sizeof(char)=1

sizeof(int)=4

sizeof(float)=4

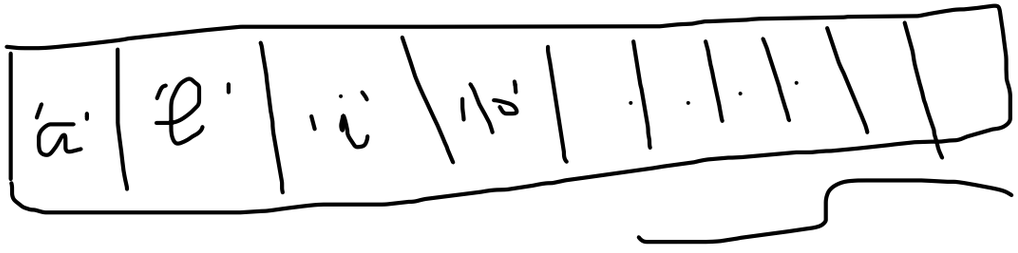
sizeof(double)=8

```
main()
{
 int a;
float x;
printf("la taille de a est %d\n",sizeof(a));
printf("la taille de x est %d\n",sizeof(x))
}
```

Exam

```
la taille de a est 4
la taille de x est 4
```

char nom[10]="ali"



→ 4

6 octets

int T[100];

100 x 4 = 400 octets

N = 10 → 40 octets → Perte 360 octets

int T[100] ⟶ on a réservé 4 kio

(1 ko = 1000 octets)

N = 5 ⟶ 2 octet

⟶ puis 9 bits = octets

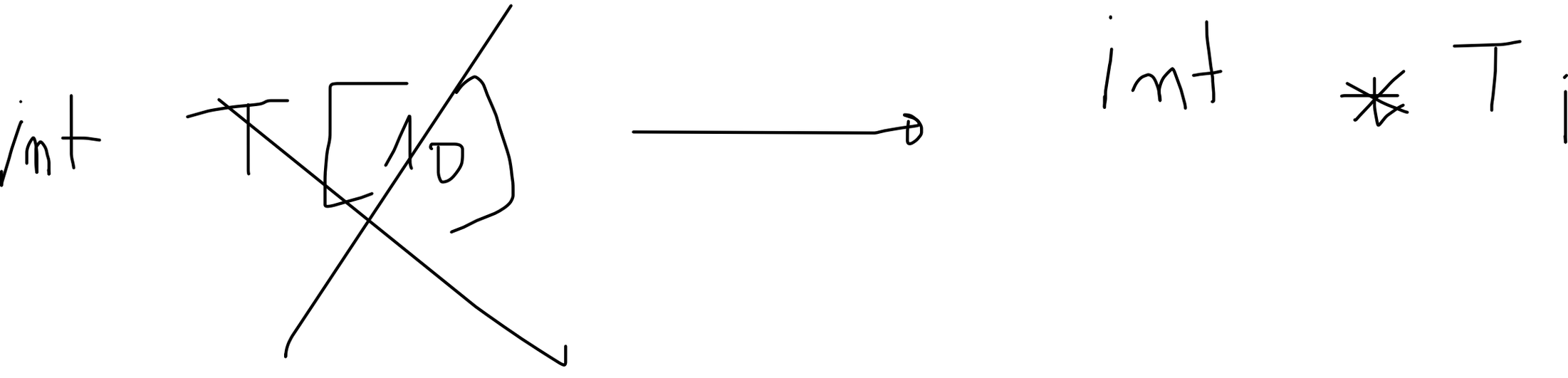
La réservation statique: On fixe la taille qu'on va utiliser au moment de compilation:

Réservation dynamique:se fait au moment

Réservation statique	Réservation dynamique
1. Fin de l'espace	1. Pas de fin
2. Compilation	2. Exécution
3. Dans la zone statique RAM	3. Zone : Tas

