

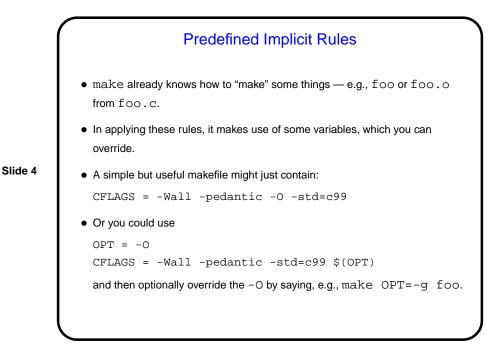
Makefiles

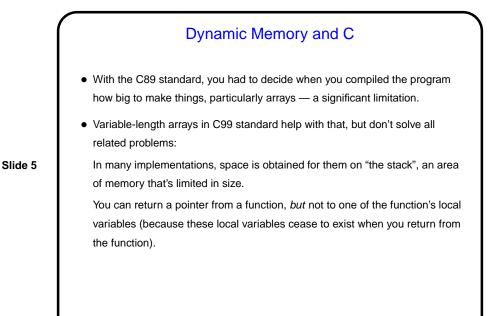
• First step in using make is to set up "makefile" describing how files that make up your program (source, object, executable, etc.) depend on each other and how to update the ones that are generated from others. Normally call this file Makefile or makefile.

Slide 3

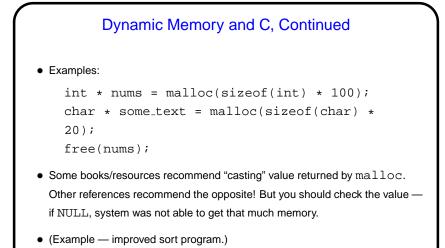
Simple example on sample programs page.

• When you type make, make figures out (based on files' timestamps) which files need to be recreated and how to recreate them.





Dynamic Memory and C, Continued • "Dynamic allocation" of memory gets around these limitations - allows us to request memory of whatever size we want (well, up to limitations on total memory the program can use) and have it stick around until we give it back to the system. (The trick here is that most implementations differentiate between two areas Slide 6 of memory, a "stack" used for local variables, and a "heap" used for dynamic memory allocation. Usually the former is more limited in size.) To request memory, use malloc. To return it to the system, use free. (For short simple programs you can skip this, but not good practice, since in "real" programs you may eventually run out of memory.) • Python and Scala hide most of this from you - allocating space for objects is automatic/hidden, and space is reclaimed by automatic garbage collection. Makes for easier programming but possibly-unpredictable performance.



Slide 7

