Project Details

In this project, you will need to solve a simple problem in each of four programming languages: Ada, C#, Clojure and Ruby. Each of your programs go from year 2000 through year 2100 and show all the years that **does not** have any month with five weekends (Friday, Saturday and Sunday) within the range. For example, July 2016 has five Fridays, five Saturdays and five Sundays, and therefore, year 2016 should not be in the result. On the other hand, since there does not exist a month with five Fridays, five Saturdays and five Sundays in year 2012, your result list should include 2012.

Division of labor and writeup

Work in teams of two or individually. Note that if you split up the language, you and your team member will share the same grade. This means that even though you may not use one of the four languages, you are still responsible for the provided solution. That is, you will need to really understand your partner's code.

In your write-up (part of README file), you must compare and contrast the programming experience in the different languages (all four of them). What was easy? What was hard? Are there noticeable differences in speed (for comparable algorithms)? What do you like/dislike?

README files

Your README file for each project should contain your name, the project name, and all information you think is necessary for a typical user to run your program and for a software maintainer to understand and evaluate your code. Much of this information may be included on the original project description and you are responsible for filling in the parts not provided for you. More specifically:

- Information for the prospective user includes details on calling conventions, input and output data formats, limitations, bugs, and special features.
- Explain both the negative aspects of your program (limitations, known bugs) and the positive aspects (extensions, special features). Note that if you do not include the former, I will assume you did not realize the limitations were there.
- For the person who has to understand the insides of your code, you may need to describe your choice of modularization (abstractions), data structures, and algorithms. Be sure to explain anything you did that is likely to be different from what other students may have done, and justify any design decisions for which the rationale isn't immediately clear.

Your write-up should be coherently written, using full sentences and paragraphs. This write-up need not necessarily be long, but it is important. All write-ups must be in plain text (README.txt) or Adobe PDF (README.pdf) format. Other formats (e.g., Word or postscript) will not be accepted.

Grading scale

The project will be graded based on your source code and the README file:

- Code (80%): for each language,

  - Correctness, completeness and efficiency (15%): Your code should implement everything that was required in the assignment. It should produce the right output given normal, expected inputs, and some sort of reasonable response to unexpected inputs. Unless otherwise instructed, and as long as it does not severely compromise programming style, you are expected to use the most efficient data structures and algorithms.
  - Programming style (including internal documentation and program organization) (5%): Your code should have appropriate abstractions, data structures, algorithms, and variable names; declarations for all constants; and a judicious number of helpful comments. You should think of programming as explaining to the readers of your programs what you want the computer to do.
- README file (20%)
  - Completeness (15%): Your write-up should include all the items discussed in the README files.
  - Readability (5%): Your write-up should be well organized, clear, and concisely presented.

Resources

I will not be devoting lecture time to how to use these languages. You'll need to find online tutorials or other resources, and teach yourself. Here are some decent starting points:

- Ada
  - `http://www.adahome.com/Ammo/Cplpl2Ada.html`
  - `http://www.adahome.com/Tutorials/Lovelace/tutors.htm`
- C#
  - `www.hitmill.com/programming/dotNET/csharp.html`
  - `www.ecma-international.org/publications/standards/Ecma-334.htm`
  - `msdn.microsoft.com/`
- Clojure
  - `http://www.braveclojure.com/introduction/`
  - `http://www.tutorialspoint.com/clojure/index.htm`
- Ruby
  - `http://www.tutorialspoint.com/ruby/`
  - `https://www.ruby-lang.org/en/documentation/quickstart/`
  - `http://rubylearning.com/satishtalim/tutorial.html`

What to turn in:

Zip your source files and README files (one for each solution of these four languages) and upload your zip file to Canvas under category Project1 by the deadline.