OMNotebook – Interactive WYSIWYG Book Software for Teaching Programming,

Anders Fernström, Ingemar Axelsson, Peter Fritzson, Anders Sandholm, Adrian Pop
PELAB – Programming Environment Lab, Dept. Computer Science
Linköping University, S-581 83 Linköping, Sweden
{petfr,andsa,adrpo}@ida.liu.se

Abstract

OMNotebook is one of the first open source software systems that makes is possible to create interactive WYSIWYG books for teaching and learning programming. It has currently been used for course material (DrModelica) in teaching the Modelica language, but can easily be adapted to electronic books on teaching other programming languages, or even other subjects such as physics, chemistry, etc., where phenomena can be illustrated by dynamic simulations within the book. This could substantially improve teaching in a number of areas, including programming.

1 Need for more Interactive Learning

Traditional teaching methods are often too passive and do engage the student. A typical example is traditional lecturing.

Another typical learning method is reading a textbook on a subject matter. This is a good method, but sometimes requires a lot from the student. Also, learning programming needs interaction and programming exercises in order to grasp the concept.

A third way, would be to make the book active – be able to run programs and exercises within the book, and mix lecturing with doing exercises and reading in the interactive book.

2 Interactive Notebooks with Literate Programming

Interactive Electronic Notebooks are active documents that may contain technical computations and text, as well as graphics. Hence, these documents are suitable to be used for teaching and experimentation, simulation scripting, model documentation and storage, etc.

2.1 Mathematica Notebooks

Literate Programming (Knuth 1984) is a form of programming where programs are integrated with documentation in the same document. Mathematica notebooks (Wolfram 1997) is one of the first WYSIWYG (What-You-See-Is-What-You-Get) systems that support Literate Programming. Such notebooks are used, e.g., in the MathModelica modeling and simulation environment, e.g. see Figure 1 below and Chapter 19 in (Fritzson 2004)

2.2 OMNotebook

The OMNotebook software (Axelsson 2005, Fernström 2006) is a new open source free software that gives an interactive WYSIWYG (What-You-See-Is-What-You-Get) realization of Literate Programming, a form of programming where programs are integrated with documentation in the same document.

2.3 Tree Structured Hierarchical Document Representation

Traditional documents, e.g. books and reports, essentially always have a hierarchical structure. They are divided into sections, subsections, paragraphs, etc. Both the document itself and its sections usually have headings as labels for easier navigation. This kind of structure is also reflected in electronic notebooks. Every notebook corresponds to one document (one file) and contains a tree structure of cells. A cell can have different kinds of contents, and can even contain other cells. The notebook hierarchy of cells thus reflects the hierarchy of sections and subsections in a traditional document such as a book.

