2D Arrays

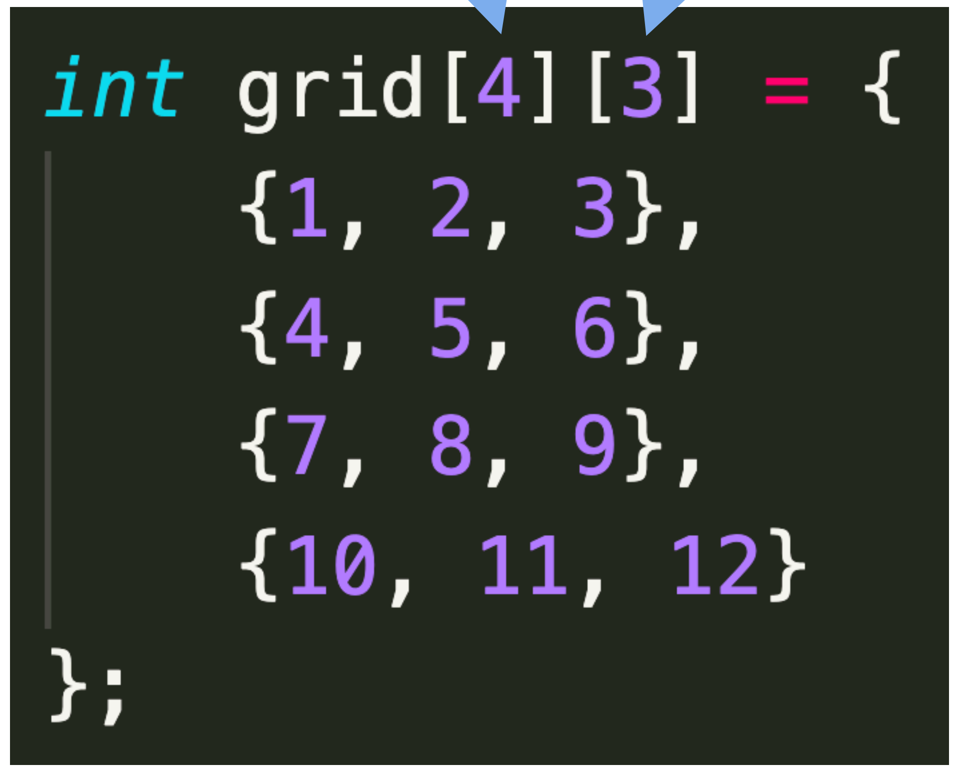
What are they?

- An array of arrays.
- We call them 2D arrays because they we can view them as a grid.

Sample Declaration and Initialisation

number of 1D arrays (num rows)

number of elements in each 1D array (num columns)



```
int grid[4][3] = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9},
    {10, 11, 12}
};
```