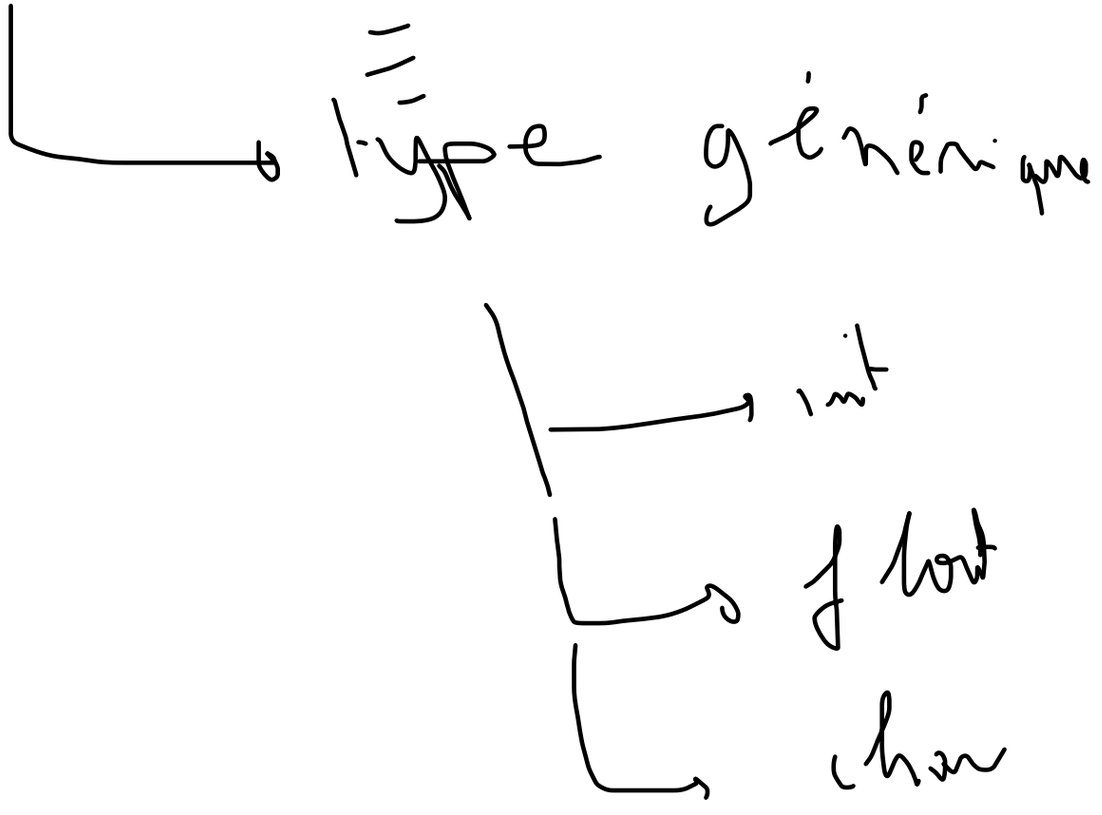
La structure logique de la RAM

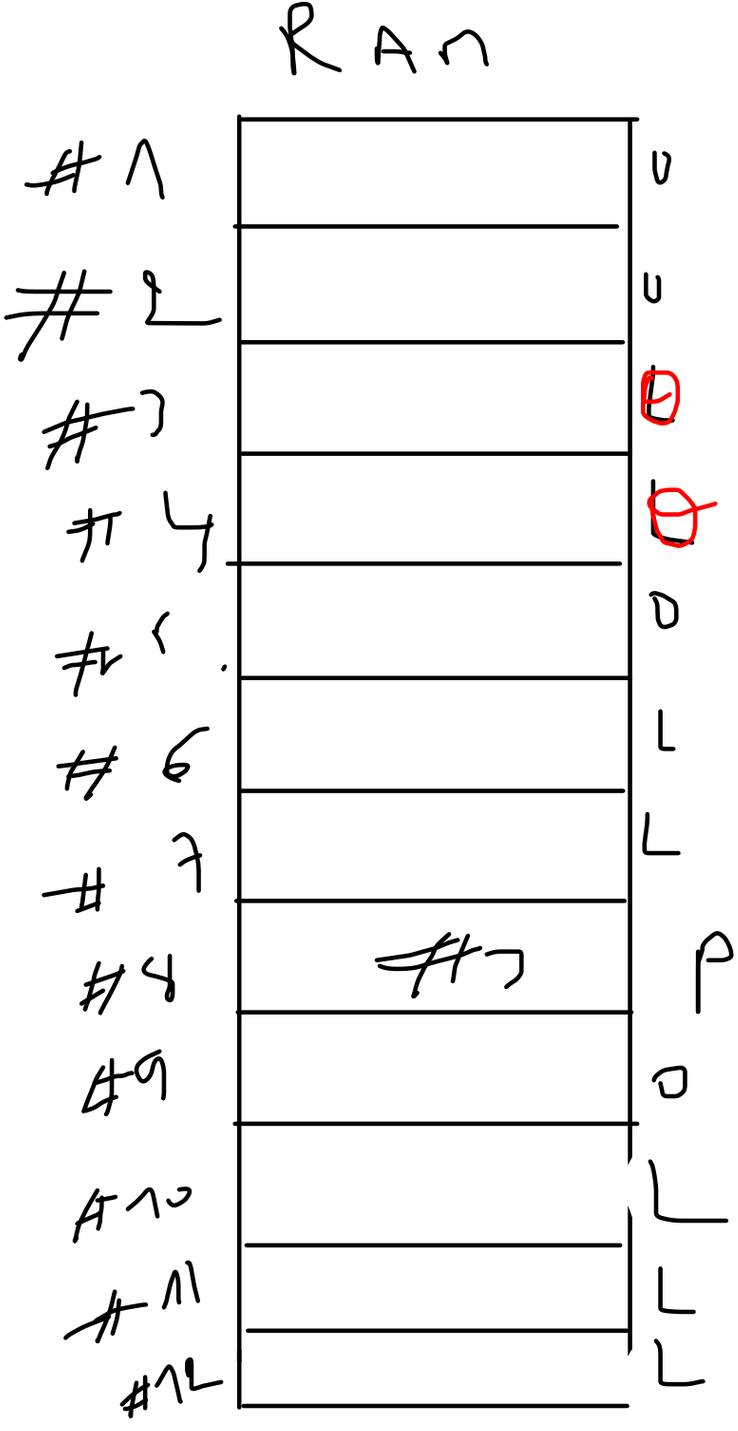
La réservation dynamique se fait à l'aide d'un pointeur



1

void * malloc(int size)

