CSE 341, Modules and References
April 28, 2014

Programmer 1 : ______________________________ Programmer 2 : ______________________________

Programmer 3 : ______________________________ Programmer 4 : ______________________________

Assume this implementation of `fold`:

```
(* fold : (a -> b -> b) -> b -> a list -> b *)
fun fold f base [] = base
  | fold f base (h::t) = f h (fold f base t)
```

(A) With your group, use `fold` to write a function `split` of type:

$$\alpha\ \texttt{list} \rightarrow \alpha\ \texttt{list} * \alpha\ \texttt{list}$$

such that if `split l = (a, b)`, then half of l's elements are in `a` and the other half in `b` (don't worry about order). *Hint:* your accumulator for fold will need to carry extra information.

(B) With your group, use references to implement `fact` and `fib`:

(C) Which way does our version of `fold` associate? Use `fold` to implement another version of fold that associates the other way.

(D) Implement a module satisfying this signature:

```
signature Ref = sig
  type t
  val create : int -> t
  val set : t -> int -> unit
  val get : t -> int
end
```