Figure 2.1 An analogy: mathematical functions vs. Java methods.

Java methods behave much like mathematical functions. For example, let

$$f(x) = x^2 + 2 \quad \text{and} \quad g(x, y) = y^2 - x^2.$$

Here's a table giving some properties of these two functions and a couple of others that you know:

function	**parameters**	**sample call**	**arguments**	**return value**
$\sqrt{x}$	x	$\sqrt{4}$	4	2
$\sin\theta$	θ	$\sin\pi/2$	$\pi/2$	1
$f(x)$	x	$f(3)$	3	11
$g(x, y)$	x, y	$g(3, 4)$	$3, 4$	7

The names of the parameters and arguments don't have to match. For example, if $x = 9$ then

$$g(6, x) \equiv g(6, 9) = 45.$$

Mathematical functions usually have parameters (and arguments) and return values that are *numbers*. Java methods have a richer set of parameter and return value types, but the idea is the same.

We can see how a Java method is supposed to behave by looking at its *signature*. For example, our class WashingMachine 'add' method has the signature:

public void add (String item)

which tells us the following about the method

- **public**—can be called from outside the class
- **void**—returns nothing
- **add**—the method name
- **(String item)**—the parameter list

Similarly, the 'getContents' method has the signature

public String getContents ()

'getContents' doesn't take a parameter, but it returns a value of type *String*.