

NAME

`rc` – Ratfor compiler

SYNOPSIS

`rc [-c] [-r] [-f] [-v] file ...`

DESCRIPTION

`Rc` invokes the Ratfor preprocessor on a set of Ratfor source files. It accepts three types of arguments:

Arguments whose names end with `‘.r’` are taken to be Ratfor source programs; they are preprocessed into Fortran and compiled. Each subroutine or function `‘name’` is placed on a separate file `name.f`, and its object code is left on `name.o`. The main routine is on `MAIN.f` and `MAIN.o`; block data subprograms go on `blockdata?.f` and `blockdata?.o`. The files resulting from a `‘.r’` file are loaded into a single object file `file.o`, and the intermediate object and Fortran files are removed.

The following flags are interpreted by `rc`. See `ld(I)` for load-time flags.

- `-c` Suppresses the loading phase of the compilation, as does any error in anything.
- `-f` Save Fortran intermediate files. This is primarily for debugging.
- `-r` Ratfor only; don’t try to compile the Fortran. This implies `-f`.
- `-v` Don’t list intermediate file names while compiling.

Arguments whose names end with `‘.f’` are taken to be Fortran source programs; they are compiled in the normal manner. (Only one Fortran routine is allowed in a `‘.f’` file.)

Other arguments are taken to be either loader flag arguments, or Fortran-compatible object programs, typically produced by an earlier `rc` run, or perhaps libraries of Fortran-compatible routines. These programs, together with the results of any compilations specified, are loaded to produce an executable program with name **a.out**.

FILES

<code>ratjunk</code>	temporary
<code>/usr/bin/ratfor</code>	preprocessor
<code>/usr/fort/fc1</code>	Fortran compiler

SEE ALSO

RATFOR – A Preprocessor for a Rational Fortran by B. W. Kernighan.
`fc(I)` for Fortran error messages.

DIAGNOSTICS

Yes, both from `rc` itself and from Fortran.

BUGS

Limit of about 50 arguments, 10 block data files.

`#define` and `#include` lines in `“.f”` files are not processed.