_h

Sample Report for Simple ML Example

Alfred Murabito

February 8, 2020

Abstract

The project brings together elements of functional programming in ML and documentation ins LaTeX. The purpose of this project is to lay the groundwork for credibility: results that are thoroughly documented and easily reproducible by independent third parties. We establish the documentation and programming infrastructure where each chapter documents a problem or exercise. Within each chapter are sections stating or showing:

- Problem statement
- Relevant code
- Test results

For each problem or exercise-oriented chapter in the main body of the report is a corresponding chapter in the Appendix containing the source code in ML. This source code is not pasted into the Appendix. Rather, it is input directly from the source code file itself. This means changes in source code are easily captured in the report by recompiling the report in LaTeX.

We introduce the use of style files and packages. Specifically, we use:

- a style file for the course, *634format.sty*,
- the *listings* package for displaying and inputting ML source code, and
- HOL style files and commands to display interactive ML/HOL sessions.

Finally, we show how to:

- easily generate a table of contents for the report, and
- refer to chapter and section labels in our report.

There are numerous LaTeX tutorials on the web, for example, `https://www.latex-tutorial.com`, is very accessible for beginners.

Acknowledgments: We gratefully acknowledge the hard work, trust, and dedication of our past students in the Syracuse University Cyber Engineering Semester and the Air Force Research Laboratory's Advanced Course (ACE) in Engineering Cybersecurity Boot Camp. They bridged dreams and reality.

Contents

Chapter 1

Executive Summary

All requirements for this project are satisfied. Specifically,

Report Contents
Our report has the following content:

- Chapter 1: Executive Summary
- Chapter 2: Report Body
  - Section 2.1: Problem statement
  - Section 2.2: Relevant code
  - Section 2.3: Test results
- Chapter A: Source Code for Sample Exercise

Reproducibility in ML and LaTeX
Our ML and LaTeX source files compile with no errors.

Chapter 2

Body

2.1 Problem Statement

Generating Law firm data to Mr. How on the total hours of experience from all of the members of the law firm. The total hours and male female splits are found for all Associate members of the firm.

2.2 Relevant Code

2.2.1 Employee Firm Data

First piece of code we need is a definition in ML of the table containing all of the relevant employee information.

The following code is from *table.sml*

```
(* Table of relevant information in the form of
a (string * string * string * int) list


(*****************************************************************************)
(*    Table of relevant information in the form                               *)
(*    a (string * string * string * int) list                                 *)
(*                                                                            *)
(*****************************************************************************)
val table =
[("Nancy", "F", "Associate", 7),
("Peter", "M", "Associate", 5),
("Paul", "M", "Sr. Partner", 23),
("Robert", "M", "Jr. Partner", 11),
("Ralph", "M", "Associate", 4),
("Rhonda", "F", "Associate", 11),
("Sam", "M", "Associate", 5),
("Sara", "F", "Associate", 9),
("Alice", "F", "Sr. Partner", 27),
("Bob", "M", "Jr. Partner", 14),
("Eve", "F", "Associate", 9),
("John", "M", "Jr. Partner", 17),
("Gary", "M", "Associate", 6),
("Adam", "M", "Associate", 8),
("Carl", "M", "Associate", 2),
("Douglas", "M", "Jr. Partner", 14),
("Sally", "F", "Jr. Partner", 19),
("Joseph", "M", "Sr. Partner", 33),
("Mary", "F", "Jr. Partner", 18),
("Esther", "F", "Associate", 8),
("Frank", "M", "Associate", 7),
```

```
("Jane", "F", "Associate", 3),
("George", "M", "Jr. Partner", 15),
("Harry", "M", "Associate", 1),
("Robin", "F", "Associate", 8),
("David", "M", "Jr. Partner", 16),
("Jan", "F", "Associate", 6),
("Jack", "M", "Associate", 4),
("Ken", "M", "Associate", 9),
("Kathy", "F", "Associate", 1),
("Larry", "M", "Associate", 7),
("Mike", "M", "Jr. Partner", 12),
("Ted", "M", "Jr. Partner", 18),
("Terry", "M", "Sr. Partner", 30),
("Victor", "M", "Associate", 3),
("Vera", "F", "Sr. Partner", 28),
("Wilma", "F", "Associate", 7),
("Zack", "M", "Associate", 1),
("Debra", "F", "Associate", 12),
("Bonnie", "F", "Associate", 5),
("Dan", "M", "Jr. Partner", 17),
("Meredith", "M", "Jr. Partner", 17),
("Randal", "M", "Associate", 9),
("Jake", "M", "Jr. Partner", 12),
("Karen", "F", "Jr. Partner", 9)];
```

2.2.2 Calculate Total Hours

```
(* Calculate the total hours in experience of  *)
(* a given seniority (status)                  *)

fun calc_h (status:string) [] = 0
  | calc_h (status) (head::table:(string*string*string*int)list) =
  let
    val (name, gender, j_status, exp) = head
    val match:bool = (status = j_status)
  in
    if match then (2080*exp)+(calc_h status table) else (calc_h status table)
  end;

(* For calculating total hours for all Associates, use below *)
calc_h ``Associate'' table;
```

2.2.3 Get Male/Female Split

```
(* Calculate the male/female split *)

fun get_male_female(status:string) [] = (0, 0)
  | get_male_female(status:string) (head::table) =
  let
```

```
    (* Use pattern matching to get elmenets of list entry *)
    val (name, gender, j_status, exp) = head
    val f_match:bool = ((status = j_status) andalso (gender = "F"))
    val m_match:bool = ((status = j_status) andalso (gender = "M"))
    val (p_male, p_fem) = get_male_female status table
  in
    ((if m_match then 1 else 0)+p_male,(if f_match then 1 else 0)+p_fem)
  end;

(* To get the male female split for Associates, use below *)
get_male_female ``Associate'' table;
```

2.3 Test Results

2.3.1 Test results for Employee Data

2.3.2 Test results for calc_h

2.3.3 Tests results for get_male_female

Appendix A

Source Code for Sample Exercise

The following code is from *simple.sml*

```
(*****************************************************************************)
(* Author: Shiu-Kai Chin                                                     *)
(* Date: 20 January 2017                                                     *)
(* email: skchin@syr.edu                                                     *)
(*****************************************************************************)
fun plus x y = x + y;

fun times x y = x*y;



(*****************************************************************************)
(* Test Cases Specified in the requirements                                  *)
(*****************************************************************************)
plus 0 0;
plus 0 1;
plus 0 2;
plus 0 3;

plus 0 0;
plus 1 0;
plus 2 0;
plus 3 0;

plus 1 0;
plus 1 1;
plus 1 2;
plus 1 3;

plus 0 1;
plus 1 1;
plus 2 1;
plus 3 1;

(*****************************************************************************)
(* Test Cases Specified in the requirements                                  *)
(*****************************************************************************)
times 1 0;
times 1 1;
times 1 2;
times 1 3;
```

```
times 0 1;
times 1 1;
times 2 1;
times 3 1;

times 4 0;
times 4 1;
times 4 2;
times 4 3;

times 0 4;
times 1 4;
times 2 4;
times 3 4;
```