The goal of this homework is to complete our `class_rational` and to test out the functionality. When writing these routines you should check out the notes because some of the routines were discussed there. You should write one routine and test it before proceeding to the next. To submit your homework you should submit the two programs `class_rational.f90` containing your modified module and the program to test out the class along with a file containing your output. In your module, list your routines in alphabetical order. Be sure to add some comments if it is not clear from your variable names what you are doing.

Your module should contain the following routines (some of these routines have already been provided):

- a manual constructor and destructor as described in notes
- a function to add two rationals where input is not destroyed and then reduce to simplest form; this is used to overload “+”
- a function to subtract two rationals where input is not destroyed and then reduce to simplest form; this is used to overload “-”
- a function to multiply two rationals where input is not destroyed and then reduce to simplest form; this is used to overload “*”
- a function to divide two rationals where input is not destroyed and then reduce to simplest form; make sure to check that you are not dividing by a zero rational number; this is used to overload “/”
- a recursive function to find greatest common divisor
- a subroutine to reduce a rational to its simplest form
- a routine to print out a rational number in the form $p/q$
- a subprogram to copy a rational into a new rational
- a subprogram to invert a rational and overwrite
- a subprogram to multiply a rational by a scalar and then reduce to simplest form; overload operator “*”
- a subprogram to convert a rational to a real number based upon whether you are using single or double precision
- a subroutine to set a rational number equal to an integer which overloads the assignment “=” as described in notes
- a routine to overload the operator “>” whose output is a logical (see notes)
- a routine to overload the operator “<” whose output is a logical
- a routine to overload the operator “==” whose output is a logical; here we want to determine if two rational numbers are equal so we first reduce them (see notes)
- a routine of your choosing to add some capability