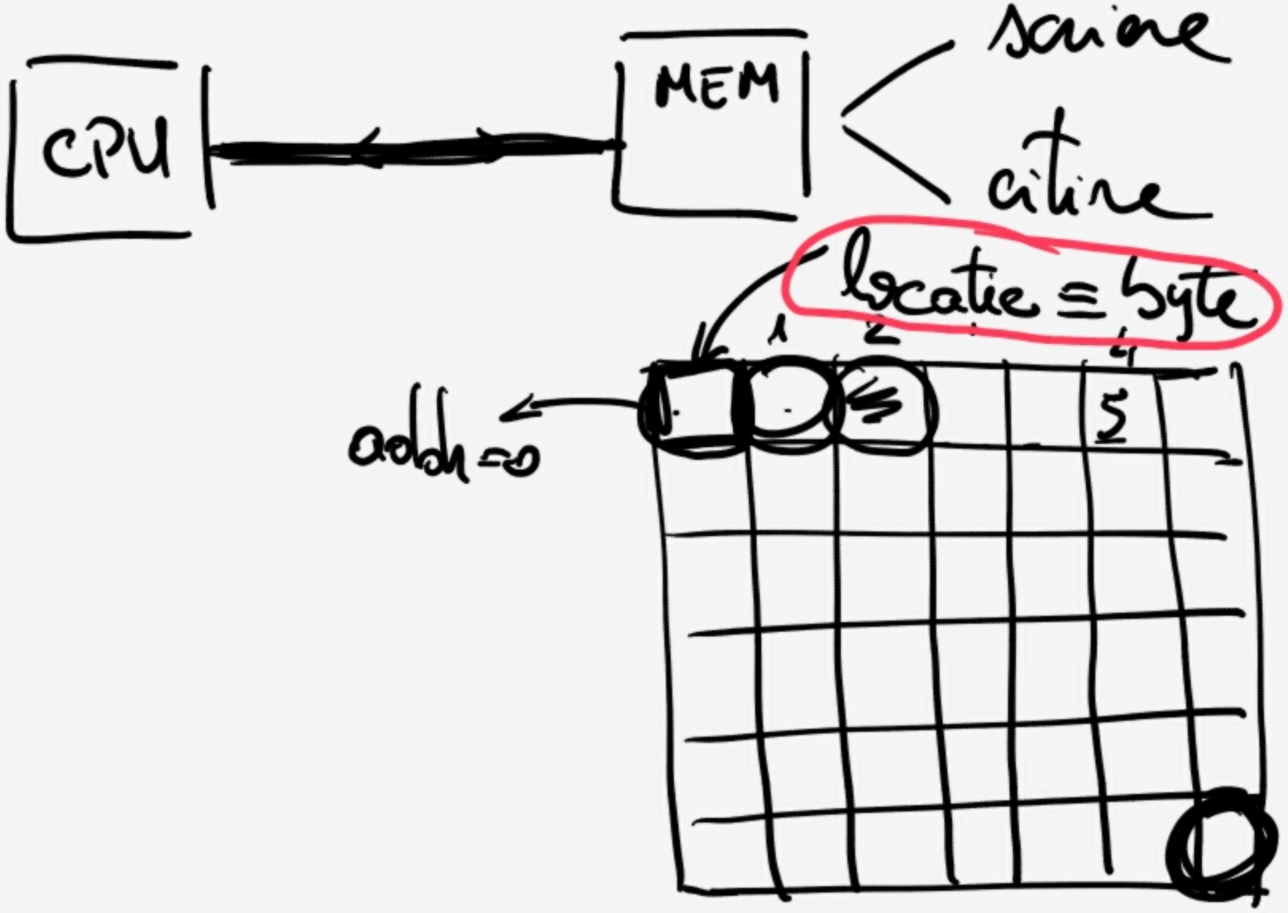
Wiki.DCAE.PUB.RO

TIPURI DE DATE

char - 1 byte
short → 2 bytes
int → 4 bytes
long → 8 bytes
long long → 8 bytes

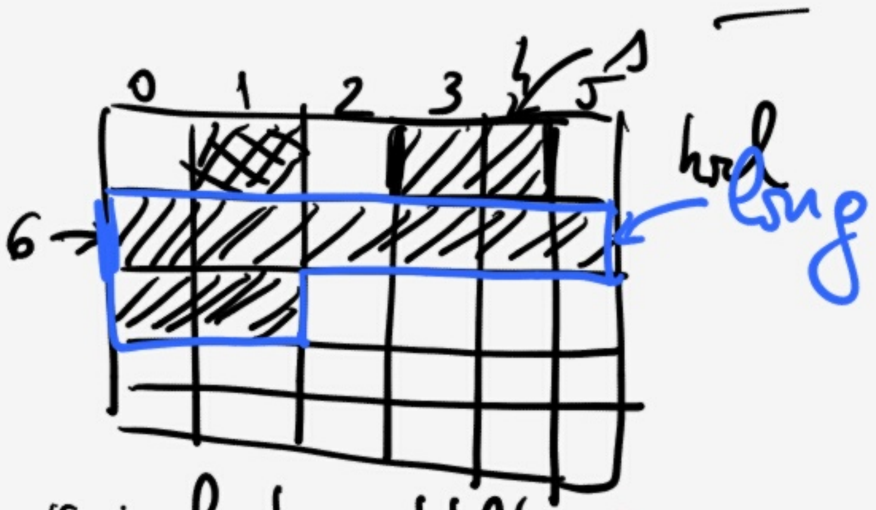
MEMORIE



$1024 = 2^{10}$

$1024^3 = 2^{30}$ → 30 biti ↔ 1GB MEM
31 biti ↔ 2GB MEM
32 biti ↔ 4GB MEM

