

Contracts

Contracts tell us how to use a function. For example: `ellipse :: (Number, Number, String, String) -> Image` tells us that the name of the function is `ellipse`, it takes four inputs (two Numbers and two Strings), and it evaluates to an `Image`. From the contract, we know `ellipse(100, 50, "outline", "red")` will evaluate to an `Image`.

| Name | | Domain | | Range |
|---|----|----------------------------------|----|--------|
| # num-sqr | :: | (Number) | -> | Number |
| <i>num-sqr(9)</i> | | | | |
| # num-sqrt | :: | (Number) | -> | Number |
| <i>num-sqrt(25)</i> | | | | |
| # star | :: | (Number, String, String) | -> | Image |
| <i>star(50, "solid", "teal")</i> | | | | |
| # circle | :: | (Number, String, String) | -> | Image |
| <i>circle(30, "outline", "fuchsia")</i> | | | | |
| # triangle | :: | (Number, String, String) | -> | Image |
| <i>triangle(80, "solid", "darkgreen")</i> | | | | |
| # square | :: | (Number, String, String) | -> | Image |
| <i>square(10, "outline", "red")</i> | | | | |
| # rectangle | :: | (Number, Number, String, String) | -> | Image |
| <i>rectangle(20, 80, "solid", "gold")</i> | | | | |
| # ellipse | :: | (Number, Number, String, String) | -> | Image |
| <i>ellipse(30, 70, "outline", "lightblue")</i> | | | | |
| # regular-polygon | :: | (Number, Number, String, String) | -> | Image |
| <i>regular-polygon(8, 40, "solid", "red")</i> | | | | |
| # radial-star | :: | (Number, Number, String, String) | -> | Image |
| <i>radial-star(17, 50, 10, "solid", "orange")</i> | | | | |