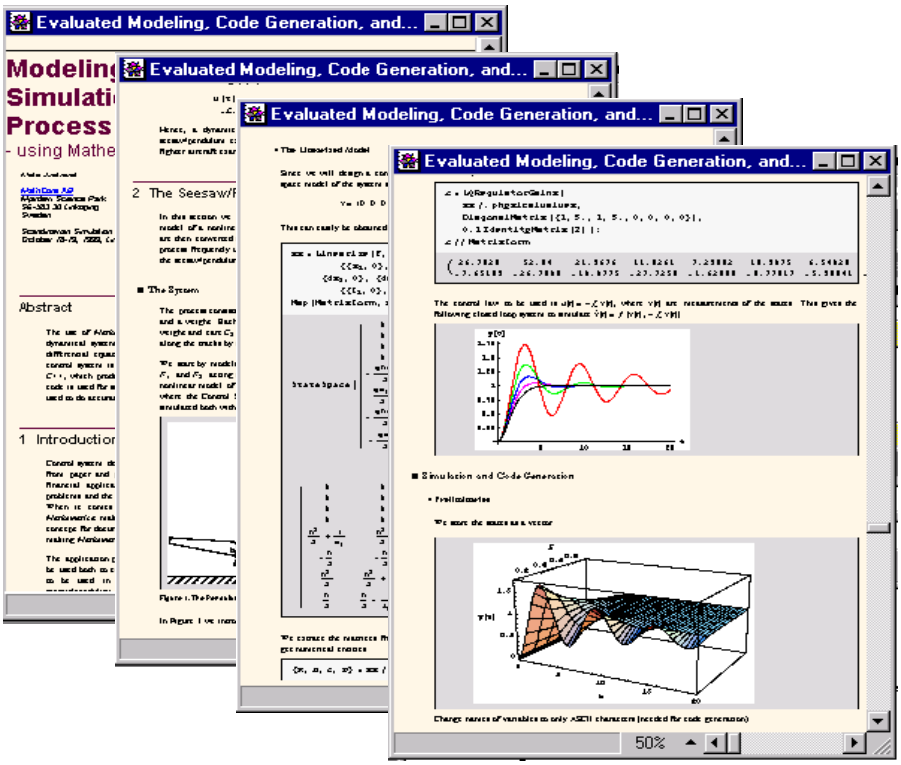

Figure 1. Examples of Mathematica notebooks in the MathModelica modeling and simulation environment.

3 The DrModelica Tutoring System – an Application of OMNotebook

Understanding programs is hard, especially code written by someone else. For educational purposes it is essential to be able to show the source code and to give an explanation of it at the same time.

Moreover, it is important to show the result of the source code's execution. In modeling and simulation it is also important to have the source code, the documentation about the source code, the execution results of the simulation model, and the documentation of the simulation results in the same document. The reason is that the problem solving process in computational simulation is an iterative process that often requires a modification of the original mathematical model and its software implementation after the interpretation and validation of the computed results corresponding to an initial model.

Most of the environments associated with equation-based modeling languages such as Modelica focus more on providing efficient numerical algorithms rather than giving attention to the aspects that should facilitate the learning and teaching of the language. There is a need for an environment facilitating the learning and understanding of Modelica. These are the reasons for developing the DrModelica teaching material for Modelica and for teaching modeling and simulation.

