

* So how large each of the data types, typically, is in C?

BOOLEAN VALUE - 0 or 1 — 1 byte
T or F

CHAR — 1 BYTE OR 8 BITS

INT — 4 BYTES OR 32 BITS

FLOAT — 4 BYTES OR 32 BITS

LONG — 8 BYTES OR 64 BITS

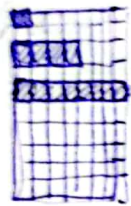
STRING — (?) BYTES (just for today)
it depends.

RAM memory → each of these little black chips on this circuit board, the green thing, these little black chips are where 0s and 1s are actually stored.

If this is like, 1GB of memory, for the sake of discussion, it stands to reason that, if this thing is storing 1 billion bytes, 1 gigabyte,

then we can number them, arbitrarily.

maybe this will be byte 0, 1, 2, 3, 4, 5, 6, 7, 8, ...
and maybe down in the bottom is byte number
1 billion. We can just NUMBER these things or
might be our convention. So, graphically, we
have something like that:



the data is taking up some number of
bytes. So if you're storing a CHAR
in a computer's memory, which was 1
byte, it might be stored at this top left-hand
location of this block chip of memory.

If you were to store something like an integer
that was 4 bytes, as well, it might use 4 of
these bytes but they're going to be contiguous, in
this case.

If you were to store a LONG or a DOUBLE, you
might, actually, need 8 bytes. So I'm filling
in these squares to represent how much me-
mory and given variable of some data type would
take up.

So memory is being a grid ... Or like a
room that we can point only types of data

into that we want. At the end of the day, all
of this data is just going to be 0s and 1s.

But it's up to you and I to build struc-
tures on top of these things like actual
numbers, colors, images, movies and beyond.

→ So I have a program that needs 3 integers.

↳ SCORES.C and 3, I need to
convert to float.

1/3 So I divide for 13.0 I'm treating
that as a float now.
YES! ok now it's correct. ←

But ... if you ever want to have to take the
average of any number of scores other than
3 ? So it turns out, there are better ways,
in languages like C, if you want to have
multiple values stored in memory that
happened to be of the same data type.

what's really being stored in the computer's
memory are patterns of 0s and 1s / 32 total
because 4 bytes = 32 bits (32 0s or 1s).

But there might be a better way to store may-
be more by declaring 1 variable to store all of them
instead 3/4/5 or more individual variables. B

↳ ARRAYS !!!