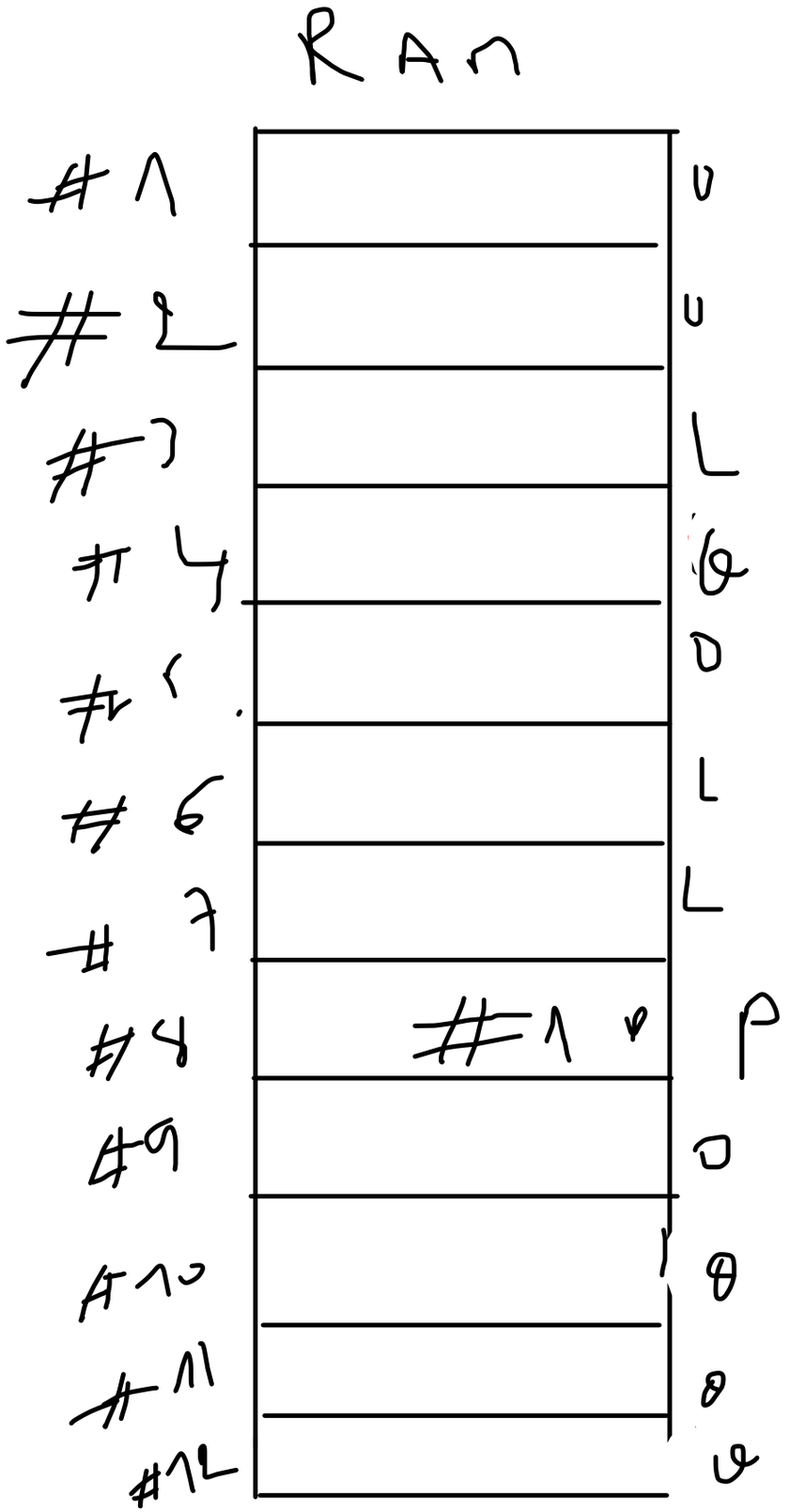




int *p;

p = malloc (8)

