Introducing QuickCheck

```
-- In QuickCheck, the Arbitrary class associates a random value generator to a type
class Arbitrary t where
  arbitrary :: Gen t -- Gen t is a way of generating random values of type t


-- Let's define Arbitrary instances of our CRDTs
instance arbitraryGSet :: (Arbitrary t, Hashable t) ⇒ Arbitrary (GSet t) where
  arbitrary = GSet <<< Set.fromArray <$> arbitrary

instance gCounterArbitrary :: Arbitrary GCounter where
  arbitrary = GCounter <<< Map.fromArray <$> arrayOf kvGen
    where
    kvGen :: Gen (Tuple ReplicaId Int)
    kvGen = Tuple <$> arbitrary <*> suchThat arbitrary (\x → greaterThanOrEq x 0
```

```haskell
stateBasedCRDTLaws ::
  forall t. StateBasedCRDT t ⇒ Arbitrary t ⇒ String → Eq t ⇒ Show t ⇒ Proxy t → Spec Unit
stateBasedCRDTLaws name _ =
  describe name do
    it "should be associative"
      $ quickCheck associativity
    it "should be commutative" do
      quickCheck commutativity
    it "should be idempotent" do
      quickCheck idempotence
    it "should have a neutral element" do
      quickCheck identity'
  where
  associativity a b c = (merge (merge a b) c) ≡≡ (merge a (merge b c))

  commutativity a b = merge a b ≡≡ merge b a

  idempotence a = merge a a ≡≡ a

  identity' a = (merge a mempty) ≡≡ a
```