W 2. (a) Declare structure variables named `c1`, `c2`, and `c3`, each having members `real` and `imaginary` of type `double`.

(b) Modify the declaration in part (a) so that `c1`'s members initially have the values 0.0 and 1.0, while `c2`'s members are 1.0 and 0.0 initially. (`c3` is not initialized.)

(c) Write statements that copy the members of `c2` into `c1`. Can this be done in one statement, or does it require two?

(d) Write statements that add the corresponding members of `c1` and `c2`, storing the result in `c3`.

Section 16.2

3. (a) Show how to declare a tag named `complex` for a structure with two members, `real` and `imaginary`, of type `double`.

(b) Use the `complex` tag to declare variables named `c1`, `c2`, and `c3`.

(c) Write a function named `make_complex` that stores its two arguments (both of type `double`) in a `complex` structure, then returns the structure.

(d) Write a function named `add_complex` that adds the corresponding members of its arguments (both `complex` structures), then returns the result (another `complex` structure).

W 4. Repeat Exercise 3, but this time using a *type* named `Complex`.

5. Write the following functions, assuming that the `date` structure contains three members: `month`, `day`, and `year` (all of type `int`).

(a) `int day_of_year(struct date d);`

Returns the day of the year (an integer between 1 and 366) that corresponds to the date `d`.

(b) `int compare_dates(struct date d1, struct date d2);`

Returns –1 if `d1` is an earlier date than `d2`, +1 if `d1` is a later date than `d2`, and 0 if `d1` and `d2` are the same.

6. Write the following function, assuming that the `time` structure contains three members: `hours`, `minutes`, and `seconds` (all of type `int`).

`struct time split_time(long total_seconds);`

`total_seconds` is a time represented as the number of seconds since midnight. The function returns a structure containing the equivalent time in hours (0–23), minutes (0–59), and seconds (0–59).

7. Assume that the `fraction` structure contains two members: `numerator` and `denominator` (both of type `int`). Write functions that perform the following operations on fractions:

(a) Reduce the fraction `f` to lowest terms. *Hint:* To reduce a fraction to lowest terms, first compute the greatest common divisor (GCD) of the numerator and denominator. Then divide both the numerator and denominator by the GCD.
(b) Add the fractions `f1` and `f2`.
(c) Subtract the fraction `f2` from the fraction `f1`.
(d) Multiply the fractions `f1` and `f2`.
(e) Divide the fraction `f1` by the fraction `f2`.

The fractions `f`, `f1`, and `f2` will be arguments of type `struct fraction`; each function will return a value of type `struct fraction`. The fractions returned by the functions in parts (b)–(e) should be reduced to lowest terms. *Hint:* You may use the function from part (a) to help write the functions in parts (b)–(e).