remember it's better style to use **`#define`**'d constants for the size

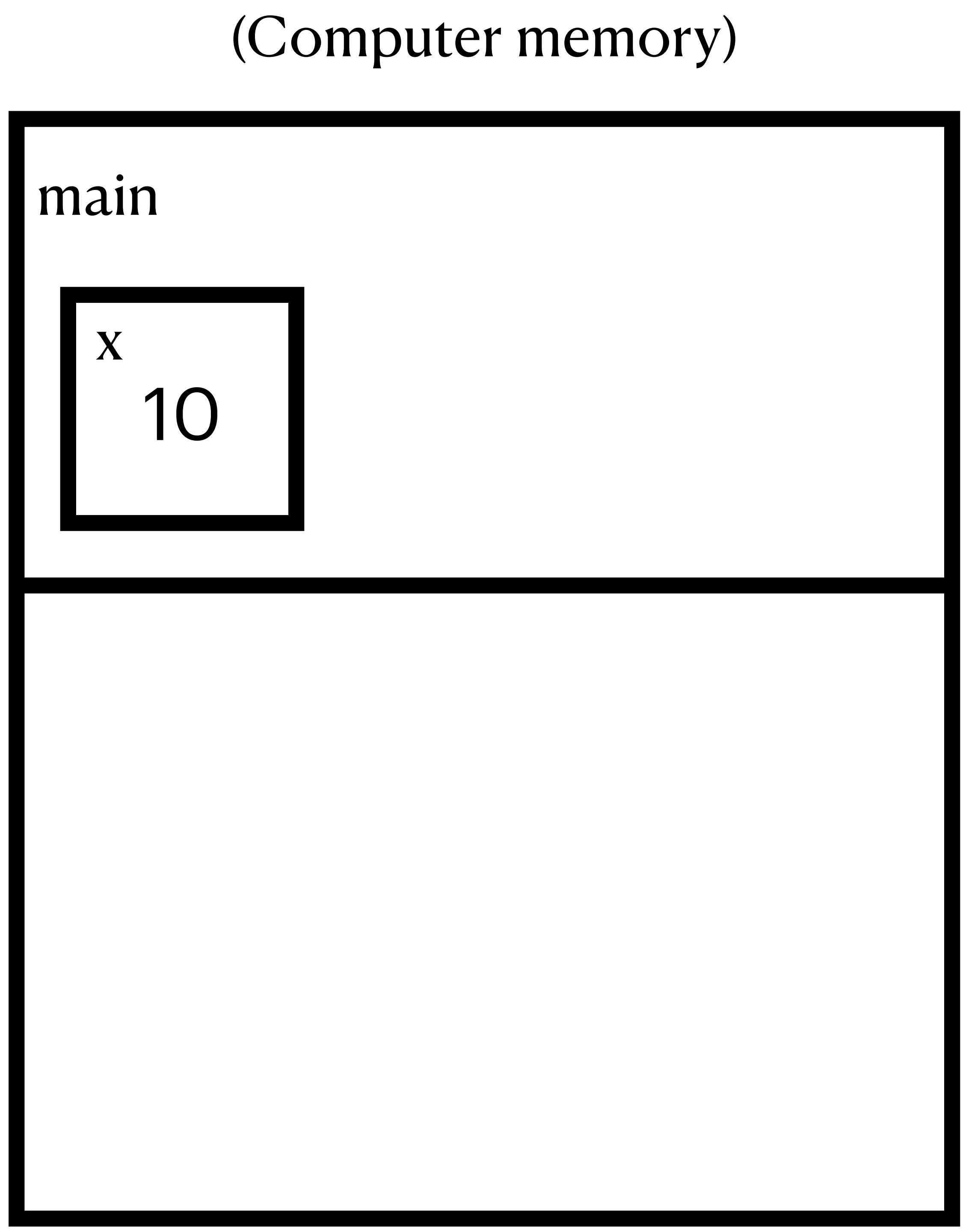
Visualisation

Index	0	1	2
0	1	2	3
