

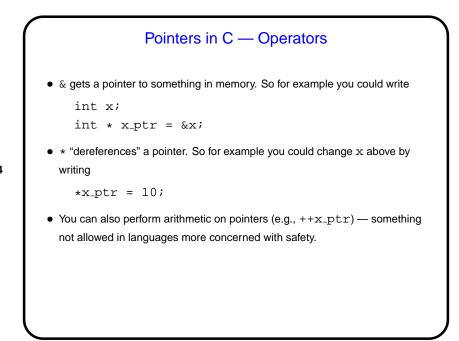
## Pointers in C

 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;



## 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.

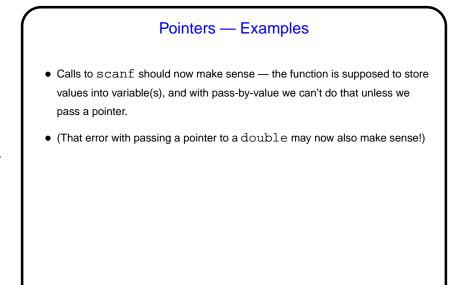
Slide 5

- 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