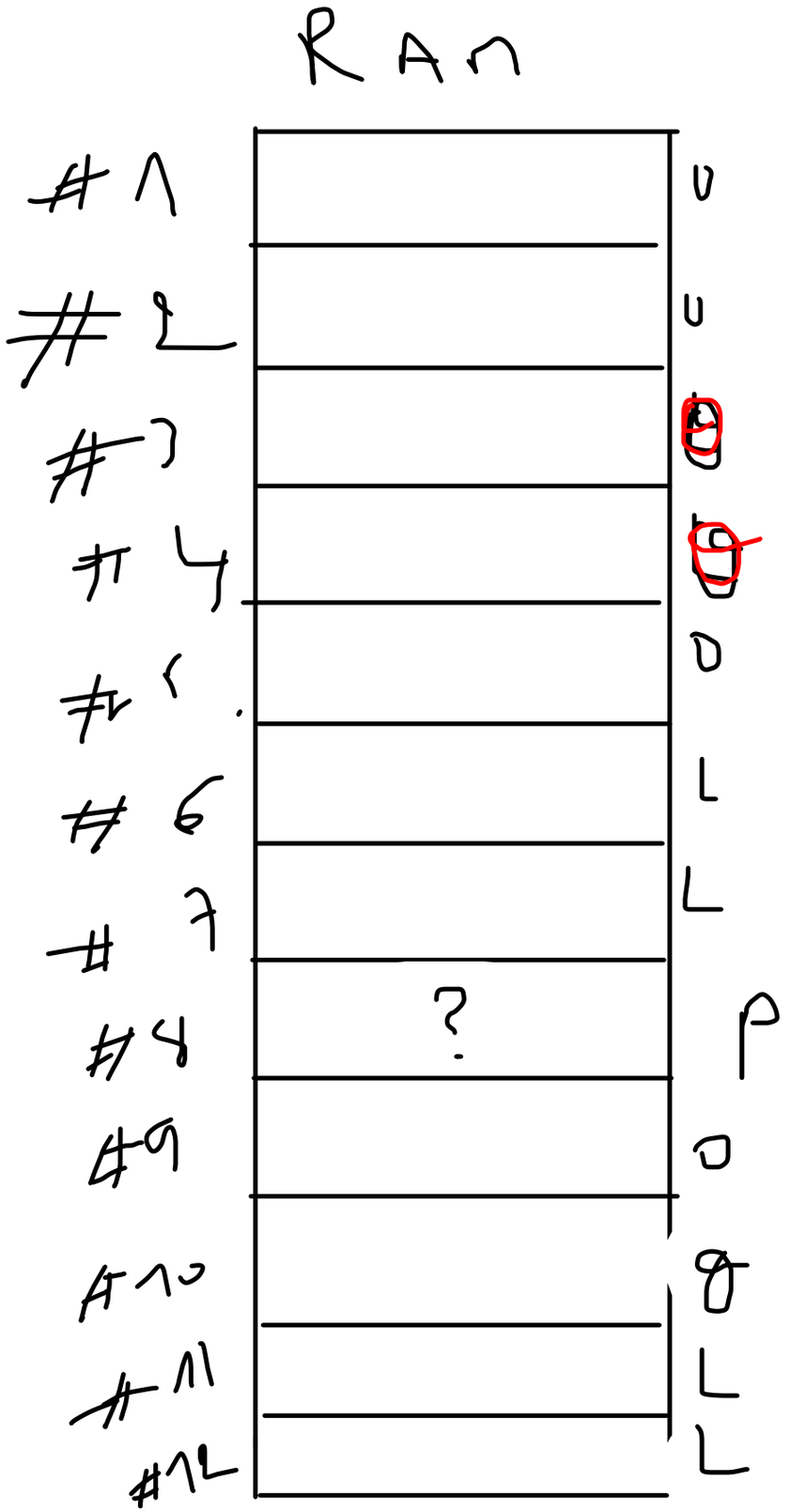
p = #7



int *p;

p = malloc(12)

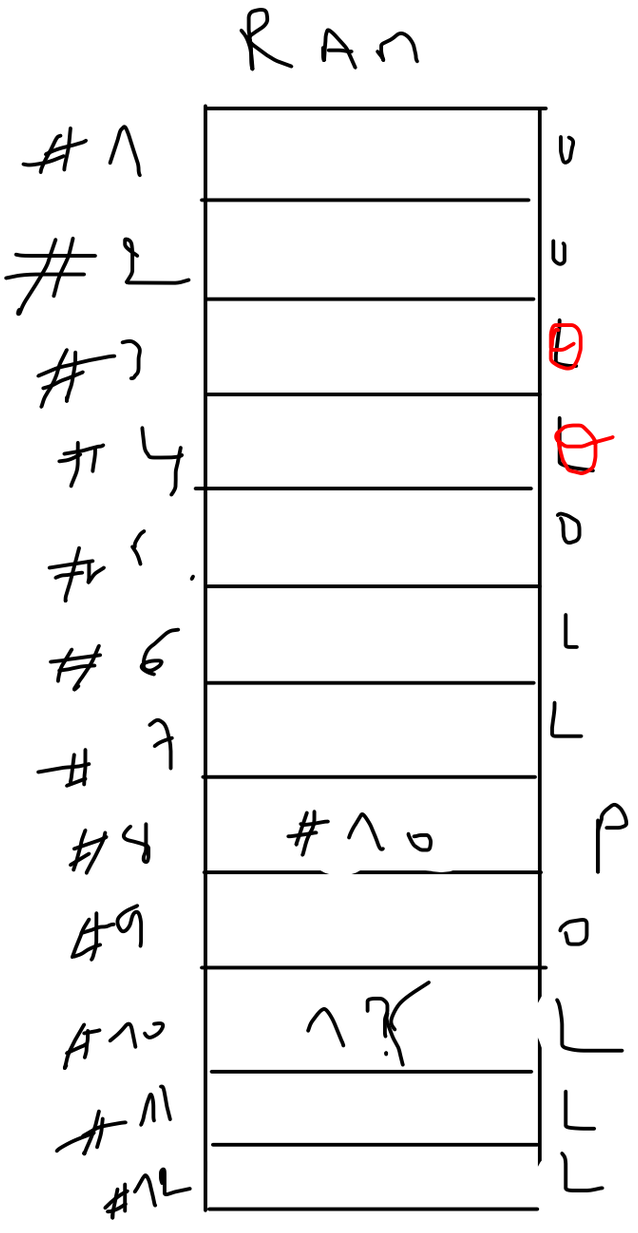
p = #8



int *p;

p = malloc(12)

p = NULL



int *p;

int a = 10

p = &a

*p = 10;