1	4	5	6
2	7	8	9
3	10	11	12

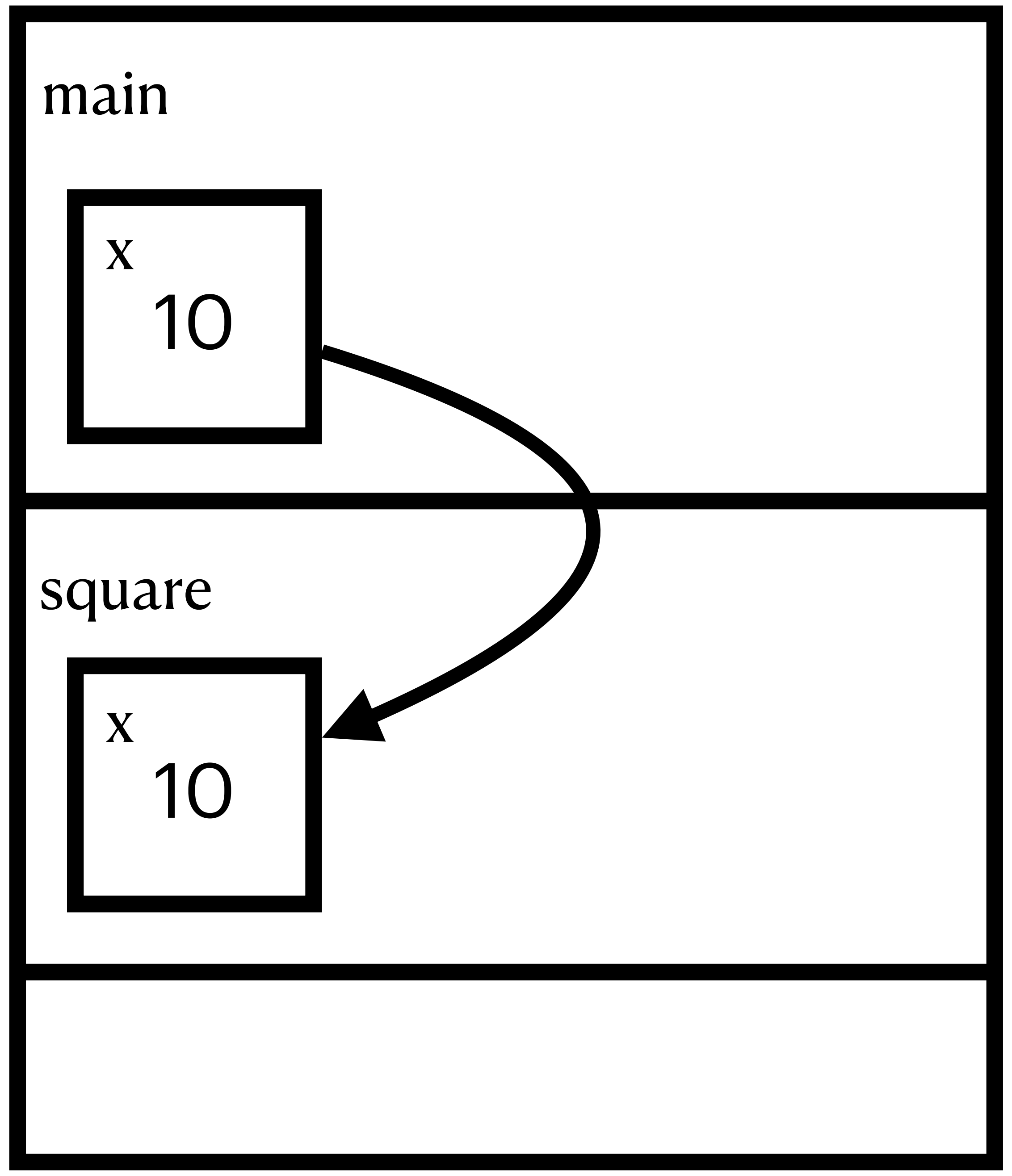
`grid[2][1]`

Note: row number is indexed first before column number

