LEPL1503/LSINC1503 - Cours 2

O. Bonaventure, B. Legat, L. Metongnon

☐ Full Width Mode ☐ Present Mode

Table of Contents

Variables, pointeurs et doubles pointeurs

Visualisation

Arguments de main

- `int argc` : Le nombre d'arguments (incluant le nom de l'exécutable)

- `char **argv` : liste des arguments initialisée par le système d'exploitation au lancement du programme

```c
#include <stdio.h>
int main(int argc, char **argv) {
  for(int i=0; i<argc; i++) {
    printf("arg[%d]: %s\\n", i, argv[i]);
  }
}
```

```
Running : `/tmp/jl_fKGdVG/bin ab cd ef`
arg[0]: /tmp/jl_fKGdVG/bin\narg[1]: ab\narg[2]: cd\narg[3]: ef\n
```

Matrices en C

- Comment construire une matrice ?

```c
float A[2][2]={ {1,0}, {2,3} };
```

- Une telle matrice sera stockée sur le **stack** (variable locale) ou dans la zone données (variable globale)

Matrices sur le heap

Comment mettre une matrice 2x2 sur le heap ?

```c
float *matrice0 = (float *) malloc(4 * sizeof(float *));
matrice0[1][1] = 4;
```

```
/tmp/jl_sGyhqD/main.c:5:14: error: subscripted value is not an array, poi
nter, or vector
  matrice0[1][1] = 4;
  ~~~~~~~~~~~^~
1 error generated.
```

```c
float *matrice1=(float *) malloc(4 * sizeof(float));
matrice1[1][1] = 4;
```

```
/tmp/jl_Ychgu7/main.c:5:14: error: subscripted value is not an array, poi
nter, or vector
  matrice1[1][1] = 4;
  ~~~~~~~~~~~^~
1 error generated.
```

Première approche, un malloc par ligne

C (C17 + GNU extensions)

```
 1 #include <stdlib.h>
 2
 3 int main(int argc, char **argv) {
 4   float **matrice2 = (float **) m
 5   float *ligne0 = (float *) mallo
 6   float *ligne1 = (float *) mallo
 7   matrice2[0] = ligne0;
 8   matrice2[1] = ligne1;
 9   matrice2[1][1] = 4;
10 }
```

Edit Code & Get AI Help

line that just executed
next line to execute

< Prev | Next >

Step 1 of 7

Visualized with pythontutor.com

Stack | Heap

main

argc	int ?
argv	pointer to char* ?
matrice2	pointer to float* ?
ligne0	pointer to float ?
ligne1	pointer to float ?

Note: ? refers to an uninitialized value

C/C++ details: none [default view]

Deuxième approche, un malloc pour toutes les valeurs

C (C17 + GNU extensions)

```
1  #include <stdlib.h>
2
3  int main(int argc, char **argv) {
4    float **matrice3 = (float **) ma
5    float *valeurs = (float *) mallo
6    matrice3[0] = valeurs;
7    matrice3[1] = valeurs + 2;
8    matrice3[1][1] = 4;
9  }
```

Edit Code & Get AI Help

line that just executed
next line to execute

< Prev | Next >

Step 1 of 6

Visualized with pythontutor.com

Stack | Heap

main

argc	int ?
argv	pointer to char* ?
matrice3	pointer to float* ?
valeurs	pointer to float ?

Note: ? refers to an uninitialized value

C/C++ details: none [default view]

Initialisation d'une valeur

Comment initialiser la première valeur ?

```
float **matrice = (float **) malloc(2 * sizeof(float *));
float *ligne0 = (float *) malloc(2 * sizeof(float));
float *ligne1 = (float *) malloc(2 * sizeof(float));
matrice[0] = ligne0;
matrice[1] = ligne1;
matrice[0][0] = 1; // 1
*(matrice[0]) = 1; // 2
*(matrice) = 1;    // 3
```

wooclap https://app.wooclap.com/JAPRXX

Comment faire matrice[0][1] = 0 ?

```
float **matrice = (float **) malloc(2 * sizeof(float *));
float *ligne0 = (float *) malloc(2 * sizeof(float));
float *ligne1 = (float *) malloc(2 * sizeof(float));
matrice[0] = ligne0;
matrice[1] = ligne1;
*(*matrice+1)=0;   // 1
*(matrice++)=0;    // 2
*(matrice[0]+1)=0; // 3
```

