1 Exponentiation by squaring

Problem 1. *Input*:
Given associative operator $\cdot$, element a and positive integer n.
Output:
Find a^n, where $a^n = a \cdot a^{n-1}$.

The general method to solve the problem is exponentiation by squaring. It is originally used for integer exponentiation, but any associate operator can be used in it's place. Here is a theorem stated in algebraic flavor.

Theorem 1.1. *For any semigroup $(S, \cdot)$, $x \in S$ and $n \in \mathbb{N}$, x^n can be computed with $O(\log n)$ applications of $\cdot$.*

Proof.
```
import Data.Digits
exponentiationBySquaring :: Integral a => (b -> b -> b) -> b -> a -> b
exponentiationBySquaring op a n = foldr1 op $ snd . unzip $ filter (\(x, _). x /= 0) (zip binary twoPow)
  where twoPow = a : zipWith op twoPow twoPow
        binary = digitsRev 2 n
```

One can analyze the number of times the operator is used. The *twoPow* is the infinite list $[a, a^2, \ldots, a^{2^i}, \ldots$. It takes k operations to generate the first $k+1$ elements. At most k additional operations are required to combine the result with the operator. Therefore the operator is used $O(\log n)$ times. $\square$