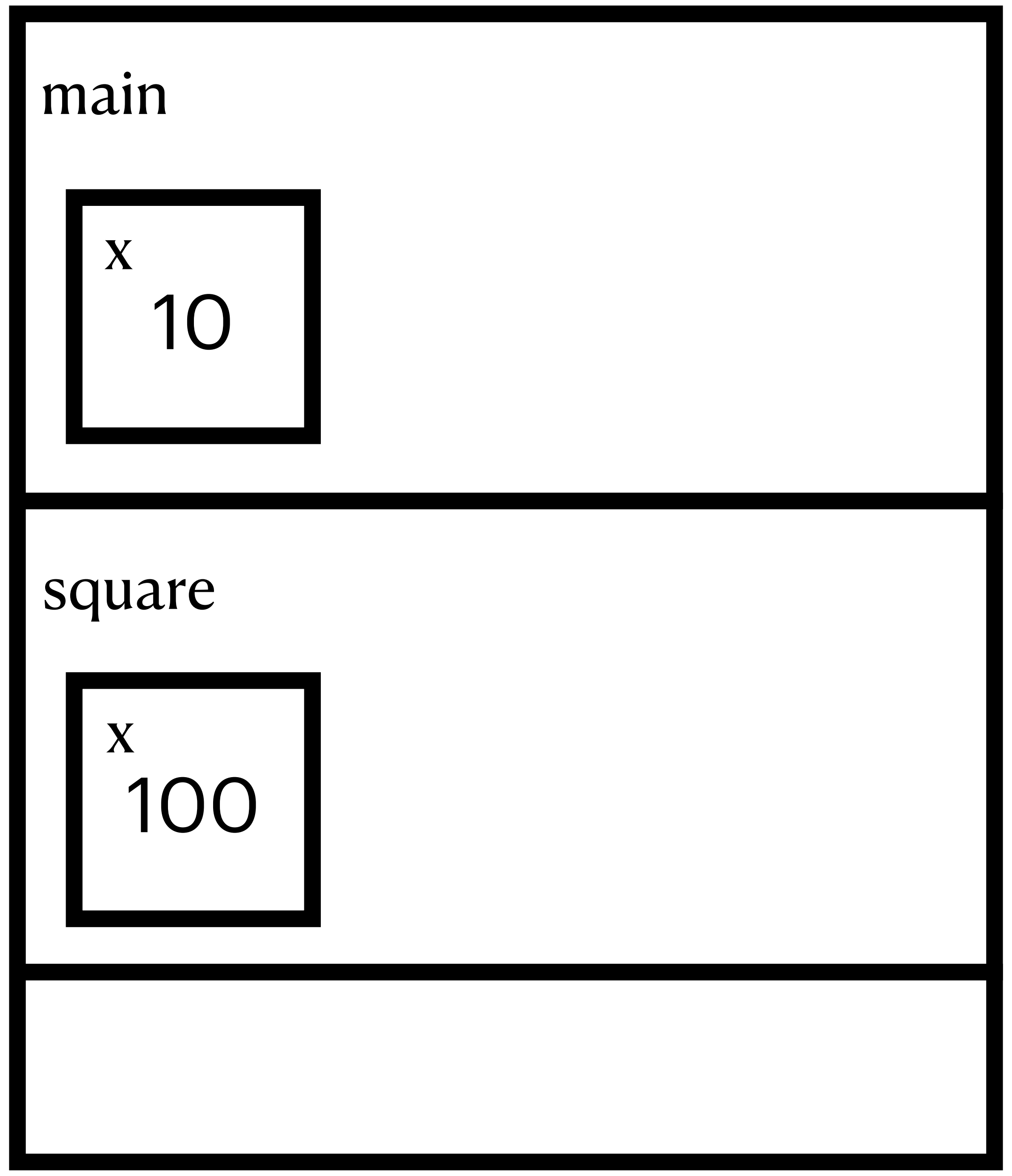
Memory Model: Function Calls



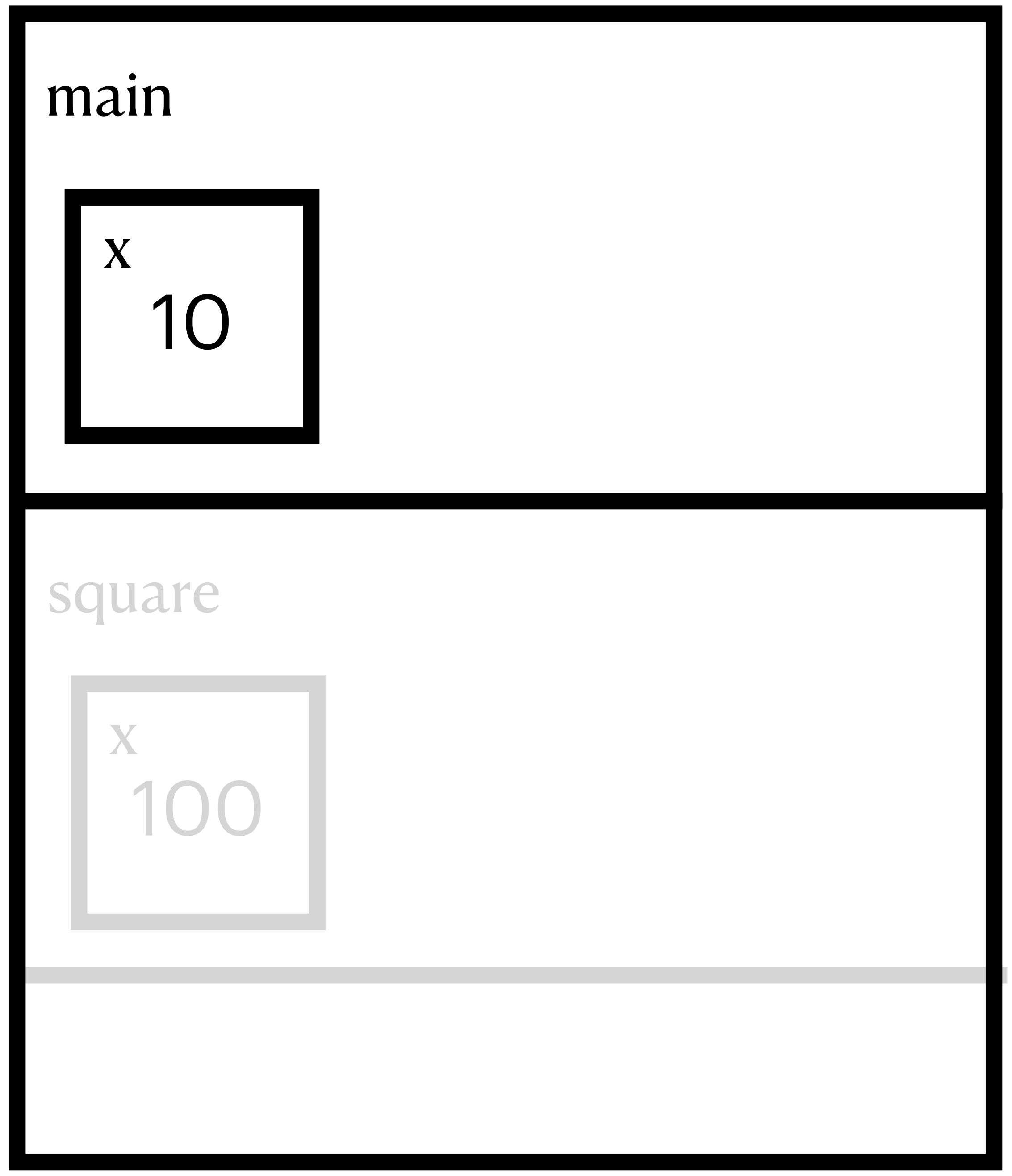
In the beginning, `main` has its own block of memory. The variable `x` is given the value 10.



`main` calls `square`. The value 10 is copied over to the block of memory for `square`