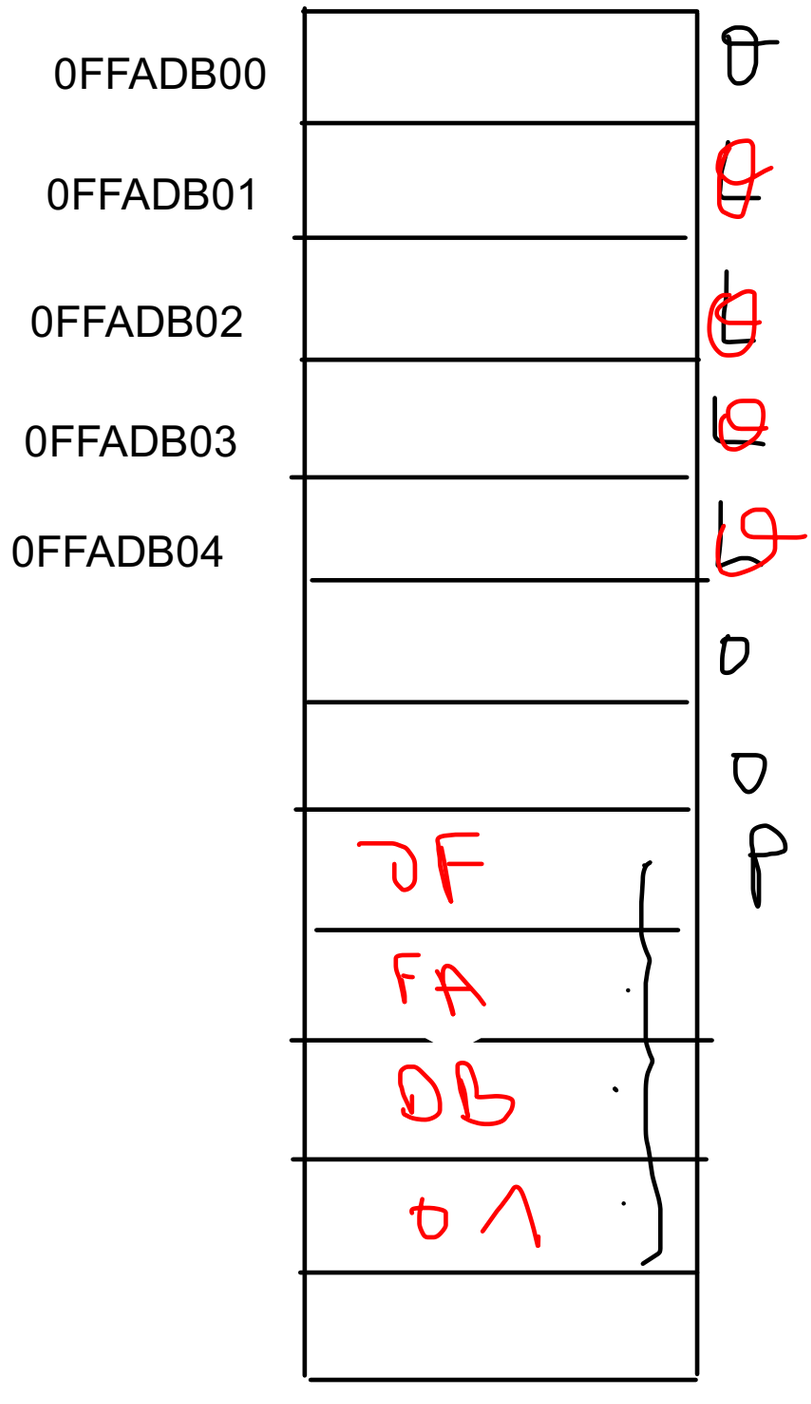
*p = 10

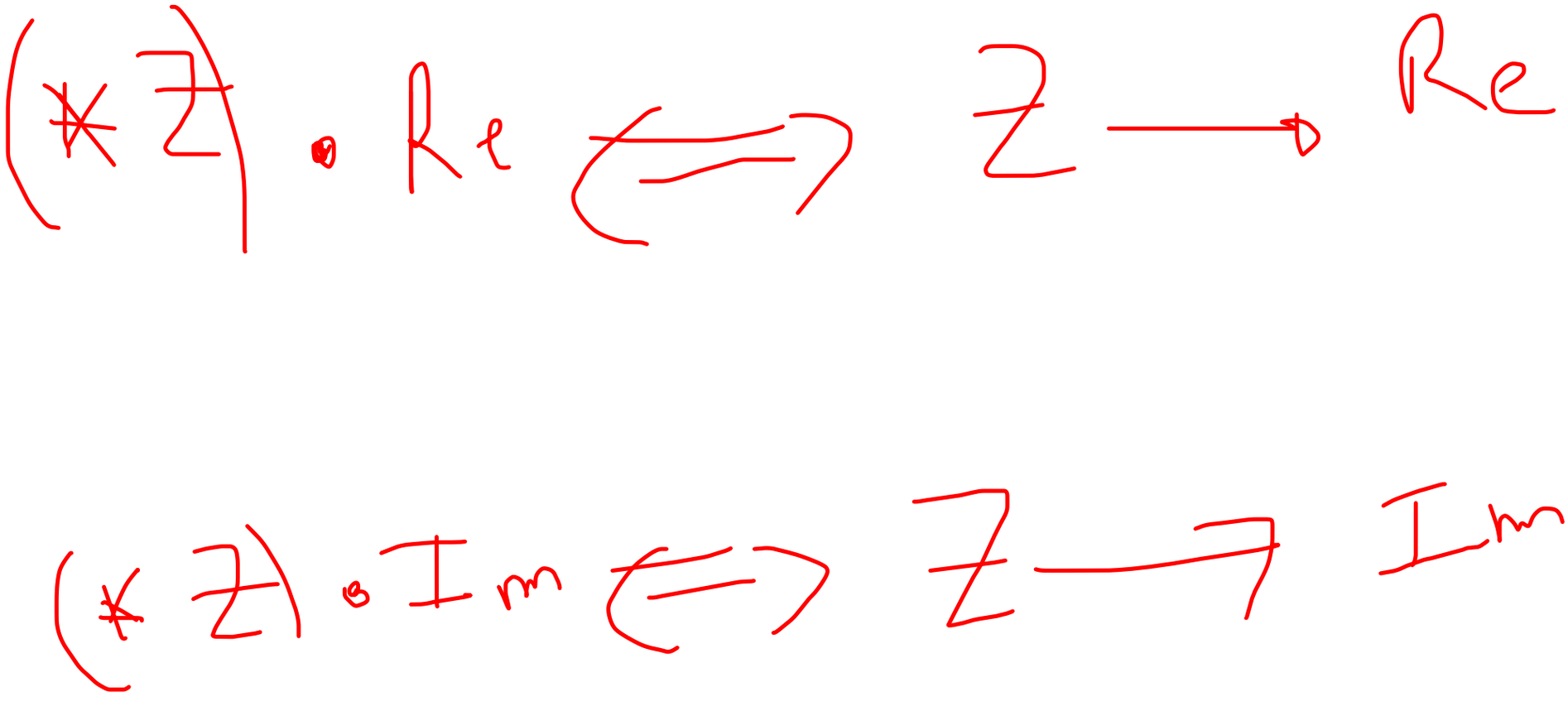
p = malloc(4)

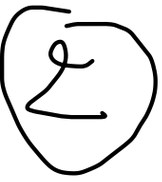
*p = 12

```
#include<stdio.h>
#include<stdlib.h>
//Réservation statique
main()
{
char *p;
c=(char*)malloc(4);
printf("la valeur de p est %p",p);
}
```

La valeur de c est 0FFADB01







void *calloc(N,sizeof(type)

