C Pointers and Arrays

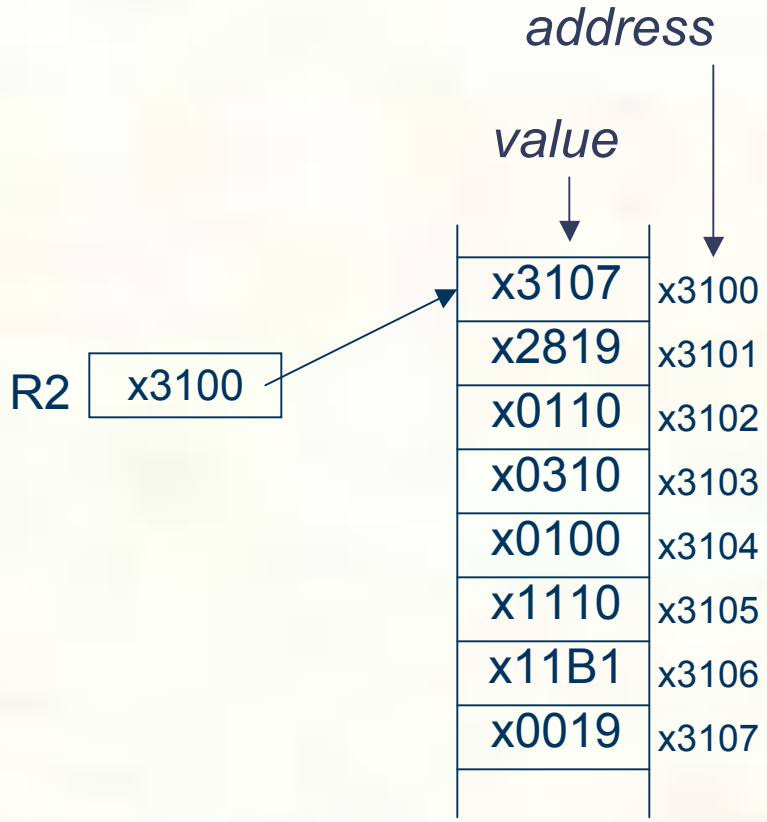
Pointers and Arrays

- We've seen examples of both of these in our LC-3 programs; now we'll see them in C.

- Pointer
  - Address of a variable in memory
  - Allows us to indirectly access variables
    - in other words, we can talk about its *address* rather than its *value*
- Array
  - A list of values arranged sequentially in memory
  - Example: a list of telephone numbers
  - Expression `a[4]` refers to the 5th element of the array `a`

Address vs. Value

- Sometimes we want to deal with the address of a memory location, rather than the value it contains.
- Recall example from Chapter 6: adding a column of numbers.
  - R2 contains address of first location.
  - Read value, add to sum, and increment R2 until all numbers have been processed.
- R2 is a pointer -- it contains the address of data we're interested in.

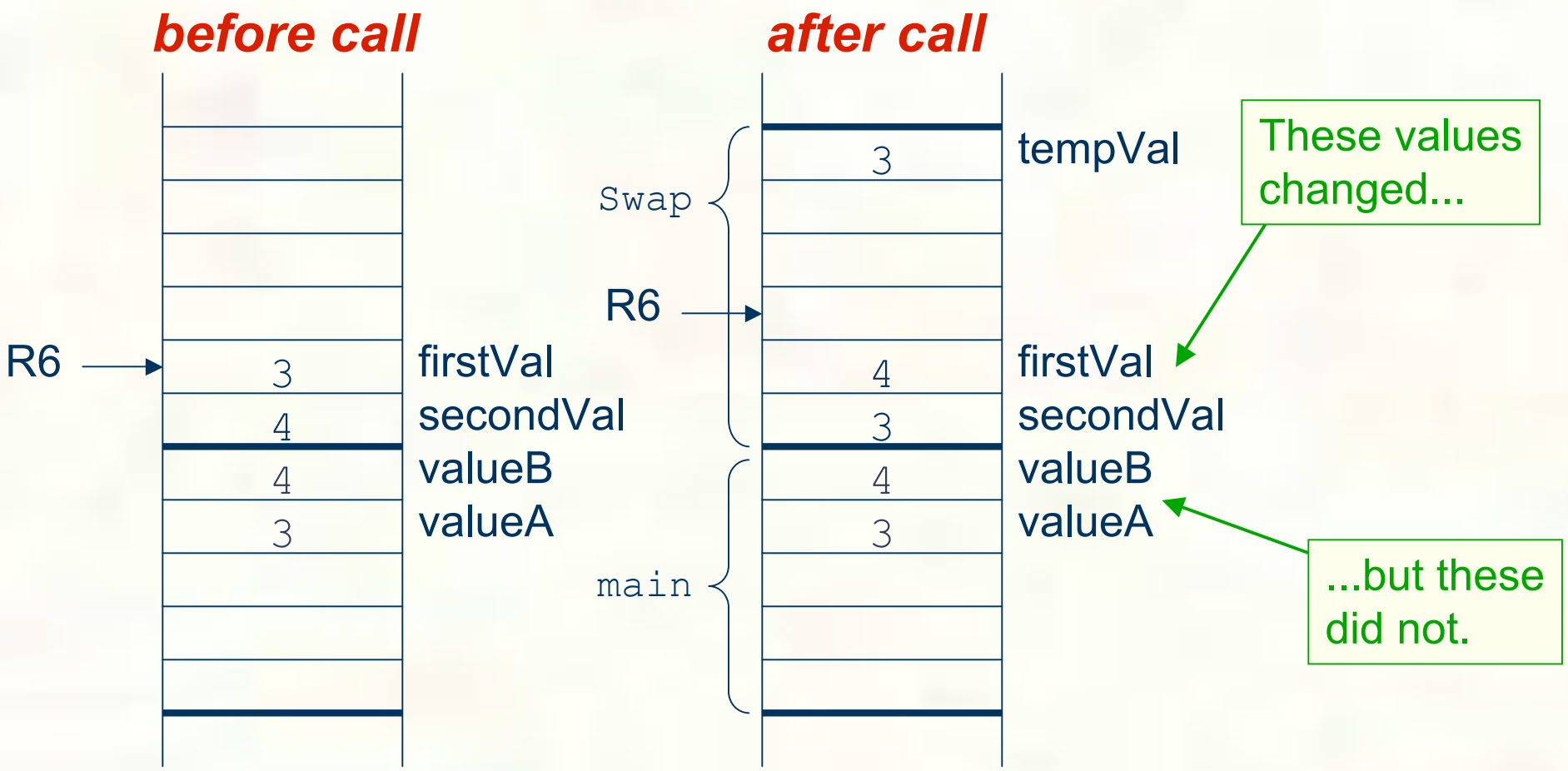


Another Need for Addresses

- Consider the following function that's supposed to swap the values of its arguments.

```
void Swap(int firstVal, int secondVal)
{
  int tempVal = firstVal;
  firstVal = secondVal;
  secondVal = tempVal;
}
```

Executing the Swap Function



Swap needs addresses of variables outside its own activation record.