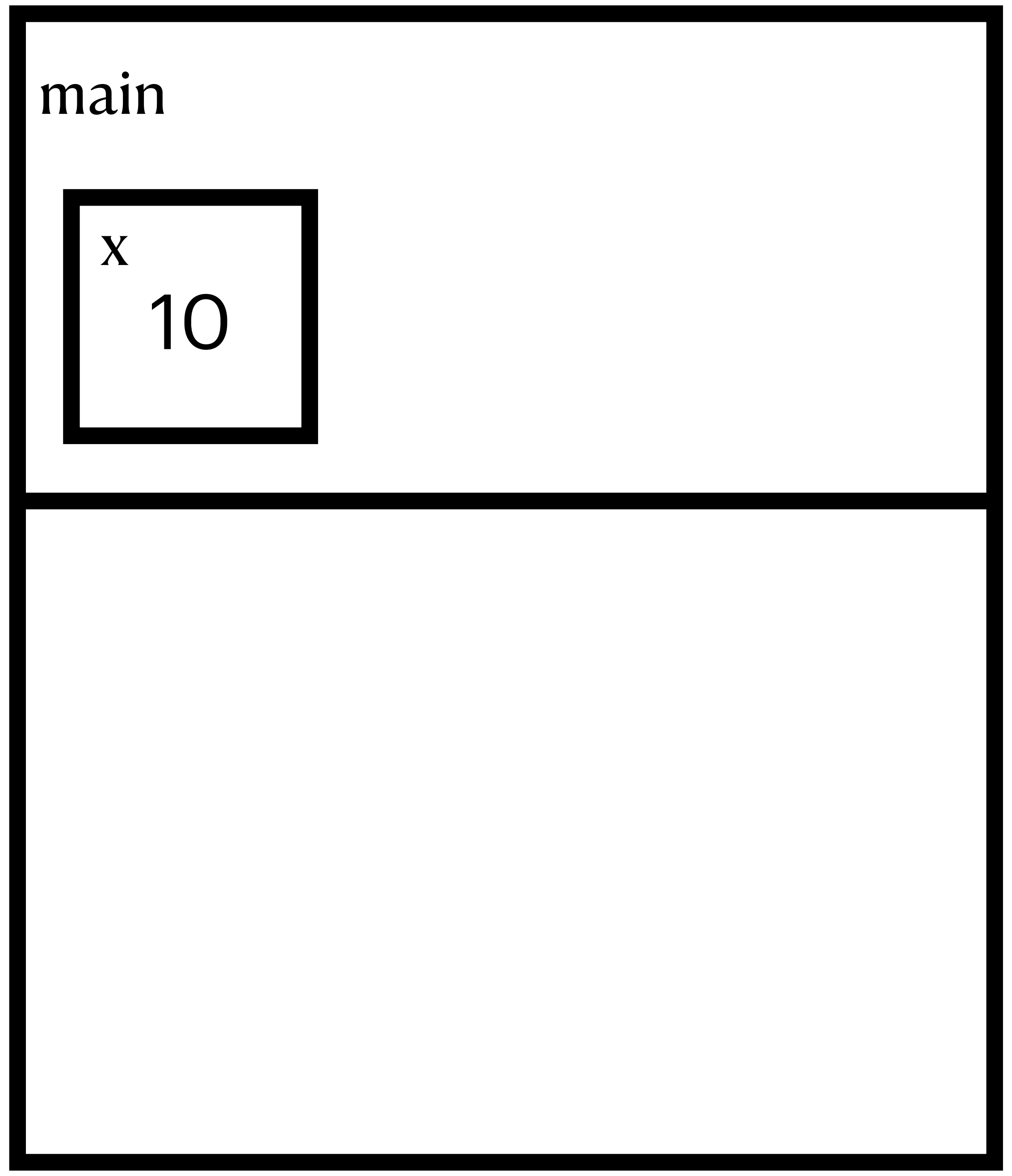


`square` changes the value of `x` at the block of memory that was assigned to it.