wooclap https://app.wooclap.com/JAPRXX

Comment faire matrice[1][0] = 2 ?

```
float **matrice = (float **) malloc(2 * sizeof(float *));
float *ligne0 = (float *) malloc(2 * sizeof(float));
float *ligne1 = (float *) malloc(2 * sizeof(float));
matrice[0] = ligne0;
matrice[1] = ligne1;
*(*(++matrice))=2; // 1
*(*(matrice+1))=2; // 2
*(*matrice+1)=2;   // 3
```

wooclap https://app.wooclap.com/JAPRXX

Libération de la mémoire

L'importance de l'ordre de désallocation

Si la matrice a été créée par

```
float **matrice = (float **) malloc(2 * sizeof(float *));
float *ligne0 = (float *) malloc(2 * sizeof(float));
float *ligne1 = (float *) malloc(2 * sizeof(float));
matrice[0] = ligne0;
matrice[1] = ligne1;
```

▶ **Comment libérer la mémoire avec `free` ?**

wooclap https://app.wooclap.com/JAPRXX

Toujours free dans le sens inverse de malloc

C (C17 + GNU extensions)

```
1  #include <stdlib.h>
2
3  int main(int argc, char **argv) {
4    float **matrice = (float **) ma
5    matrice[0] = (float *) malloc(2
6    matrice[1] = (float *) malloc(2
7    free(matrice);
8    free(matrice[0]);
9    free(matrice[1]);
10 }
```

Edit Code & Get AI Help

line that just executed
next line to execute

< Prev Next >

Done running (6 steps)

ERROR: Invalid read of size 8
(Stopped running after the first err r. Please fix your code.)

see UNSUPPORTED FEATURES:undefined

Stack Heap

main

argc	int 1
argv	pointer to char* 💀
matrice	pointer to float* 💀

Note: 💀 means a pointer points to memory that is ei
misaligned with data boundaries. 💀 locations are *apı*
not match the pointer's real address. Select 'byte-leve
to see more details:

C/C++ details: none [default view]

Un vecteur "Java" en C

Première implémentation

```
struct vector_t {
  int size;
  float *v;
};
```

```
struct vector_t * init(int size, float val) {
```

```
float get(struct vector_t *t, int i) {
```

```
void set(struct vector_t *t, int i, float val) {
```

```
void destroy(struct vector_t *vect) {
```

wooclap https://app.wooclap.com/JAPRXX

Une autre implémentation ...

Toutes les fonctions doivent vérifier leurs arguments et retourner

- -1 en cas d'erreur
- 0 en cas de succès

```c
struct vector_t {
  int length; // nombre d'élements
  float *tab; // tableau avec les réels
};
```

Initialisation


```c
struct vector_t {
  int length; // nombre d'élements
  float *tab; // tableau avec les réels
};
```


Quels arguments ?

```c
/*
 * @pre
 * alloue la mémoire pour un vecteur de size éléments
 * initialisés à la valeur val
 * @return -1 si erreur, 0 sinon
 */

int init(int size, float val,
  struct vector_t  v,     // 1
  struct vector_t ** v,   // 2
  struct vector_t * v,    // 3
)
```

wooclap https://app.wooclap.com/JAPRXX

Allocation de la mémoire


```c
struct vector_t {
  int length; // nombre d'élements
  float *tab; // tableau avec les réels
};
```