1 GB = 1024 MB = 1024^2 kB
= 1024^3 B
≃ 1.000.000.000



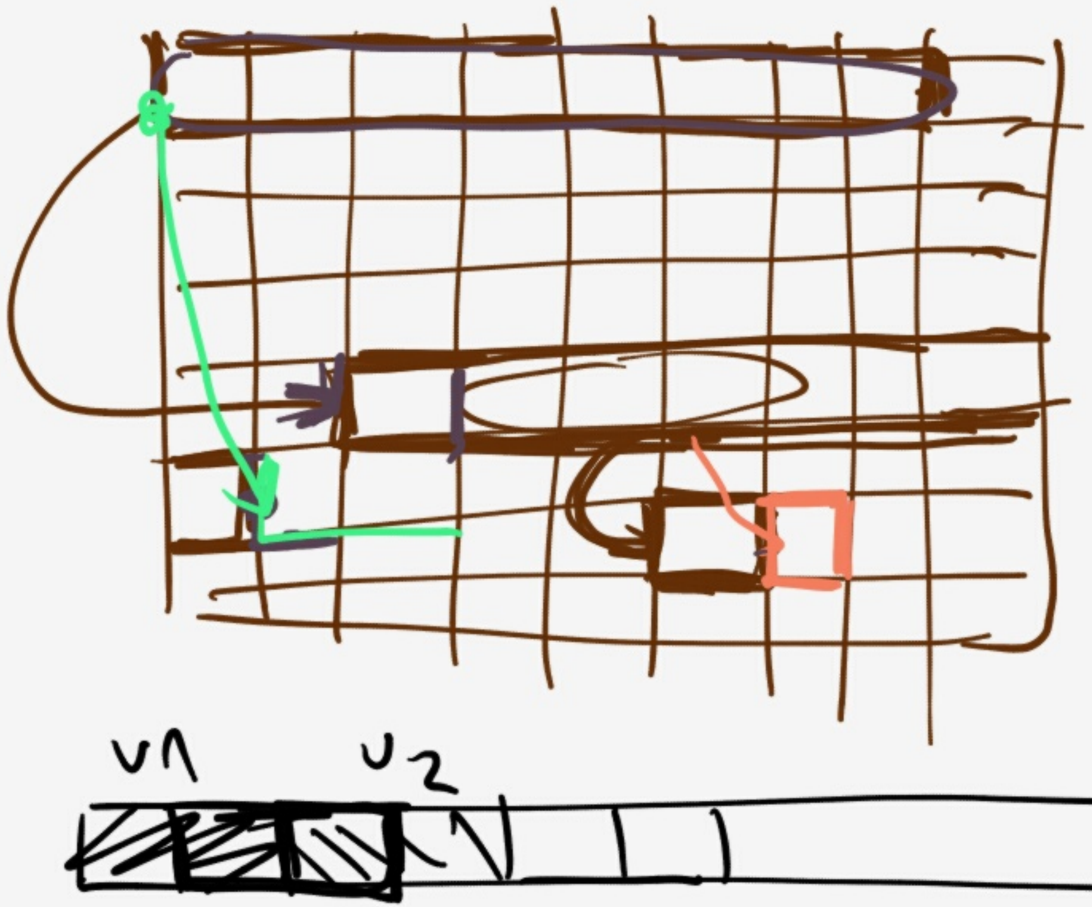
PHYSICAL MEM

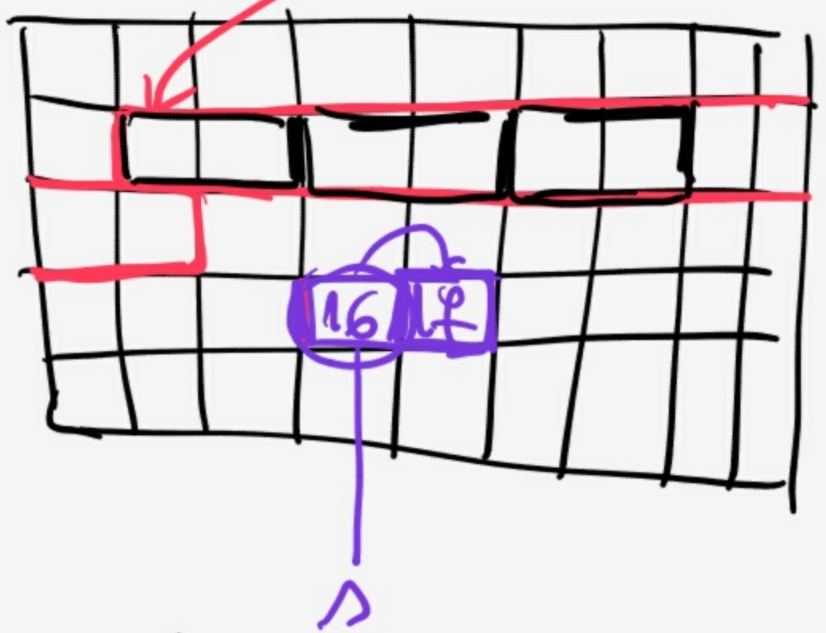
(1 << 2) |

```
#include <stdlib.h>

int globalInt;

int func(int a, char b){
    int c = a+5;
    return c;
}

char * func2(){
    return (char*) malloc(10);
}

int *p;
