Sushmith Ramesh

Project #1 – Check for months with 5 full weekends – Ruby

How to run program:

Store the ruby file in some directory. Open the command prompt and go to the directory. Then enter "ruby file_name.rb" to run the program (install ruby compiler if you haven't). The program will ask for a value (any year after 2000) to be entered, after which it will display all the years from 2000 to the year you entered that do not have any months with 5 full weekends (Friday, Saturday, and Sunday).

About the Language and Program:

Ruby for me in particular was not too difficult to understand as I have written a Ruby program before (search and sort). Additionally I have worked with Python, which is quite similar to Ruby and other languages such as Java and C++ that have similar ideas and structures. My initial program followed the same algorithm as the FullWeekend.rb file that was provided to us, however, I used a "for" loop and nested if statements to go through each month and count the number of Fridays, Saturdays, and Sundays. When I saw the line, "result1 = (start_month..end_month).to_a.select {|k1| id_Fridays.include?(k1.wday)}" in the FullWeekends.rb file, I broke down each methods and realized that it preformed the same function as my for loop and nested if statements. I decided to adopt this line into my code as it helped me realize some of the more powerful methods and functions available in Ruby as everything was done in one line. While my code is very similar to the FullWeekends.rb file, there are noticeable differences, such as the acceptance of user input and outputting the proper count (FullWeekends.rb outputs all the years with 5 full weekends). Additionally, FullWeekends.rb has a small error with the iterations in that it does not end up checking all the months of the final year indicated by the user. This is fixed in my code where the iterations occur until December of the year indicated by the user. Bugs that may occur with the code will probably exist around the user input as I do not completely check and handle potential errors.

Overall, I found working with Ruby enjoyable as I was able to practice my previous skills as well as learn some new methods and functions in the language.