Here's an even simpler answer: A Monad is something that "computes" when monadic context is collapsed by join :: m (m a) -> m a (recalling that >>= can be defined as (join .) . flip fmap). This is how Monads carry context through a sequential chain of computations: because at each point in the series, the context from the previous call is collapsed with the next.

A **free monad** satisfies all the Monad laws, but does not do any collapsing (i.e., computation). It just builds up a nested series of contexts. The user who creates such a free monadic value is responsible for doing something with those nested contexts, so that the meaning of such a composition can be deferred until after the monadic value has been created.