CIS 343 – Structure of Programming Languages
Winter 2012

Programming Assignment #5
Programming in Ruby

Due Date: Monday, April 9, 2012

Problem Specification

Write a class in Ruby named `SimpleDate` that represents a calendar date and contains the methods to manipulate dates. Complete the following methods in the `SimpleDate.rb` file provided to you (see the Assignments folder on the Blackboard).

- `initialize(month, day, year)`
- `to_s()`
- `each()`
- `<=>(other)`
- `dayOfWeek()`
- `daysInYear()`
- `daysInYear(year)` `// class method`
- `daysInMonth(month, year)` `// class method`
- `leapYear?()`
- `leapYear?(year)` `// class method`
- `validDate?(month, day, year)` `// class method`
- `ordinalDate()`
- `nextDate()`
- `prevDate()`
- `daysAgo(n)`
- `daysFromNow(n)`

Taking possible dependencies (such as a public method using a helper method) between methods into consideration, you may find implementing the methods in the order suggested below helpful:

- `to_s()`
- `leapYear?(year)`
- `daysInMonth(month, year)`
- `validDate?()`
- `initialize(month, day, year)`
- `leapYear?()`

- `daysInYear(year)`
- `daysInYear()`
- `each()`
- `<=>(other)`
- `ordinalDate()`
- `dayOfWeek()`
- `nextDate()`
- `prevDate()`
- `daysAgo(n)`
- `daysFromNow(n)`

Testing Your Program

The supplied `SimpleDateTest.rb` file contains unit tests that test your implementation of `SimpleDate` class. This file will be available on Blackboard within the next few days.

Run the `SimpleDateTest.rb` file to test your code.

Instructions/Deliverables

1. Upload only the completed **`SimpleDate.rb`** file on Blackboard. DO NOT upload the SimpleDateTest.rb file.
    - I will use the submission date/time on Blackboard as your official submission date/time.
    - It is your responsibility to make sure the submission on Blackboard went through successfully.
2. Because of possible version issues between Ruby 1.8 and Ruby 1.9, please clearly indicate which version of Ruby is used to implement and test your program.