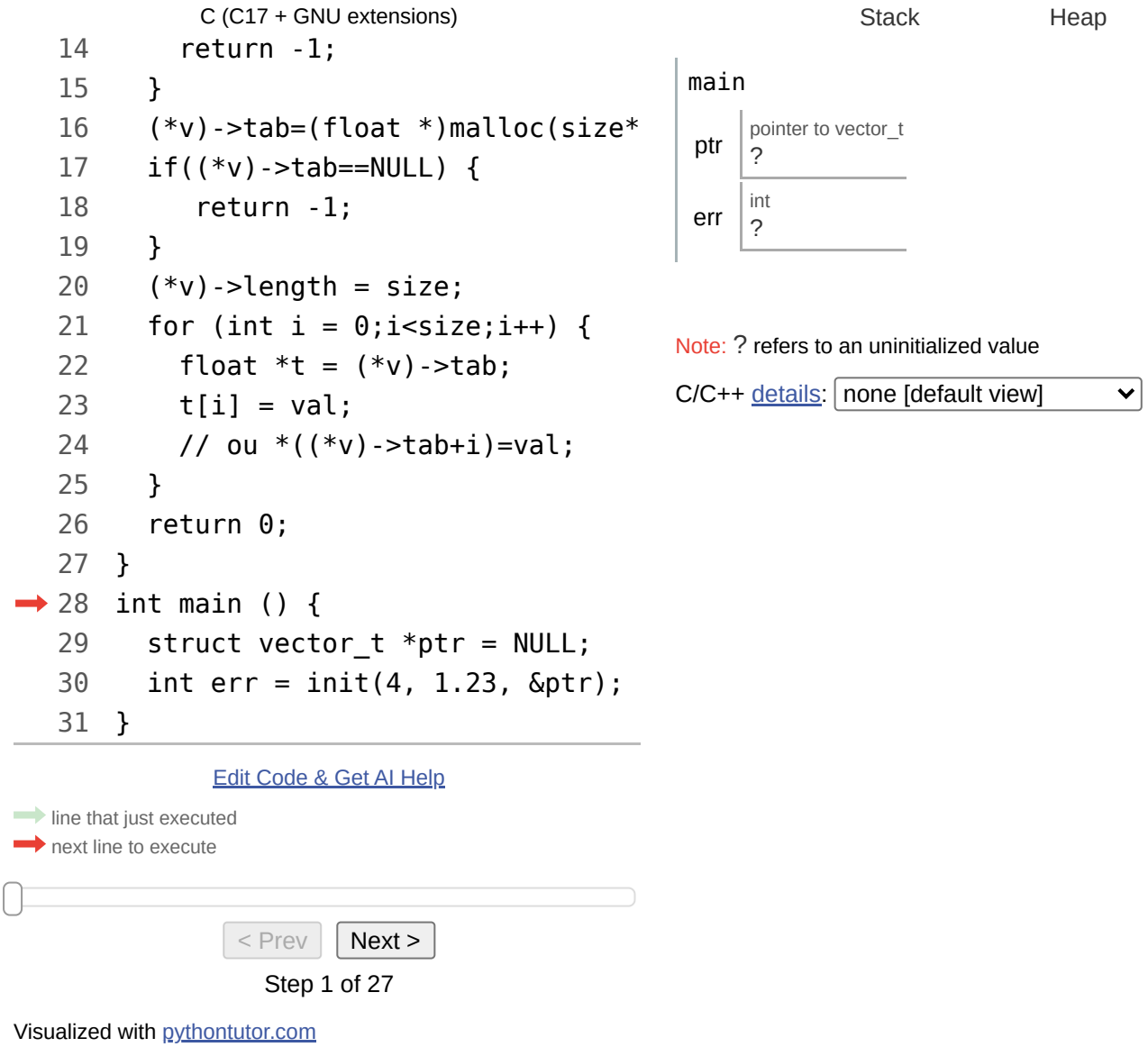
```c
int init(int size, float val, struct vector_t ** v) {
  if ((size<0) || v==NULL)
    return -1;
  *v=(struct vector_t *)
     malloc(sizeof(struct vector_t));
  if (*v == NULL) {
    return -1;
  }
  (*v)->tab=(float *)malloc(size*sizeof(float));
  if ((*v)->tab == NULL) {
     return -1;
  }
  (*v)->length = size;
  for (int i = 0; i<size; i++) {
```

▶ **Comment assigner la valeur `val` au `i`ième élément ?**

```c
  }
  return 0;
}
```

wooclap https://app.wooclap.com/JAPRXX

Visualization de init



Récupération d'un élément

```c
struct vector_t {
  int length; // nombre d'élements
  float *tab; // tableau avec les réels
};
```

```
/*
 * @pre
 * @post retourne le ième élément du tableau dans val
 *       -1 en cas d'erreur, 0 sinon
 */

int get(struct vector_t *v, int i,
  float val   // 1
  float **val // 2
  float *val  // 3
) {
```

▶ **Comment implémenter `get` ?**

wooclap https://app.wooclap.com/JAPRXX

Modification d'un élément

```
/*
 * @pre
 * @post place val comme ième élément
 *             du tableau
 *       -1 en cas d'erreur, 0 sinon
 */

int set(struct vector_t *v, int i,
struct vector_t  v     // 1
struct vector_t ** v   // 2
struct vector_t * v    // 3
) {
```

▶ **Comment implémenter `set` ?**

wooclap https://app.wooclap.com/JAPRXX

Libération de la mémoire

Comment libérer la mémoire quand le vecteur est devenu inutile ?

```
/*
 * libère la mémoire utilisée pour le vecteur
 * -1 en cas d'erreur, 0 sinon
 */
```

