6. This question is about defining and using *folding from the left* in functional programming.

   (a) Define the Haskell function,

   ```
   foldl :: (a -> b -> a) -> a -> [b] -> a
   ```

   for *folding from the left.* [5]

   (b) Using `foldl` define the Haskell function `length :: [a] -> Int` which returns the length of a list. For example, `length "abc"` evaluates to `3` . [5]

   (c) Using `foldl` define the Haskell function `reverse :: [a] -> [a]` which reverses the order of the items in a finite list. For example, `reverse "abc"` evaluates to `"cba"` . [5]

   (d) Using `foldl` define the Haskell function `map :: (a -> b) ->[a] -> [b]` such that `map f l` applies `f` to each item in `l` . For example, `map (* 2) [1,3,5]` evaluates to `[2,6,10]` . [5]