# 2 Data abstraction

As already mentioned in section 1.2, data abstraction is a central concept in programming, especially in programming on a large scale. It involves combining several methods and variables into self consistent units, so that these units form logical building blocks (abstract data types) from which we assemble entire programmes. We only need to have abstract ideas of such building blocks in our minds when we put them together, not implementation details - hence the terms. In this way we hope to simplify programming to such an extent that even large programming larger programming tasks become manageable.

We first look at the principle behind data abstraction and the corresponding language constructs in Java. Examples follow, that illustrate the smooth transition from data abstractions to data structures - a distinction that lies primarily in the way it is viewed. Finally, we use data abstraction to realisation of recursive data structures.

2.1 Principle and Language Support

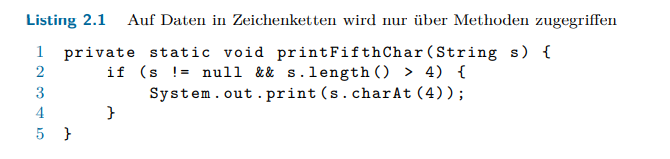
We introduce objects, classes and related concepts in Java, after we have looked at the use of some simple abstract data types by way of example.

2.1.1 Use of abstract data types

Let us recall String, the type of strings in Java. The is not only a reference type, but also an abstract data type:

While we know that somewhere in a string there is data from which the length and the individual characters of the string can be reconstructed, but we do not know exactly where and in what form this data is stored. Form this data is stored. We do not need to know this in order to use the string:

Listing 2.1 Auf Daten in Zeichenketten wird nur über Methoden zugegriffen



Since the formal parameter s, like any variable of a reference type, can contain null we make sure that s != null and that s has a sufficient length to access the fifth character - the one with index 4. the one with index 4, since the first character is referenced by index 0.

Expressions like s.length() and s.charAt(4) represent method calls in the string referenced by s.1 As results we get back the length and the character. We visualise the string like this: