Ado-files were introduced in [U] 17 Ado-files.

When a user types ‘gobbledygook’, Stata first asks itself if gobbledygook is one of its built-in com-

mands. **If so**, *the command is executed*. **Otherwise**, it *asks itself if gobbledygook is a defined program*.

If so, the program is executed. Otherwise, Stata looks in various directories for gobbledygook.ado.

**If there is no such file**, *the process ends with the “unrecognized command” error*.

**If Stata finds the file**, *it quietly issues to itself the command ‘run gobbledygook.ado’* (specifying the

path explicitly). **If that runs without error**, *Stata asks itself again if gobbledygook is a defined program*.

**If not**, *Stata issues the “unrecognized command” error*. (Here somebody wrote a bad ado-file.) If the

program is defined, as it should be, Stata executes it.

Thus you can arrange for programs you write to be loaded automatically. For instance, if you were

to create hello.ado containing

After finding and running gobbledygook.ado, Stata calculates the total size of all programs that it

has automatically loaded. **If this exceeds adosize** , *Stata begins discarding the oldest*

*automatically loaded programs* until the total is less than adosize. Oldest here is measured by the

time last used, not the time loaded. This discarding saves memory and does not affect you, because

any program that was automatically loaded could be automatically loaded again if needed.

It does, however, affect performance. Loading the program takes time, and you will again have to

wait if you use one of the previously loaded-and-discarded programs. Increasing adosize reduces

this possibility, but at the cost of memory. The set adosize command allows you to change this

parameter; see [P] sysdir. The default value of adosize is 1,000. A value of 1,000 for adosize

means that up to 1,000 K can be allocated to autoloaded programs. Experimentation has shown that

this is a good number — increasing it does not improve performance much.