char *cp;
```

```
void test(){
    int *pi;
    pi = &globalInt;
    printf("%d", *pi);
    char *pc;
    pc = (char*)pi;
    printf("%d", *pc);
```

```
char **ppc;
(char*) *ppc;
char *pc = *ppc;
pc = pc + 1;
ppc = ppc + 1;
int a = 3;
ppc += a;
```





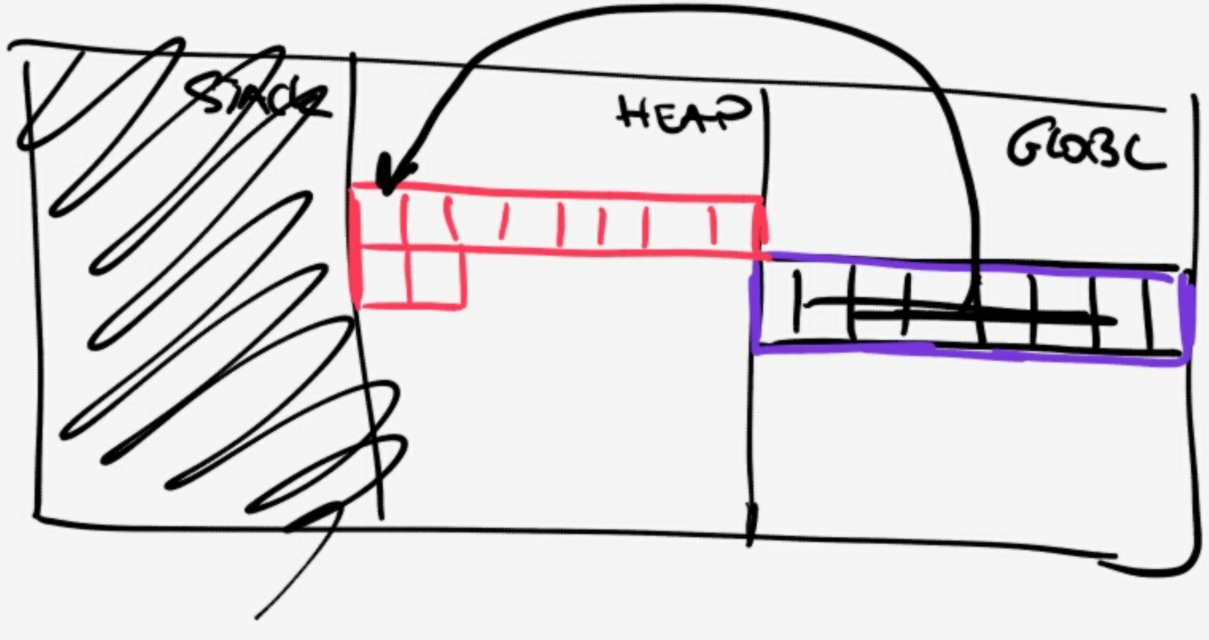
short A[5];

