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
box-plot	::	(t :: Table, col :: String)	->	Image
box-plot(animals-table, "age")				
modified-box-plot	::	(t :: Table, col :: String)	->	Image
modified-box-plot(animals-table, "age")				
scatter-plot	::	(t :: Table, labels :: String, xs :: String, ys :: String)	->	Image
scatter-plot(animals-table, "species", "pounds", "weeks")				
image-scatter-plot	::	(t :: Table, xs :: String, ys :: String, f :: (Row -> Image))	->	Image
image-scatter-plot(animals-table, "pounds", "weeks", animal-img)				
r-value	::	(t :: Table, xs :: String, ys :: String)	->	Number
r-value(animals-table,"pounds", "weeks")				
lr-plot	::	(t :: Table, labels :: String, xs :: String, ys :: String)	->	Image
lr-plot(animals-table, "species", "pounds", "weeks")				
random-rows	::	(t :: Table, num-rows :: Number)	->	Table
random-rows(animals-table, 5)				
<Table>.row-n	::	(n :: Number)	->	Row
animals-table.row-n(5)				
<Table>.order-by	::	(col :: String, increasing :: Boolean)	->	Table
animals-table.order-by("species", true)				
<Table>.filter	::	(test :: (Row -> Boolean))	->	Table
animal-table.filter(is-cat)				
<Table>.build-column	::	(col :: String, builder :: (Row -> Any))	->	Table
animals-table.build-column("sticker", label)				
bar-chart-summarized	::	(t :: Table, labels :: String, values :: String)	->	Image
bar-chart-summarized(animals-table, "species", "pounds")				
pie-chart-summarized	::	(t :: Table, labels :: String, values :: String)	->	Image
pie-chart-summarized(animals-table, "age", "pounds")				