# ECMWF training course – 2016
# Compiling environment – ecgate
# Practical solutions

Have a look at the Makefile in .solutions/Makefile

**EXERCISE 1 (Compiling, linking)**

% ***gfortran –o hello hello.f***

% ***./hello***

**EXERCISE 2 (Creating libraries)**

**Static version:**

% ***gfortran –c prime.f***

% ***ar -rv libmy.a prime.o***

% ***gfortran -o primecheck primecheck.f -L. -lmy***

or

% ***gfortran -o primecheck primecheck.f libmy.a***

***%*** ***./primecheck***

***(%*** ***rm libmy.a; ./primecheck)***

**Dynamic (shared library) version:**

% ***gfortran -c -fPIC prime.f***

% ***gfortran -shared -o libmy.so prime.o***

% ***gfortran -o primecheck primecheck.f -L$PWD -Wl,-rpath,$PWD -lmy***

**EXERCISE 3 (Using libraries)**

**GRIB API:**

% ***gfortran -o grdemo grdemo.f90 $GRIB_API_INCLUDE $GRIB_API_LIB***

% ***./grdemo***

**EMOSLIB:**

% ***gfortran -o bfdemo bfdemo.f90 $EMOSLIB # this is the single precision version***

% ***./bfdemo***

**EXERCISE 4 (Basic Debugging)**

% ***gfortran -o debug debug.f***

% ***./debug***

What can you do to get more information about the problems? (see lecture notes)

% ***gfortran -ffpe-trap=overflow,invalid,zero -o debug debug.f***

% ***./debug***

Possibly …

% ***gfortran -ffpe-trap=overflow,invalid,zero -g -O0 -fbacktrace -fdump-core -o debug debug.f***

% ***./debug***

Generate a core and read with gdb …

% ***ulimit –c 100000***

% ***./debug***

% **gdb –c core.* debug**

***You may want to use gdb internal commands like where, print, etc.***

**EXERCISE 5 (Profiling and 'optimisation')**

% ***gfortran –o primecheck primecheck.f prime.f***

% ***time ./primecheck***

Any trivial way to improve the execution time of this program?

% ***gfortran –O3 -o primecheck primecheck.f prime.f***

% ***time ./primecheck***

To obtain a profile?

% ***gfortran -O0 -g -pg -o primecheck primecheck.f prime.f***

% ***gprof primecheck gmon.out***