This means that an organizer does not duplicate information that is available to it from elsewhere. If a user requests from the organizer information about the number of file names in it, the organizer will pass the question on to the **files** object, and then return whatever answer it gets from there.

Duplication of information or behavior is something we often work hard to avoid. Duplication can represent wasted effort, and can lead to inconsistencies where two things that should be identical turn out not to be, through error. We will have a lot more to say about duplication of functionality in later chapters.

4.6 Generic classes

The new notation using the angle brackets deserves a little more discussion. The type of our **files** field was declared as:

```
ArrayList<String>
```

The class we are using here is simply called **ArrayList**, but it requires a second type to be specified as a parameter when it is used to declare fields or other variables. Classes that require such a type parameter are called *generic classes.* Generic classes, in contrast to other classes we have seen so far, do not define a single type in Java, but potentially many types. The **ArrayList** class, for example, can be used to specify an *ArrayList of String*, an *ArrayList of Person*, an *ArrayList of Rectangle*, or an **ArrayList** of any other class that we have available. Each particular **ArrayList** is a separate type that can be used in declarations of fields, parameters, and return values. We could, for example, define the following two fields:

```
private ArrayList<Person> members;
private ArrayList<TicketMachine> machines;
```

These definitions state that **members** refers to an **ArrayList** that can store **Person** objects, while **machines** can refer to an **ArrayList** to store **TicketMachine** objects. Note that **ArrayList<Person>** and **ArrayList<TicketMachine>** are different types. The fields cannot be assigned to each other, even though their types were derived from the same **ArrayList** class.

Exercise 4.4 Write a declaration of a private field named **library** that can hold an **ArrayList**. The elements of the **ArrayList** are of type **Book**.

Exercise 4.5 Write a declaration of a local variable called **cs101** that can hold an **ArrayList** of **Student**.

Exercise 4.6 Write a declaration of a private field called **tracks** for storing a collection of **MusicTrack** objects.

Exercise 4.7 Write assignments to the **library**, **cs101**, and **track** variables (which you defined in the previous three exercises) to create the appropriate **ArrayList** objects. Write them once using diamond notation and once without diamond notation, specifying the full type.