Due April 10, 5:00 p.m.

**Problem 1**

Write a Scheme function called `make-polite` that takes as its only argument a list of symbols representing a polite way of introducing a command. `make-polite` should return as its output a function. The function returned by `make-polite` should take a list of symbols representing a command as its input and it should return a list of symbols representing a polite way of giving the command. E.g.,

```scheme
> (define say-please (make-polite '(please)))
> (say-please '(don_t eat the daisies))
(please don_t eat the daisies)
> (map say-please '((don_t smoke) (don_t run)))
((please don_t smoke) (please don_t run))

> (define be-timid (make-polite '(if it_s not too much trouble)))
> (be-timid '(don_t eat the daisies))
(if it_s not too much trouble don_t eat the daisies)
```

**Problem 2**

Write a procedure called `safe-fn` that takes two procedures, `fn` and `test` as arguments and returns a procedure. When the procedure returned by `safe-fn` is applied to an argument, `arg`, it returns the symbol `error` whenever the value of `(test arg)` is `#f`. Otherwise, it returns the value of `(fn arg)`. E.g.,

```scheme
> (define safe-sqrt (safe-fn sqrt (lambda (n) (>= n 0))))
> (safe-sqrt 4)
2
> (safe-sqrt -4)
error
```

**Submitting**

Don't forget to submit your work using the `submit101` command!

```
submit101 g-asmt06 asmt06
```

(If the name of your directory is different from `asmt06`, change `asmt06` to whatever the name of your directory is.)