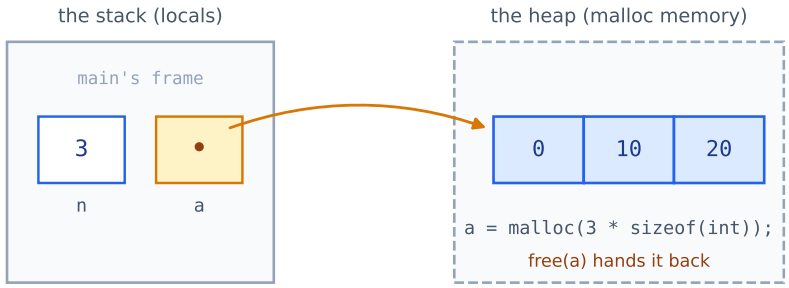
Dynamic memory

C Reference

malloc, free & realloc

`malloc` allocates 分配 memory on the heap 堆 at run time and returns a pointer to it. `realloc` resizes that block; `free` gives it back. Every `malloc` needs a matching `free`, or you get a memory leak 内存泄漏. These live in `<stdlib.h>`.



09-stack-heap

Locals live on the stack; malloc memory lives on the heap until you free it

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

int main(void) {
    int n = 3;
    int *a = malloc(n * sizeof(int));    // room for 3 ints
    for (int i = 0; i < n; i++) a[i] = i * 10;

    a = realloc(a, 4 * sizeof(int));     // grow to 4 ints
    a[3] = 30;

    for (int i = 0; i < 4; i++) printf("%d ", a[i]);
    printf("\n");                        // 0 10 20 30

    free(a);                             // hand the memory back
    return 0;
}
```


- `calloc(n, size)` allocates an array like `malloc` AND fills it with zeros.

Common mistakes

- Every `malloc` needs a matching `free`; forgetting to free leaks memory.
- Using memory after `free` (a "use-after-free") is a serious bug.
- Check that `malloc` did not return `NULL` before you use the memory.