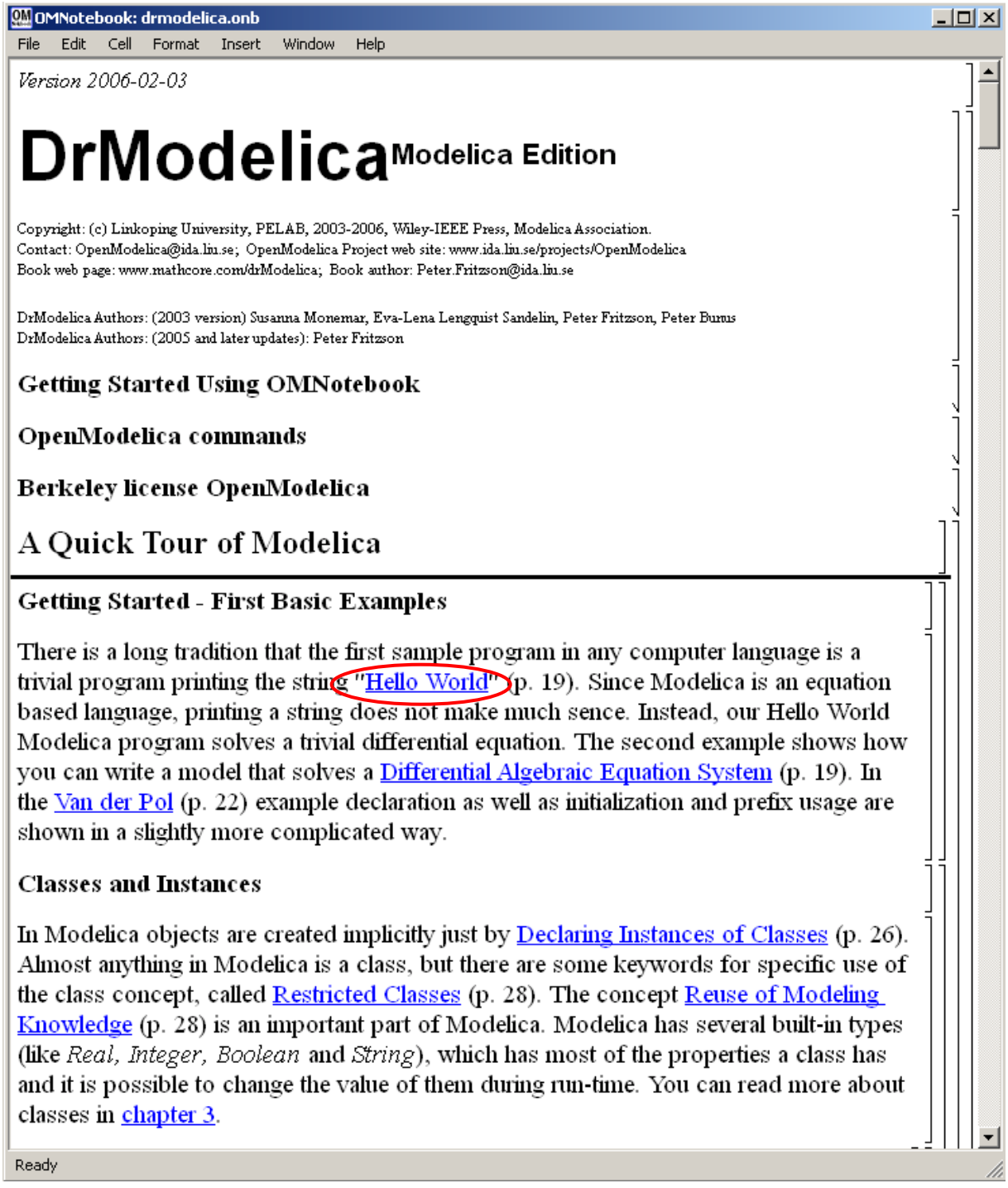

Figure 2. The start page (main page) of the DrModelica tutoring system using OMNotebook. The link to the HelloWorld example shown in Figure 3 is marked with an oval.

DrModelica has a hierarchical structure represented as notebooks. The front-page notebook is similar to a table of contents that holds all other notebooks together by providing links to them. This particular notebook is the first page the user will see (Figure 2).

In each chapter of DrModelica the user is presented a short summary of the corresponding chapter of the book "Principles of Object-Oriented Modeling and Simulation with Modelica 2.1" by Peter Fritzson. The summary introduces some *keywords*, being hyperlinks that will lead the user to other notebooks describing the keywords in detail.