A[0]
A[1]
A[2]

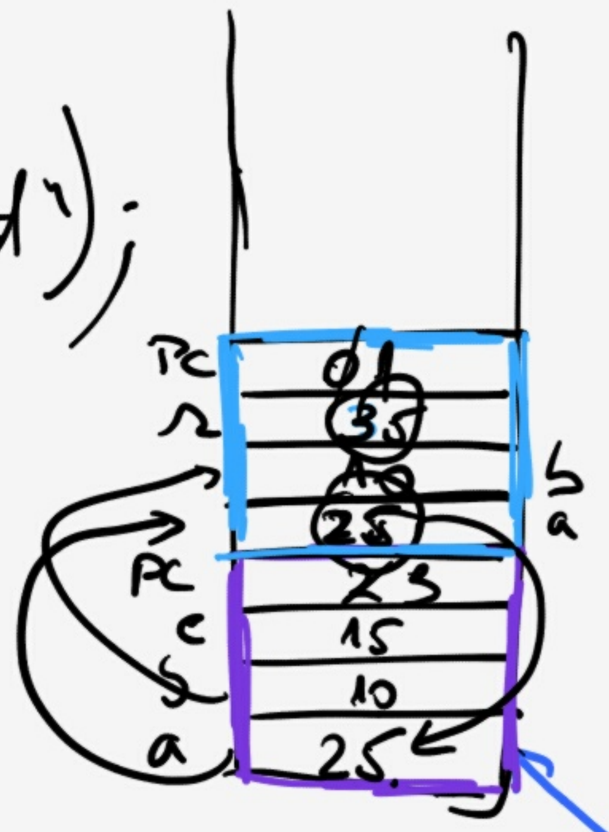
A[0] ≡ A+0
A[1] ≡ A+1
A[2] ≡ A+2

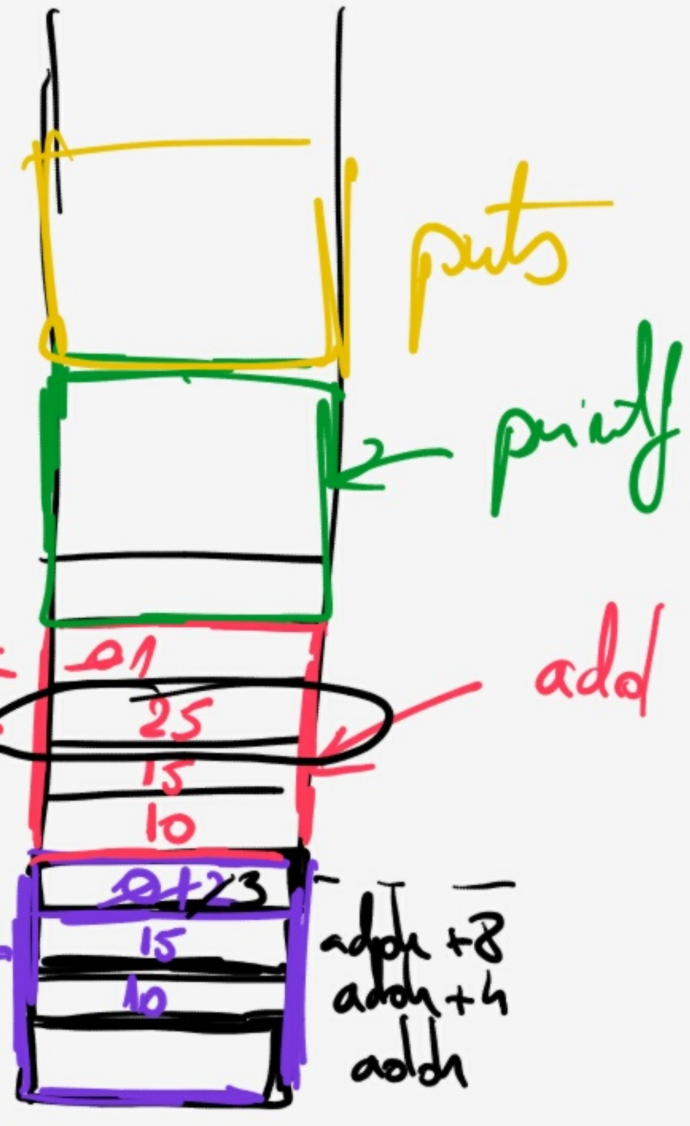
short A = 0x1211;
printf("%d", *((char*)&A)+1));

```c
#include<stdlib.h>

short *gArea;

void alloc(){
    gArea = (short*) malloc(5 * sizeof(short));
}
```



```c
#include<stdio.h>
int add(int a, int b){
    int r = a +b;
    return r;
}

int main(){
    int a, b = 10, c = 15;
    a = add(b, c);
    a = add(a, b);
    printf("%d", a);
}
```