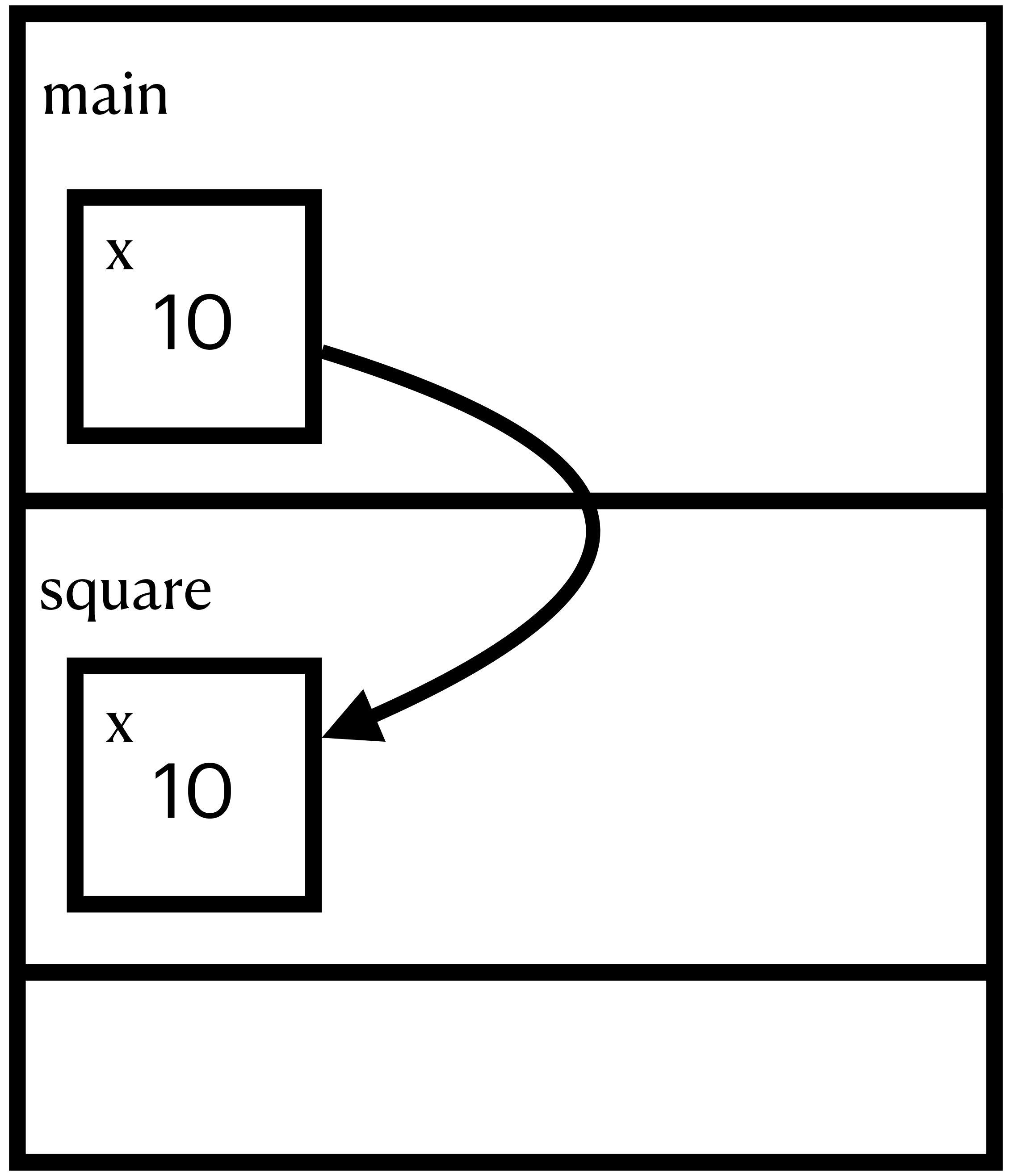


when we exit `square`, its memory is destroyed and the `x` in `main` remains unchanged