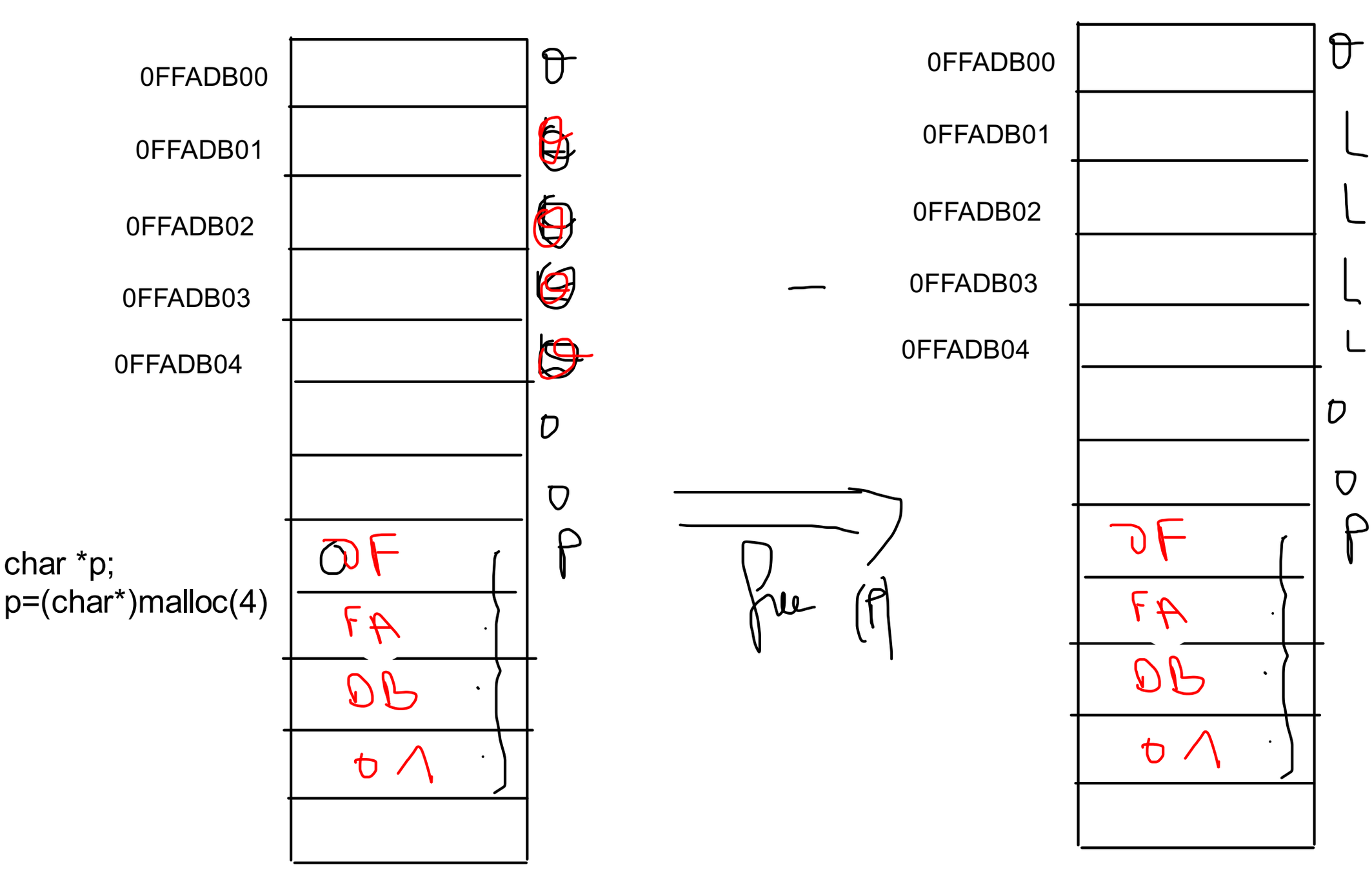
L'espace reservée sera initialisé

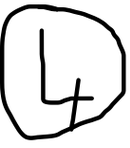
→ 0 (int | float)

→ '\0' (char)

③ free : permet de libérer l'espace réservé

free (p)


0FFADB00
0FFADB01
0FFADB02
0FFADB03
0FFADB04
char *p;
p=(char*)malloc(4)
free(p)
0FFADB00
0FFADB01
0FFADB02
0FFADB03
0FFADB04



void * realloc(p,size)