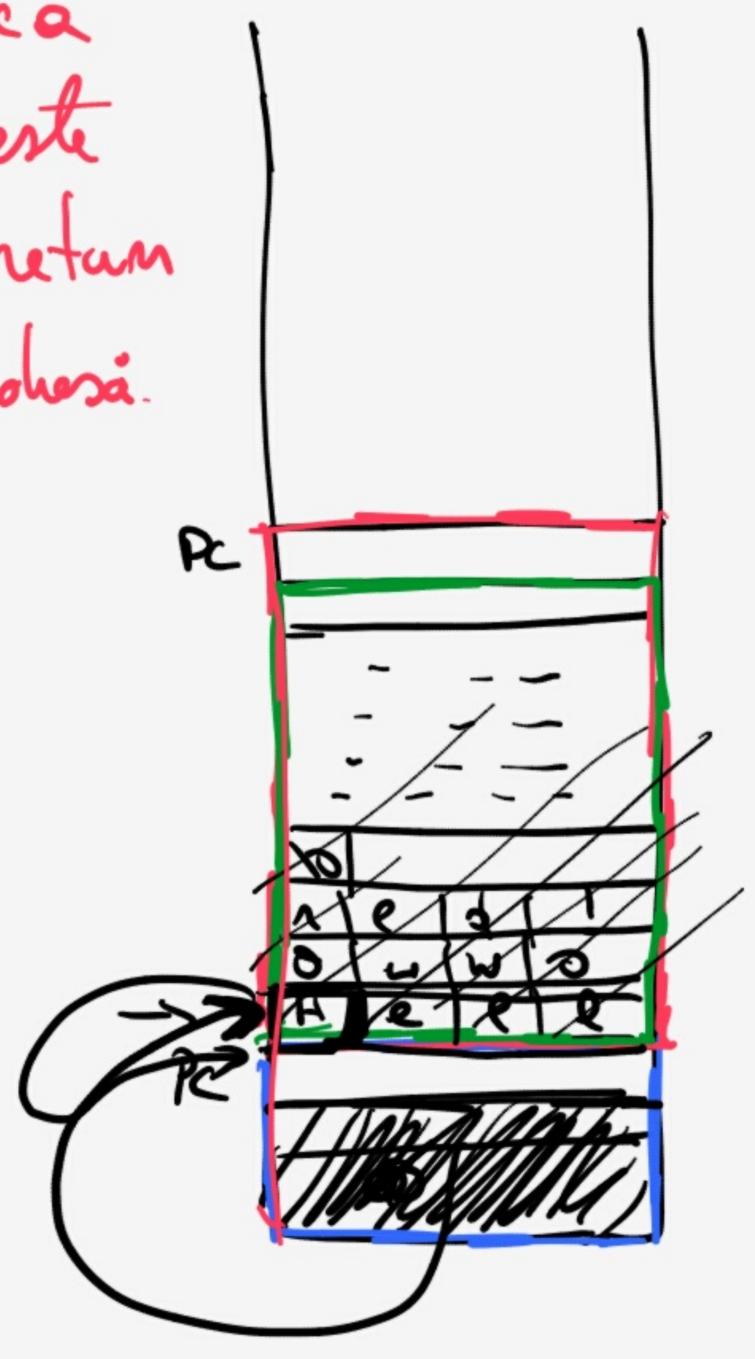




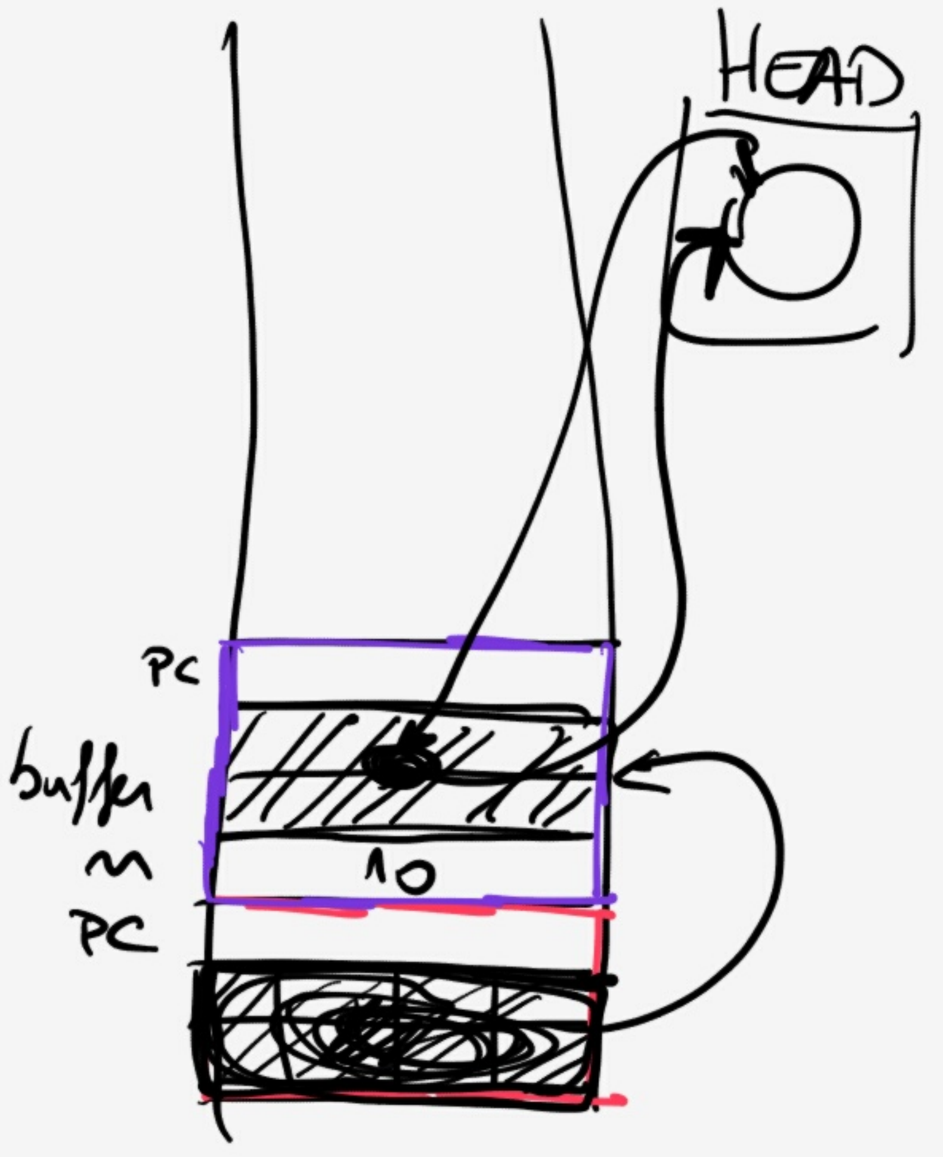

```c
int factorial (int n){
    return n * factorial(n-1);
}
```

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

char * getString(){
    char string[100];
    strcpy(string, "Hello world!", 12);
    return string;
}

int main(){
    char * string = getString();
    printf("String is:");
    printf("%s", string);
    return 0;
}
```

Când se alocă vecte local, nu este legal să dați return la adresă.



```c
#include <stdlib.h>

void alloc(int n, int *buffer){
    buffer = (int*)malloc(sizeof(int) * n);
}

int main(){
    int * newBuffer;
    alloc(10, newBuffer);
    return 0;
}
```



```
void alloc(int n, int **buffer){
    *buffer = (int*) malloc(sizeof(int) * n);
}

int main(){
    int *buffer;
    alloc(10, &buffer);
    return 0;
}
```