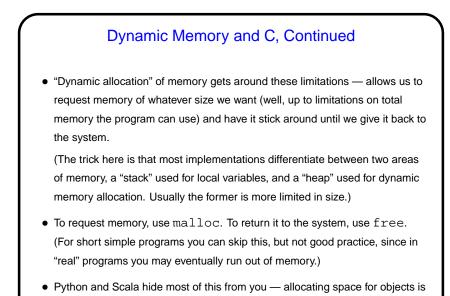


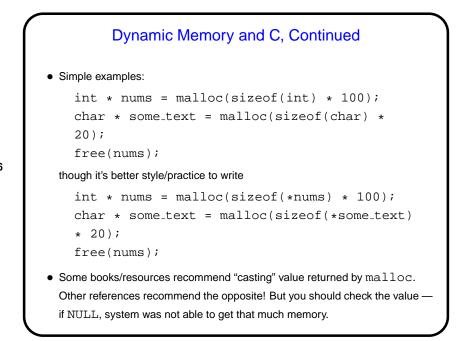
Dynamic Memory and C
With the C89 standard, you had to decide when you compiled the program how big to make things, particularly arrays — a significant limitation.
Variable-length arrays in C99 standard help with that, but don't solve all related problems:
In many implementations, space is obtained for them on "the stack", an area of memory that's limited in size.
You can return a pointer from a function, *but* not to one of the function's local variables (because these local variables cease to exist when you return from the function).

Slide 4

Slide 5



automatic/hidden, and space is reclaimed by automatic garbage collection.



Slide 6

(Example — improved sort program.)

Slide 7

You know from more-abstract languages that there are situations in which it's useful to have method parameters that are essentially code. Some languages make that easy (functions are "first-class objects") and others don't, but almost all of them provide some way to do it, since it's so useful — e.g., providing a "less-than" function for a generic sort. In C, you do this by explicitly passing a pointer to the function.

Function Pointers

Slide 8

