1. Defina três propriedades QuickCheck para testar a implementação de árvore de busca. É necessário definir um gerador.

   Exemplos (Propriedades de árvores binárias)

   - Uma árvore binária com N nós internos tem, no máximo, N+1 folhas
   - O número **máximo** de nós de uma árvore binária de altura h é $2^{h+1} - 1$

2. Defina cinco propriedades QuickCheck para para o tipo Set que devem refletir as propriedades matemáticas de conjunto (Set - Wikipedia) (Set - Wikipedia)

3. Defina uma propriedade QuickCheck para as funções de busca em grafos

4. Dado o seguinte tipo de expressões

```
data Expr a = Var a | Val Int | Add (Expr a) (Expr a)
            deriving (Show)
```

   que contém variáveis de algum tipo a , mostre que como estabelecer que este tipo é instância das classes Functor, Applicative e Monad. Com um exemplo, explique o que o operador >>= para este tipo faz

5. Algumas vezes é mais simples definir instâncias de Functor e Applicative em termos da instância de mônada, com base no fato de que a ordem das declarações não importa em Haskell. Complete as declarações abaixo usando a notação do.

```
instance Functor ST where
 -- fmap :: (a -> b) -> ST a-> ST b
 fmap g st = do ...

instance Applicative ST where
 -- pure :: a -> ST a
 pure x = S (\s -> (x,s))

 -- (<*>) :: ST (a -> b) -> ST a -> ST b
 stf <*> stx = do ...

instance Monad ST where
 -- (>>=) :: ST a -> (a -> ST b)  -> ST b
 st >>= f = S (\s ->
   let (x,s') = app st s in app (f x) s' )
```