▶

```
int destroy(struct vector_t *v) {
```

wooclap https://app.wooclap.com/JAPRXX

Pointeur de fonction

`mapreduce` donne un exemple de fonction d'ordre supérieur en C

C (C17 + GNU extensions)

```
 4 int square(int a) {
 5   return a * a;
 6 }
 7 int plus(int a, int b) {
 8   return a + b;
 9 }
10
11 int mapreduce(int (*f)(int), int
12   for (int i = 0; i < len; i++) {
13     init = op(init, f(vec[i]));
14   }
15   return init;
16 }
17
18 int main () {
19   int vec[] = {1, 2, 3, 4};
20   printf("%d\\n", mapreduce(squar
21 }
```

Edit Code & Get AI Help

line that just executed
next line to execute

< Prev | Next >

Step 1 of 43

Visualized with pythontutor.com

Print output (drag lower right corner to resize)

Stack | Heap

main

vec → array [0 1 2 3]: int ? | int ? | int ? | int ?

Note: ? refers to an uninitialized value

C/C++ details: none [default view]

Retourner un tableau

Heap ou stack ?

Laquelle de ces deux fonctions présente une façon correcte de retourner un tableau ?

```
int *stack() {
  int v[] = {1, 2, 3};
  return v;
}

int *heap() {
  int *v = (int *) malloc(3 * sizeof(int));
  v[0] = 1; v[1] = 2; v[2] = 3;
  return v;
}
```

wooclap https://app.wooclap.com/JAPRXX

Retourner un tableau : visualization

C (C17 + GNU extensions)

```
 4   int v[] = {1, 2, 3};
 5   return v;
 6 }
 7
 8 int *heap() {
 9   int *v = (int *) malloc(3 * siz
10   v[0] = 1; v[1] = 2; v[2] = 3;
11   return v;
12 }
13
14 int main () {
15   int *v1 = heap();
16   printf("%d %d %d\\n", v1[0], v1
17   free(v1); // don't forget!
18   int *v2 = stack();
19   printf("%d %d %d\\n", v2[0], v2
20   return 0;
21 }
```

Edit Code & Get AI Help

line that just executed
next line to execute

< Prev Next >

Done running (13 steps)

ERROR: Invalid read of size 4
(Stopped running after the first err r. Please fix your code.)

Print output (drag lower right corner to resize)

```
1 2 3\n
```

Stack Heap

main

v1 — pointer to int 💀

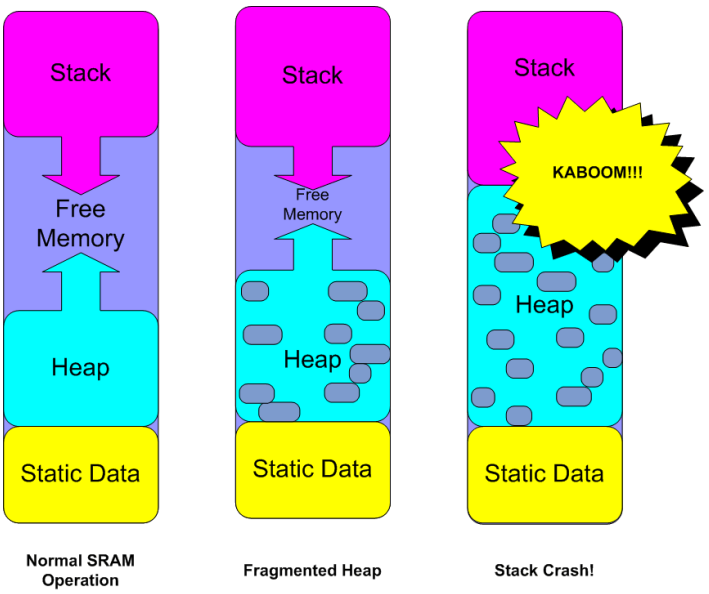
v2 — pointer to int NULL (0x0)

Note: 💀 means a pointer points to memory that is ei misaligned with data boundaries. 💀 locations are *ap* not match the pointer's real address. Select 'byte-leve to see more details:

C/C++ details: none [default view]

Structure différente du stack et heap

- Stack : Pile de mémoire locale des functions, structure connue à la **compilation**
- Heap : Alloué et désalloué avec des tailles connues à l'**exécution**. L'OS fait de son mieux pour organiser cette mémoire et réassigner les fragments formés suite aux désallocations



Source

Projet

- Si vous avez une question sur le projet, ouvrez une issue ici