Introduction to Functional Programming
Lent 2000
Suggested Exercises 3

1. **Sequences**
Define a function that takes a sequence (a lazy list) of integers, and produces a sequence containing the sums of adjacent pairs. So, the list $[x_1, x_2, x_3, x_4, \ldots]$ would be mapped to $[x_1 + x_2, x_3 + x_4, \ldots]$.

2. **Lazier Lists**
The following datatype provides an alternative implementation of lazy lists to the one given in the lectures. It avoids the unnecessary computation of the first element, when it is not required.

```
datatype 'a seq = Nil
                | Cons of unit - 'a * 'a seq;
```

Modify the functions `hdq`, `tlq`, `from` and `take` to work with this datatype.

Repeat Question 1 for this datatype.

3. **Structural Induction**
Prove the following identities by induction on lists.

$$(\texttt{map}\ f) \circ \texttt{reverse} = \texttt{reverse} \circ (\texttt{map}\ f)$$

$$\texttt{foldr}(\texttt{op}\ ::) l_1 l_2 = l_2 @ l_1$$

Write a function `mirror` that produces the mirror image of a binary tree, i.e. the tree obtained by interchanging (recursively) the left and right child of every node.
Prove, by induction on trees, that $\texttt{reflect}(\texttt{reflect} t) = t$.