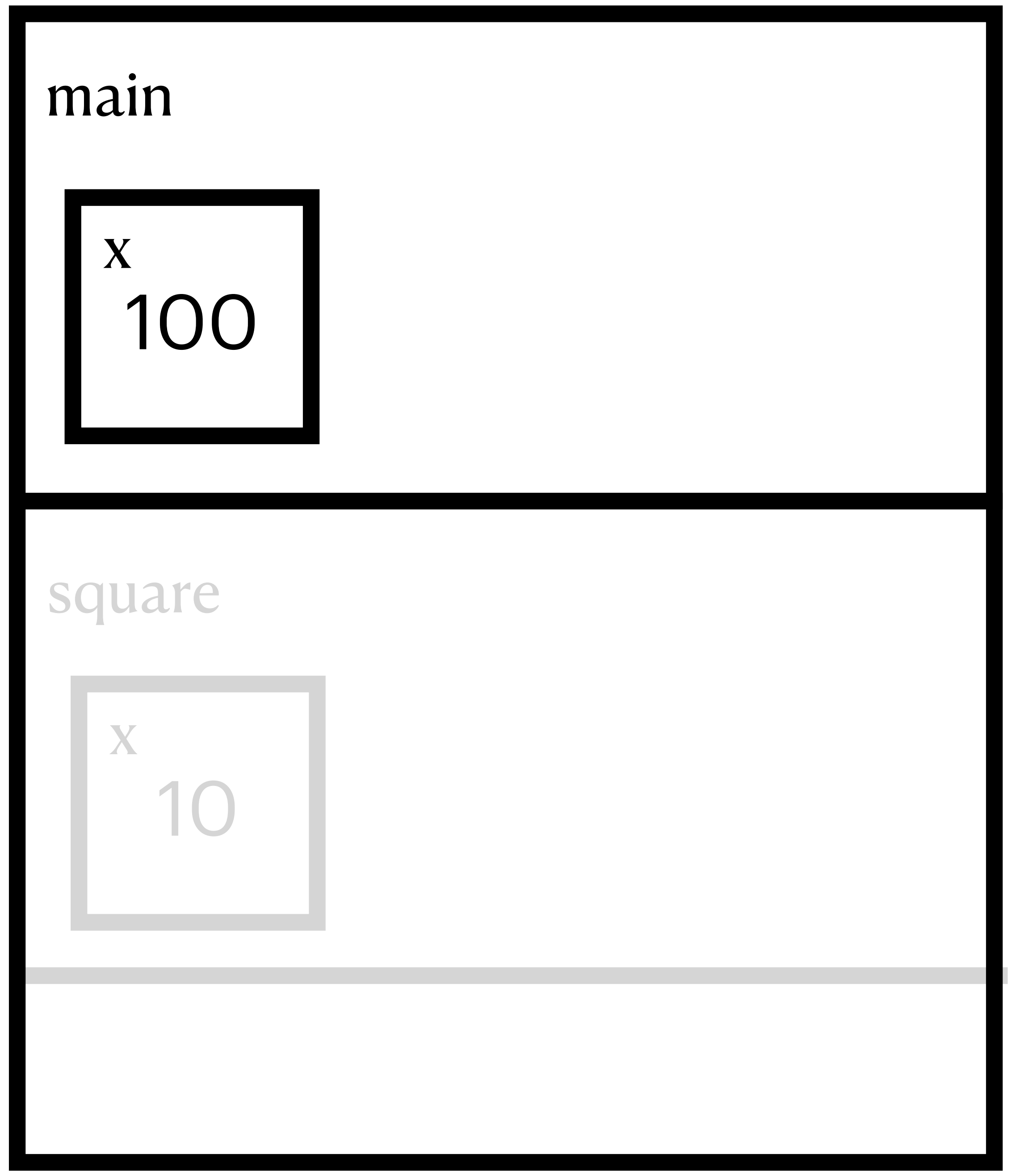
Memory Model: Return Value



In the beginning, `main` has its own block of memory. The variable `x` is given the value 10.



`main` calls `square`. The value 10 is copied over to the block of memory for `square`



when we exit `square`, its memory is destroyed and the `x` in `main` now takes on the value of whatever is returned by `square`!