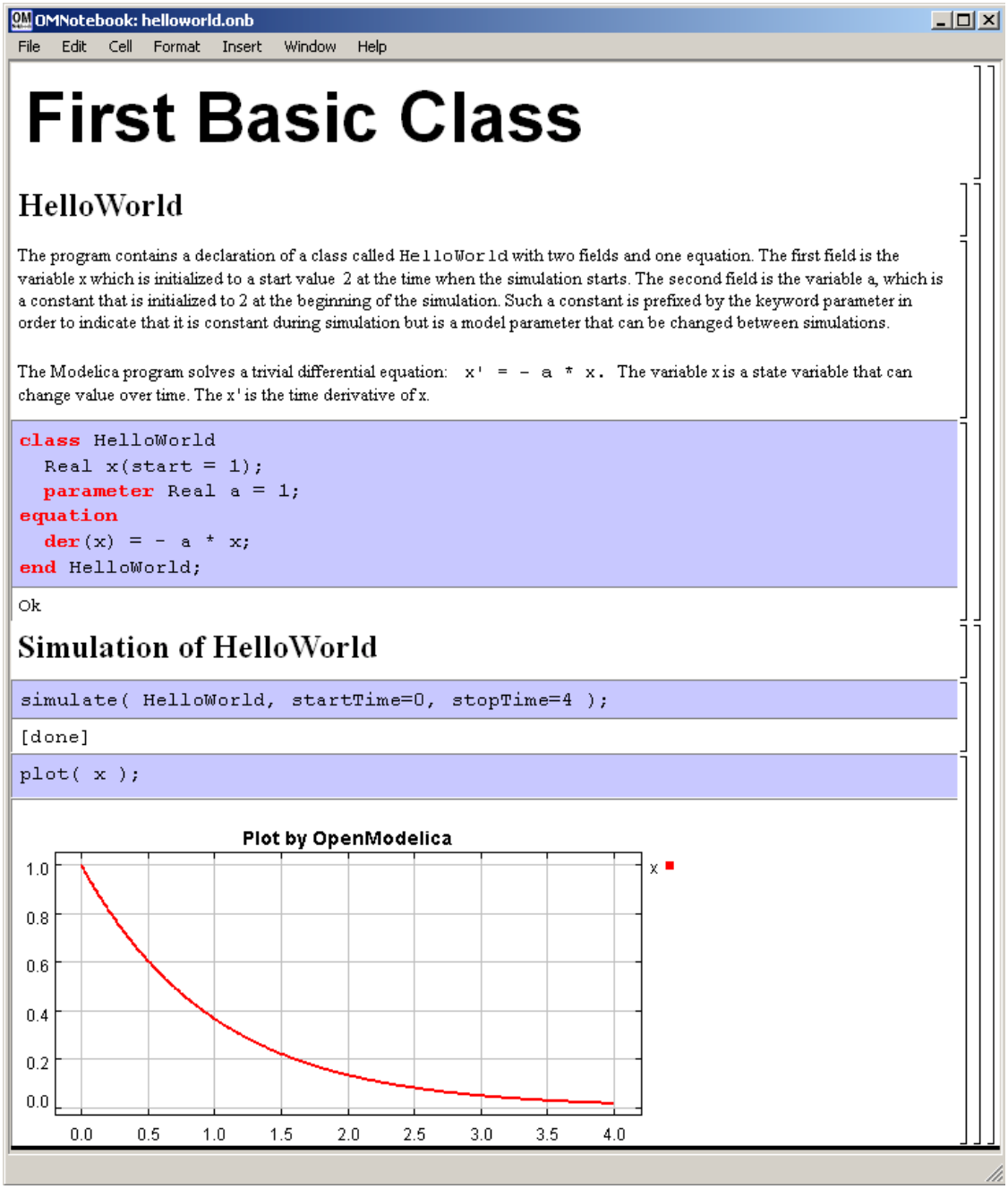

Figure 3. The `HelloWorld` class simulated and plotted using the OMNotebook version of DrModelica.

Now, let us consider that the link "*HelloWorld*" in DrModelica Section "Getting Started – First Basic Examples" in Figure 2 is clicked by the user. The new notebook, to which the user is being linked (see Figure 3), is not only a textual description but also contains one or more examples explaining the specific keyword. In this class, `HelloWorld`, a differential equation is specified.

No information in a notebook is fixed, which implies that the user can add, change, or remove anything in a notebook. Alternatively, the user can create an entirely new notebook in order to write his/her own programs or copy examples from other notebooks. This new notebook can be linked from existing notebooks.

