

Minute Essay From Last Lecture
One person asked about how to discard invalid user input. Not totally trivial but can be done (e.g., by reading and discarding individual characters until you find end-of-line).
One person asked (later) about problems compiling (actually, linking) programs with sqrt function.
(Many people had not started the homework yet.)

Pointers in C

 C, in contrast to Python and Scala, makes an explicit distinction between things and pointers-to-things. In Python and Scala variables are pointers/references to objects, and you deal with them fairly abstractly. In C, you can have variables that are "things" (integers, floating-point numbers, etc.) and variables that are "pointers to things" (in some ways more like variables in Python and Scala, but very low-level and with fewer safety checks).

Slide 3

• That is, in C, pointers are basically just memory addresses, though declared to point to variables (or data) of a particular type. Example:

int * pointer_to_int; double * pointer_to_double;

Pointers in C — Operators & gets a pointer to something in memory. So for example you could write int x; int * x_ptr = &x; * "dereferences" a pointer. So for example you could change x above by writing *x_ptr = 10; You can also perform arithmetic on pointers (e.g., ++x_ptr) — something not allowed in languages more concerned with safety.

Parameter Passing in C - Review

 In C, all function parameters are passed "by value" — which means that the value provided by the caller is copied to a local storage area in the called function. The called function can change its copy, but changes aren't passed back to the caller.

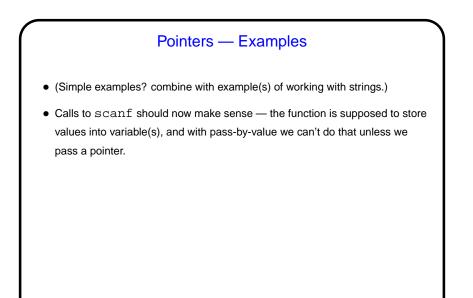
Slide 5

- An apparent exception is arrays no copying is done, and if you pass an array to a function the function can change its contents (as we did in the sort program). Why "apparent exception"? because really what's being passed to the function is not the array but a pointer! so the copying produces a second pointer to the same actual data.
- This is at least simple and consistent, but has annoying limitations ...

Pass By Reference (Sort Of)

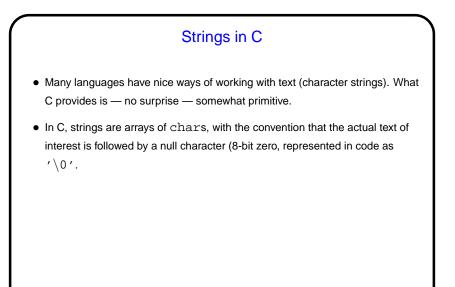
 A significant potential limitation on functions is that a function can only return a single value. Pointers provide a way to get around this restriction: By passing a pointer to something, rather than the thing itself, we can in effect have a function return multiple things.

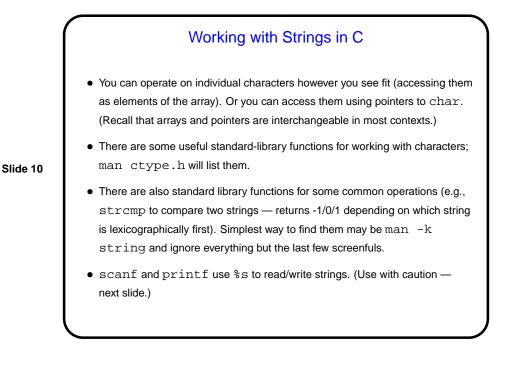
- To make this work, typically you declare the function's parameters as pointers, and pass addresses of variables rather than variables.
- (The "sort of" of the title means that this isn't true pass by reference, as it exists in some other languages such as C++, but it can be used to more or less get the same effect.)

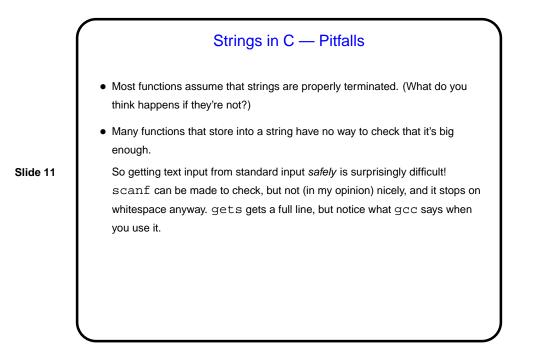


Slide 7

Pointers Versus Arrays In C, pointers and arrays are in some sense(s) equivalent — not identical, but in many contexts interchangeable. This is reflected in the man pages for many functions (e.g., printf — strings are arrays of characters). It also means that when you pass an array to a function, what you're actually passing is a pointer — so the array is not copied.







Examples • (As time permits.) Slide 12

