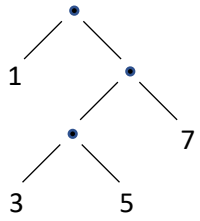
Exercise: Tree

Recursive custom types

Consider binary trees with information only at the leaves, for example:



.

Such trees can be represented with the type

```
type Tree a = Leaf a | Node (Tree a) (Tree a)
```

,

where the example above would be represented with the value

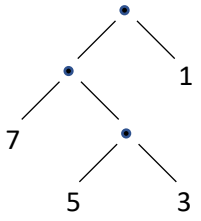
```
Node (Leaf 1) (Node (Node (Leaf 3) (Leaf 5)) (Leaf 7))
```

.

Now implement the following two functions:

- Function `frontier` with signature frontier: `Tree a -> List a` , which returns the list of values at the leaves of a tree, collected from left to right; for the example tree above, it would return the list `[1, 3, 5, 7]` .

- Function `mirror` with signature mirror: `Tree a -> Tree a` , which returns the mirror image of a tree; for the example tree above, it would return the tree



.

The provided Elm project contains a module `Tree` for the implementation of the type and the functions described above, plus a module `Tests` with some corresponding unit tests.

End of exercise