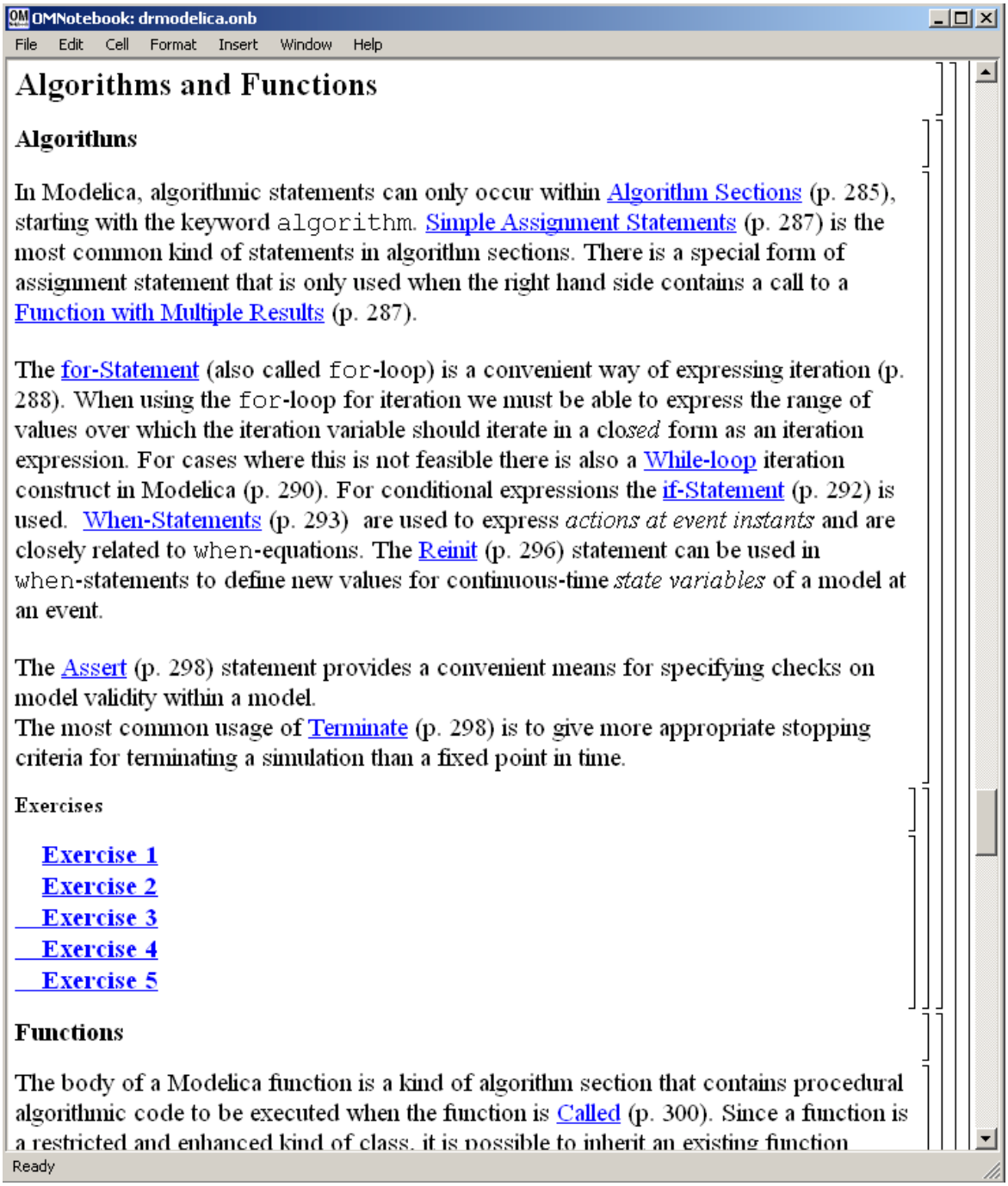

Figure 4. DrModelica Chapter “Algorithms and Functions” in the main page of DrModelica.

When a class has been successfully evaluated the user can simulate and plot the result, as depicted in Figure 3 for the simple `HelloWorld` example model..

After reading a chapter in DrModelica the user can immediately practice the newly acquired information by doing the exercises that concern the specific chapter. Exercises have been written in order to elucidate language constructs step by step based on the pedagogical assumption that a student learns better "*using the strategy of learning by doing*". The exercises consist of either theoretical questions or practical programming assignments. All exercises provide answers in order to give the user immediate feedback.

Figure 4 shows the algorithm part of the Chapter "Algorithms and Functions" of the DrModelica teaching material. Here the user can read about Modelica language constructs, like `algorithm` sections, when-statements, and `reinit` equations, and then practice these constructs by solving the exercises corresponding to the recently studied section.

