

Functions

C Reference

Parameters & return values

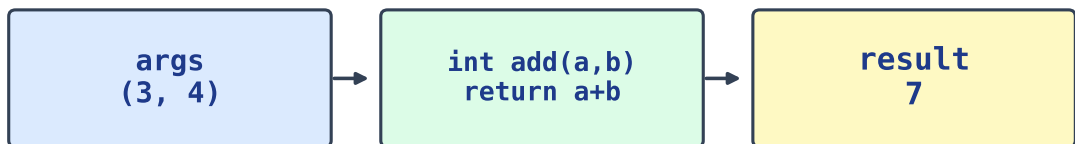
A function takes parameters (inputs) and gives back a return value . The return type comes first (int, double, ...); void means it returns nothing.

```
#include <stdio.h>

int square(int x) {           // x is a parameter
    return x * x;             // hand back a value
}

int main(void) {
    int r = square(5);
    printf("%d\n", r);        // 25
    return 0;
}
```

Call → compute → return



Exam tip: declare the return type; parameters need types; return ends the function

Parameters in; return value out

Prototypes & scope

C reads top to bottom, so a function must be known before it is called. A prototype — the header line plus ; — declares it early so you can keep `main` first. A variable's scope is the block it lives in: it is local and disappears when the block ends.

```
#include <stdio.h>

int add(int a, int b);    // prototype: declared before use

int main(void) {
    printf("%d\n", add(3, 4));    // 7
    return 0;
}

int add(int a, int b) {    // definition comes later
    int sum = a + b;       // sum is local to add
    return sum;
}
```

Common mistakes

- A function used before it is defined needs a prototype above `main`.
- Arguments are passed BY VALUE — changes inside a function do not affect the caller unless you pass a pointer.
- A variable declared inside a function is local to it.