CIS 705 — Programming Languages — Spring 2007

Assignment 2

Due by 1:30 p.m. on Monday, February 26

A Multi-word Anagram Finder

The goal of this assignment is to write an OCaml program for finding multi-word anagrams. Suppose *xss* is a value of type `'a list list` and *ys* is a value of type `'a list`. We say that a value *zss* of type `'a list list` is a *multi-word anagram of ys relative to xss* iff

- the concatenation of the elements of *zss* is a permutation of *ys*; and
- each element of *zss* is a nonempty element of *xss*.

For example `[[2;1];[1;3;2]]` and `[[1;3;2;2];[1]]` are multi-word anagrams of `[1;2;3;1;2]` relative to `[[1];[2;1];[1;3;2];[1;3;2;2]]`.

Your goal is to write an OCaml function

```
val anagrams : 'a list list -> 'a list -> 'a list list list
```

such that `anagrams` *xss ys* is the list of all multi-word anagrams of *ys* relative to *xss*. You may list the multi-word anagrams in any order, but each multi-word anagram should only appear once.

Assessment

Your program will be assessed according to three criteria:

- **Correctness**. Most importantly, your program must be correct. You should test it carefully, but you should also try to prove to yourself that it is correct. (You don't have to submit evidence of testing, or to write your proof down.)
- **Style**. You should format your program in a way that makes its structure clear. You should choose meaningful names for functions, and choose other identifiers with care. Document your functions by abstractly explaining their input/output behavior. (For a given function, say that if we know that its input has some property, that its output will have some other property).
- **Efficiency**. Make your program efficient, but don't sacrifice correctness or style. Assume that it will be common to call `anagrams` with a value *xss* of type `'a list list`, give a name to the resulting function *f* of type `'a list -> 'a list list list`, and then call *f* repeatedly with different arguments. Consequently, as much work as possible should be carried out *before* this *f* is returned.

Submission

Your program should begin with a comment containing your name. Submit your program by emailing it to me. I will acknowledge receiving it. Make sure that you retain an electronic copy of your program.