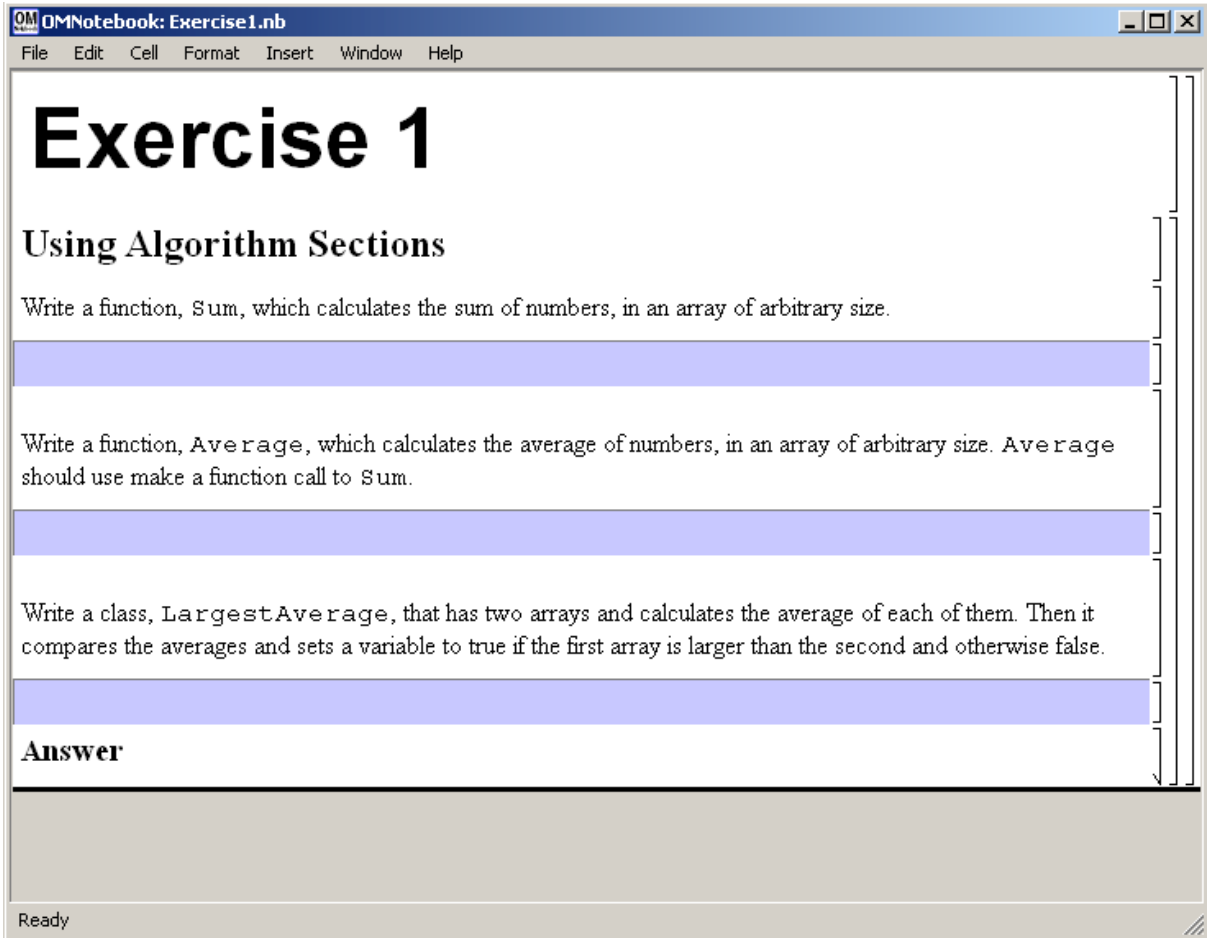

Figure 5. Exercise 1 in Chapter "Algorithms and Functions" of DrModelica.

Exercise 1 in the algorithm part of Chapter "Algorithms and Functions" is shown in Figure 5. In this exercise the user has the opportunity to practice different language constructs and then compare the solution to the answer for the exercise. Notice that the answer is not visible until the *Answer* section is expanded. The answer is shown in Figure 6.

