Specifying Software through Contracts

In the Western calendar, a year is divided into 12 months, numbered from 1 to 12, and months into days, numbered starting from 1. The 1st, 3rd, 5th, 7th, 8th, 10th, and 12th months have 31 days each, while the 4th, 6th, 9th, and 11th months have 30 days each. The 2nd month usually has 28 days, but it has 29 days in leap years. A year is a leap year if and only if either it can be divided by 4 but not by 100, or it can be divided by 400. For example, the years 2000 and 2004 are leap years, but the years 2011 and 2100 are not.

The following deferred class `DATE` defines a simple interface for dates in the Western calendar. Please fill in the missing contracts (preconditions, postconditions, and class invariants) of the class; the contracts must reflect all the details given in the informal specification above. They also must ensure that the following client procedure always executes without contract violations:

```plaintext
client (d: DATE)
    require
        d /= Void
    local
        i: INTEGER
    do
        d.set (2011, 8, 22)
        check not d.is_leap (2011) end
        check d.is_leap (2012) end
        from i := 1 until i > 366 loop
            d.proceed
            i := i + 1
        end
        check d.year = 2012 and d.month = 8 and d.day = 22 end
    end
```

Please note that the number of dotted lines is not indicative of the number of missing assertions (contract elements).

defered class

`DATE`

feature -- Access

`year: NATURAL`
    -- Year of the date.
    deferred
end

`month: NATURAL`
    -- Month of the date.
    deferred
end

`day: NATURAL`
    -- Day of the date.
    deferred
end

feature -- Status set
set \( (y, m, d: NATURAL) \)
--- Set 'year', 'month', and 'day' using 'y', 'm', and 'd', respectively.

require

deferred

ensure

end

feature  --- Auxiliary queries

\textit{is\_long\_month} \( (m: NATURAL): BOOLEAN \)
--- Does month 'm' have 31 days?

deferred

ensure

end

\textit{is\_short\_month} \( (m: NATURAL): BOOLEAN \)
--- Does month 'm' have 30 days?

deferred

ensure
end

days_in_month (y, m: NATURAL): NATURAL
    -- Number of days in month 'm' of year 'y'.
require
    .................................................................
    .................................................................
defered
ensure
    .................................................................
    .................................................................
    .................................................................
    .................................................................
    .................................................................
    .................................................................
    .................................................................
    .................................................................
    .................................................................
    .................................................................
end

is_leap (y: NATURAL): BOOLEAN
    -- Is 'y' a leap year?
defered
ensure
    .................................................................
    .................................................................
end

feature  --  Basic operation

    proceed
    --  Proceed to the next date.
    --  For example, if ‘Current’ represents the date Dec. 31, 2011 before the call,
    --  then it represents Jan. 1, 2012 afterwards.
defered
ensure
end

--- Other features omitted for brevity.

invariant

end
deferred class
    DATE

feature -- Access

    year: NATURAL
        -- Year of the date.
        deferred
        end

    month: NATURAL
        -- Month of the date.
        deferred
        end

    day: NATURAL
        -- Day of the date.
        deferred
        end

feature -- Status set

    set (y, m, d: NATURAL)
        -- Set ‘year’, ‘month’, and ‘day’ using ‘y’, ‘m’, and ‘d’, respectively.
        require
            valid_month: 1 <= m and m <= 12
            valid_day: 1 <= d and d <= days_in_month (y, m)
        deferred
        ensure
            year_set: year = y
            month_set: month = m
            day_set: day = d
        end

feature -- Auxiliary queries

    is_long_month (m: NATURAL): BOOLEAN
        -- Does month ‘m’ have 31 days?
        deferred
        ensure
            definition: Result = (m = 1 or m = 3 or m = 5 or m = 7 or m = 8 or m = 10
                                or m = 12)
        end

    is_short_month (m: NATURAL): BOOLEAN
        -- Does month ‘m’ have 30 days?
        deferred
        ensure
            definition: Result = (m = 4 or m = 6 or m = 9 or m = 11)
        end
days\_in\_month (y, m: NATURAL): NATURAL
   -- Number of days in month ‘m’ of year ‘y’.
   require
      valid\_month: 1 <= m and then m <= 12
   deferred
   ensure
      long\_months: is\_long\_month (m) implies Result = 31
      short\_months: is\_short\_month (m) implies Result = 30
      feb\_in\_nonleap: (m = 2 and not is\_leap(y)) implies Result = 28
      feb\_in\_leap: (m = 2 and is\_leap(y)) implies Result = 29
   end

is\_leap (y: NATURAL): BOOLEAN
   -- Is ‘y’ a leap year?
   deferred
   ensure
      definition: Result = (y \mod 4 = 0 and y \mod 100 /= 0) or else (y \mod 400 = 0)
   end

feature -- Basic operation

   proceed
      -- Proceed to the next date.
      -- For example, if ‘Current’ represents the date Dec. 31, 2011 before the call,
      -- then it represents Jan. 1, 2012 afterwards.
      deferred
      ensure
         within\_month: old day < days\_in\_month(old year, old month) implies
            (old year = year and old month = month and day = old day + 1)
         to\_next\_month: (old day = days\_in\_month(old year, old month) and old month <
            12) implies
            (old year = year and month = old month + 1 and day = 1)
         to\_next\_year: (old day = days\_in\_month(old year, old month) and old month =
            12) implies
            (year = old year + 1 and month = 1 and day = 1)
      end

      -- Other features omitted for brevity.

invariant

   valid\_month: 1 <= month and then month <= 12
   valid\_day: 1 <= day and then day <= days\_in\_month (year, month)

end