Predefined Data Modules

Maude has a standard library of predefined modules that, by default, are entered into the system at the beginning of each session, so that any of these predefined modules can be imported by any other module defined by the user. Also, by default, the predefined functional module `BOOL` is automatically imported (in including mode) as a submodule of any user-defined module, unless such importation is explicitly disabled. These modules can be found in the file `prelude.maude` that is part of the Maude distribution.

We describe below those predefined modules that provide commonly used data types, including Booleans, numbers, strings, and quoted identifiers. The relationships among these modules are shown in the importation graph in Figure 9.1 where all the importations are in protecting mode.

We also describe typical parameterized collections of data types such as lists and sets, and associations such as maps and arrays. The chapter ends introducing the built-in linear Diophantine equation solver, defined in the file `linear.maude` that is also part of the Maude distribution.

Other predefined modules, also in the `prelude.maude` file, are discussed later; more specifically, the `META-LEVEL` module is discussed in Chapter 14, the `LOOP-MODE` module in Section 17.1, and the `CONFIGURATION` module in Sections 11.1 and 11.4.

Furthermore, this chapter also describes a predefined module `MACHINE-INT` for machine integers, which is obtained from the module `INT` of (arbitrary size) integers, but is distributed in a separate file `machine-int.maude`.

As explained in Section 4.4.10, many operators in predefined modules are declared with the `special` attribute, so that they are to be treated as built-in operators associated with appropriate C++ code by “hooks” specified after the `special` attribute. In what follows, to lighten the exposition, we will omit the details about such hooks in special operators, writing `special (...)` instead. The full definitions can be found in the file `prelude.maude`.

Most built-in data types are algebraically constructed, that is, they are built out of constants and constructor operators; however, floating point numbers (floats), strings, and quoted identifiers (qids) are treated as countable sets of constants and are represented by “special” operators `<Floats>`, `<Strings>`,
and <Qids>, respectively. These operators are used in specifying the hooks mentioned above, but they cannot be used explicitly in terms.

### 9.1 Boolean Values

There are four modules involving Boolean values, namely, TRUTH-VALUE, TRUTH, BOOL, and EXT-BOOL. The most basic one is TRUTH-VALUE, which has the following definition.

```plaintext
fmod TRUTH-VALUE is
  sort Bool .
  op true : -> Bool [ctor special (...)] .
  op false : -> Bool [ctor special (...)] .
endfm
```