Problem Set: Higher-order functional programming

Stuart M. Shieber

February 5, 2023

This assignment focuses on programming in the functional programming paradigm, with special attention to the idiomatic use of higher-order functions like `map`, `fold`, and `filter`. In doing so, you will exercise important features of functional languages, such as recursion, pattern matching, and list processing.

Mapping, folding, and filtering are important techniques in functional languages that allow the programmer to abstract out the details of traversing and manipulating lists. Each can be used to accomplish a great deal with very little code. In this problem set, you will create a number of functions using the higher-order functions `map`, `filter`, and the fold functions `fold_right` and `fold_left`. In OCaml, these functions are available as `List.map`, `List.filter`, `List.fold_right`, and `List.fold_left` in the `List` module.

The file `mapfold.ml` contains starter code for a set of functions that operate on lists. For each one, you are to provide the implementation of that function using the higher-order (mapping, folding, filtering) functions directly. The point is to use the higher-order functional programming paradigm idiomatically. The problems will be graded accordingly: a solution, even a working one, that does not use the higher-order functional paradigm, deploying these higher-order functions properly, will receive little or no credit. For instance, solutions that require you to change the `let` to a `let rec` indicate that you probably haven't assimilated the higher-order functional paradigm. However, you should feel free to use functions you've defined earlier in the problem set to implement others later where appropriate.

Remember to provide unit tests for all of the functions in `mapfold.ml` in a file `mapfold_tests.ml`. We have included an example of its use in the starter code for `mapfold_tests.ml`. You should provide at least one test per code path for every function that you write on this problem set.

Version information: Commit b72ee47 from 2023-02-05 by Stuart Shieber. CI build of I_TAG ().