Contracts

Contracts tell us how to use a function. For example: `num-min :: (a :: Number, b :: Number) -> Number` tells us that the name of the function is `num-min`, it takes two inputs (both Numbers), and it evaluates to a `Number`. From the contract, we know `num-min(4, 6)` will evaluate to a `Number`. **Use the blank line under each contract for notes or sample code for that function!**

Name		Domain		Range
`string-repeat`	`::`	`(text :: String, repeat :: Number)`	`->`	`String`
`string-contains`	`::`	`(text :: String, search-for :: String)`	`->`	`Boolean`
`num-sqr`	`::`	`(n :: Number)`	`->`	`Number`
`num-sqrt`	`::`	`(n :: Number)`	`->`	`Number`
`num-min`	`::`	`(a :: Number, b:: Number)`	`->`	`Number`
`num-max`	`::`	`(a :: Number, b:: Number)`	`->`	`Number`
`count`	`::`	`(t :: Table, col :: String)`	`->`	`Table`
`mean`	`::`	`(t :: Table, col :: String)`	`->`	`Number`
`median`	`::`	`(t :: Table, col :: String)`	`->`	`Number`
`modes`	`::`	`(t :: Table, col :: String)`	`->`	`List<Number>`
`bar-chart`	`::`	`(t :: Table, col :: String)`	`->`	`Image`
`pie-chart`	`::`	`(t :: Table, col :: String)`	`->`	`Image`
`histogram`	`::`	`(t :: Table, values :: String, bin-width :: Number)`	`->`	`Image`