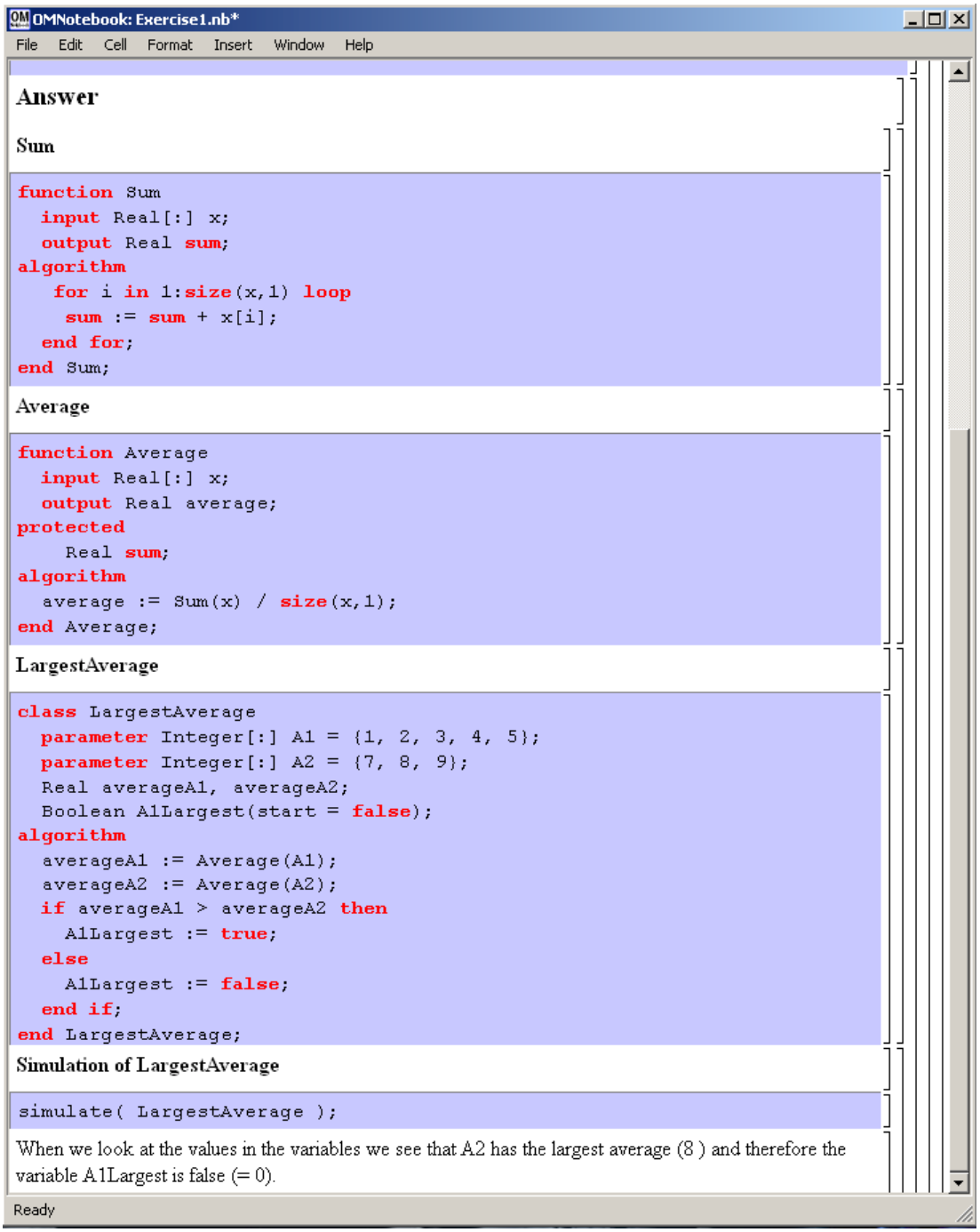

Figure 6. The answer section to Exercise 1 in Chapter “Algorithms and Functions” of DrModelica.

4 Conclusions

The OMNotebook software is one of the first open source software systems that makes is possible to create interactive WYSIWYG books for teaching and learning programming. It has currently been used for course material (DrModelica) in teaching the Modelica language, but can easily be adapted to electronic books on teaching other programming languages such as Java, Scheme, etc, through its general CORBA interface. This could revolutionize teaching in programming.

5 Acknowledgements

Support for Modelica-related research from SSF and Vinnova is gratefully acknowledged. Eva-Lena Lengquist-Sandelin and Susanna Monemar prepared the first version of the DrModelica tutorial material and thus also contributed to this paper. Daniel Hedberg at MathCore provided advice regarding Qt-based implementation. Peter Aronsson implemented large parts of the OpenModelica compiler and the communication protocol used from OMNotebook. PhD students at PELAB (Programming Environment Lab) contributed to various aspects of OpenModelica.

References

[1] Eric Allen, Robert Cartwright, Brian Stoler. DrJava: A lightweight pedagogic environment for Java. In *Proceedings of the 33rd ACM Technical Symposium on Computer Science Education* (SIGCSE 2002) (Northern Kentucky – The Southern Side of Cincinnati, USA, February 27 – March 3, 2002).

[2] Ingemar Axelsson. *OpenModelica Notebook for Interactive Structured Modelica Documents*. Final thesis, LITH-IDA-EX–05/080–SE, Linköping University, Linköping, Sweden, October 21, 2005.

[3] Anders Fernström. *Extending OMNotebook – An Interactive Notebook for Structured Modelica Documents.*Final thesis to be presented spring 2006, Dept. Computer and Information Science, Linköping University, Sweden.

[4] Peter Fritzson. *Principles of Object Oriented Modeling and Simulation with Modelica 2.1*, 940 pages, ISBN 0-471-471631, Wiley-IEEE Press. Feb. 2004.

[5] Peter Fritzson, et al. *OpenModelica Users Guide*, Preliminary Draft, for OpenModelica 1.3.1, Nov 28 2005. www.ida.liu.se/projects/OpenModelica.

[6] Knuth, Donald E. Literate Programming. *The Computer Journal*, NO27(2), pp. 97–111, May 1984.

[7] Eva-Lena Lengquist-Sandelin, Susanna Monemar, Peter Fritzson, and Peter Bunus. DrModelica – A Web-Based Teaching Environment for Modelica. In *Proceedings of the 44th Scandinavian Conference on Simulation and Modeling* (SIMS'2003), available at www.scan-sims.org. Västerås, Sweden. September 18-19, 2003.

[8] The Modelica Association. The Modelica Language Specification Version 2.2, March 2005. http://www.modelica.org.

[9] Stephen Wolfram. *The Mathematica Book*. Wolfram Media Inc, 1997.