LSRC "Ruby Craftsmanship" Training

*Practical Test-Driven Development: Lab Task A*

At FooSoft, your former coworker Paul has decided to leave the software enterprise for good and start his life anew as a contestant on "Deadliest Catch". Because you're not particularly adept at crab fishing, and prefer the hot summer weather of Austin, TX, you and your team will need to pick up where Paul left off.

Paul liked to work alone, and more or less kept to himself, so the details of his projects tend to be a bit sketchy. However, after digging through some old design documents, your team figured out the basic outline of what needs to be done.

**The Task:**

Paul was working on several components for a payroll system, and had made some progress on a periodic report of employee hours.

The sketch below gives a rough idea of what the final report should look like:

---

**Time Report ( 2009.08.17 – 2009.08.30 )**

| ***Employee*** | ***Week 1*** | ***Week 2*** | ***Totals*** |
|---|---|---|---|
| Joe Frasier | 13 | 27 | 30 |
| John Adams | 20 | 21 | 41 |
| Sarah Miller | 14 | 28 | 42 |
| Ralph Jackson | 18 | 39 | 57 |
| Albert Frost | 49 | 10 | 59 |

---

While not exactly pretty, it gives you a sense of the overall structure of the report and what data it needs to contain. FooSoft (amazingly) tends not to care about the particular look and feel of their reports, so long as they are provided in one of the following file formats: HTML, PDF, CSV, or plain text. That means your team is free to pick whatever format you're most comfortable with.

**The Data**

Before his departure, Paul had just finished up the first stages of this project, an exporter that takes data from the current payroll system and converts it to YAML. While you don't happen to have the source for this utility handy, you do have a data file produced by it ( *payroll_data.yaml* ). Unfortunately, it is without documentation, so that means you're team will need to do a bit of exploration before they can begin coding.

**Design Notes**

Typically, things are prone to change rapidly at FooSoft. For this reason, it is best to decompose the problem into 2 separate components. The first is the `DataProcessor`, which extracts and transforms the YAML data into something closer to what is needed for the report. The second is the `ReportGenerator`, which takes the processed data and renders it in a particular format. The two should be wrapped together under a common interface, allowing the report to be generated using an interface similar to the one below:

```
TimeReport.new(data, period_start_date).render(filename)
```

The main idea is that the business logic should be separated from the formatting, so that a change to one component doesn't necessarily require a change in the other.

**Procedures and Methodology**

FooSoft is encouraging teams to begin using helpful methodologies such as pair programming, test driven development, and other similar techniques. For this particular project, you'll be writing tests before you write code.

To help make sure this happens, you will be using a technique called Ping Pong TDD. One of your team members will write a small, simple test. Another will get that test to pass, and then follow up by writing a new failing test. Control cycles back and forth between team members until the job is finished.

Your team is free to choose whatever testing framework they like, though vanilla Test::Unit or RSpec tend to be a reasonable lowest common denominator.

You may also choose whether to work together as a whole group, or split up into sub-teams to tackle different parts of the problem. Just be sure that no one is left on their own, even if they're up to the challenge.

**Communications**

It is up to you to process this data and produce the final report. While your team is responsible for implementation, two other FooSoft developers, Brad and Greg, are available for strategic advice and can provide answers to questions about business logic. Since they were pretty much the only people Paul talked to, they will be your best bet for help on this project.

*When you have completed this task, be sure to let Greg or Brad know so that they can review your work, and if necessary, recommend changes.*