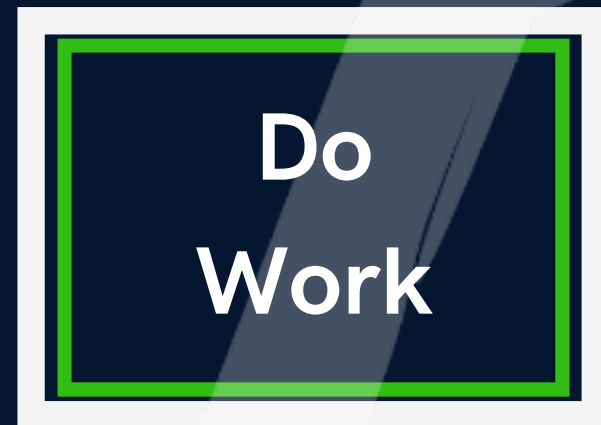


Functions



block of code that performs particular task

Take
Argument



Return
Result

it can be used **multiple** times

increase code **reusability**

Syntax 1

Function Prototype

```
void printHello(); ←
```

A



> Tell the compiler

Syntax 2

Function Definition

```
void printHello() {  
    printf("Hello");  
}
```



> Do the Work

Syntax 3

Function Call

```
int main() {  
    printHello();  
    return 0;  
}
```



> Use the Work

Properties

- Execution always starts from main
- A function gets called directly or indirectly from main
- There can be multiple functions in a program

Function Types



**Library
function**

Special functions
inbuilt in C

`scanf()`, `printf()`

**User-
defined**

declared & defined by
programmer

Passing Arguments

functions can take value & give some value

parameter

return value



Passing Arguments

void **printHello**(); ←

void **printTable**(int n); ←

int **sum**(int a, int b); ←



Passing Arguments

functions can take value & give some value

parameter

return value



Argument v/s Parameter

values that are
passed in
function **call**

used to **send**
value

actual
parameter

values in function
declaration &
definition

used to **receive**
value

formal
parameters

NOTE

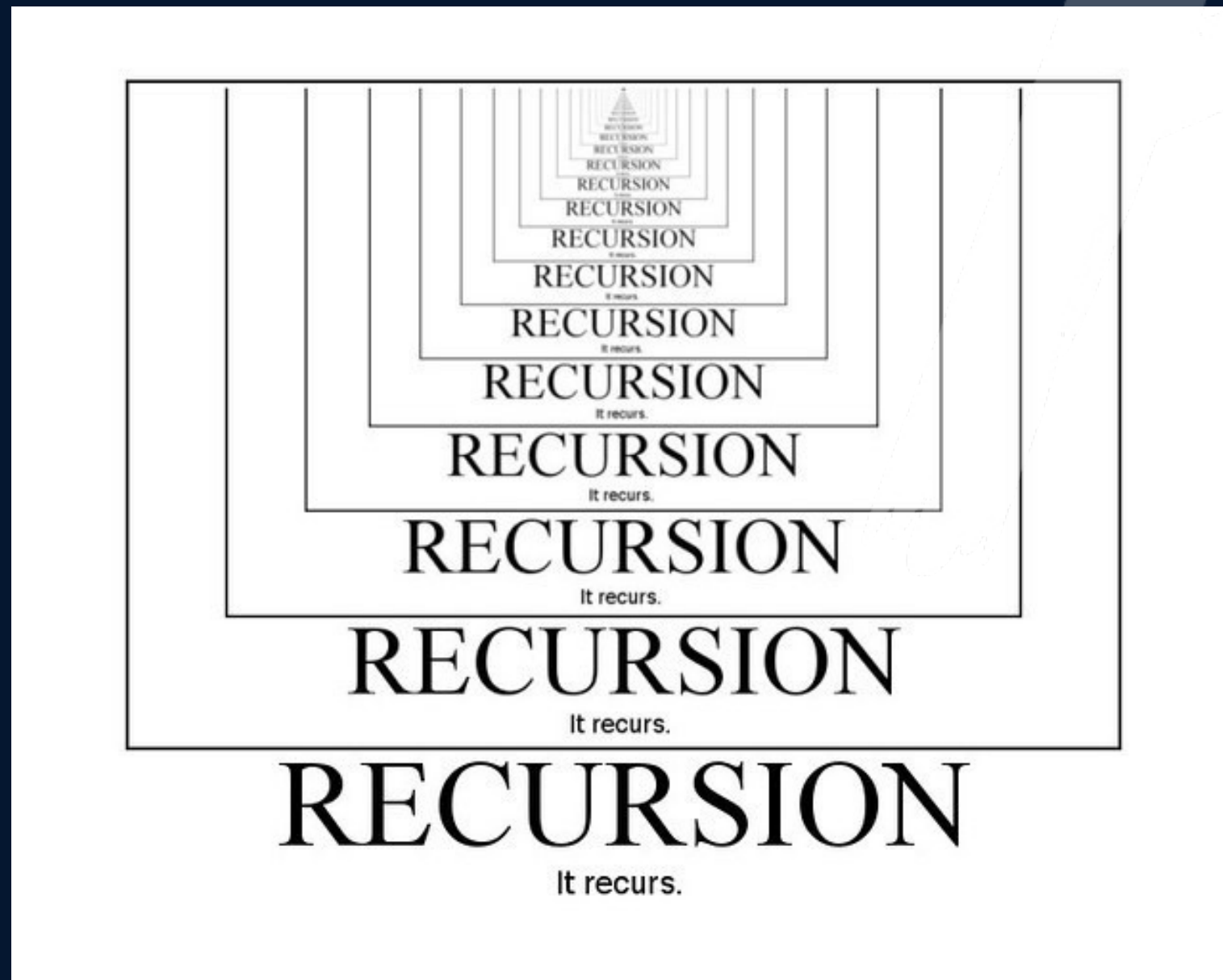
- a. Function can only return one value at a time
- b. Changes to parameters in function don't change the values in calling function.

Because a copy of argument is passed to the function

Recursion



When a **function calls itself**, it's called recursion



Properties of Recursion

- a. Anything that can be done with Iteration, can be done with recursion and vice-versa.
- b. Recursion can sometimes give the most simple solution.
- c. **Base Case** is the condition which stops recursion.
- d. Iteration has infinite loop & Recursion has **stack overflow**