométrie et Applications
Institut Galilée
Université Paris XIII.

16 novembre 2012

Plan

Introduction

L'allocation dynamique de mémoire permet en cours d'éxécution d'un programme de réserver un espace mémoire dont la taille n'est pas nécessairement connue lors de la compilation.

Introduction

L'allocation dynamique de mémoire permet en cours d'éxécution d'un programme de réserver un espace mémoire dont la taille n'est pas nécessairement connue lors de la compilation.
L'allocation dynamique est réalisée par les fonctions malloc() et calloc ().

Plan

Fonction malloc()

Prototype

```
void * malloc ( size_t t );
```

Description

permet de réserver (si possible) un bloc de taille t (en bytes) et renvoie

- un pointeur vers l'adresse du bloc alloué s'il y a suffisament de mémoire disponible
- la valeur NULL en cas d'erreur.

Fonction malloc() : Exemple 1

Code

```
1 | int *pi,n;
2 | printf("n=");scanf("%d",&n);
3 | pi= (int *)malloc(n*sizeof(int));
```

Description

La fonction malloc alloue (si possible) un bloc de n int et retourne l'adresse (non typée, void *) du bloc.
On effectue alors une conversion de type : void * vers int *.
Attention, aucun test si la fonction retourne la valeur NULL.

Fonction malloc() : Exemple 2

Code

```
1 pi= (int *)malloc(n*sizeof(int));
2 if ( pi == NULL ){
3   fprintf(stderr,
4     "Allocation impossible : fichier %s, ligne %d\n",
5     __FILE__,__LINE__-2);
6   exit(EXIT_FAILURE);
7 }
```

Description

En cas d'erreur d'allocation, le programme affiche un message d'erreur et se termine !

On utilise ici des constantes du préprocesseur :

- __FILE__ : affiche le nom du fichier source.
- __LINE__ : affiche le numéro de la ligne atteinte

Plan

Fonction calloc ()

Prototype

```
void * calloc(size_t n, size_t t);
```

Description

permet de réserver (si possible) n blocs de taille t (en bytes) et renvoie

- un pointeur vers l'adresse du bloc alloué s'il y a suffisament de mémoire disponible
- la valeur NULL en cas d'erreur.
- les blocs alloués sont tous mis à 0 (contrairement à la fonction malloc() où la mémoire est non initialisée : donc aléatoire)

Fonction calloc () : Exemple 1

Code

```
1 | int *pi,n;
2 | printf("n=");scanf("%d",&n);
3 | pi= (int *)calloc(n,sizeof(int));
```

Description

La fonction calloc alloue (si possible) un bloc de n int et retourne l'adresse (non typée, void *) du bloc.
On effectue alors une conversion de type : void * vers int *.
De plus, chaque bloc contient la valeur 0.
Attention, aucun test si la fonction retourne la valeur NULL.

Plan

Fonction free ()

Prototype

```
void  free(void *);
```

Description

permet de libérer de la mémoire préalablement allouée par les fonctions malloc() ou calloc ().
En paramètre, on passe l'adresse du bloc mémoire à désallouer.

Fonction free() - Exemple

Code

```c
#include <stdlib.h>
#include <stdio.h>

void main(){
   int *pi,n,i;
   printf("n=");scanf("%d",&n);
   pi= (int *)malloc(n*sizeof(int));
   if ( pi == NULL ){
      fprintf(stderr,
      "Allocation impossible : fichier %s, ligne %d\n",
      __FILE__,__LINE__-4);
      exit(EXIT_FAILURE);
   }
   for (i=0;i<n;i++) pi[i]=2*i;
   for (i=0;i<n;i++) printf("pi[%d]=%d\n",i,pi[i]);
   free(pi);
}
```