Final Report: Data Structures Using Ada 9X

Mark Allen Weiss
School of Computer Science
Florida International University
University Park
Miami, FL 33199

Abstract

We describe a collection of supplements that can be used to update an Ada83-based Data Structures text and course to use Ada95.

1 Introduction

For many years, we have been teaching Data Structures using Ada. The package features provided in Ada83 have made it easy to incorporate abstraction, encapsulation, and information hiding into the data structures course. With Ada95, inheritance is added into the mix. The big question then, is how this affects the Data Structures course.

The answer to this question is far from simple, and can probably invoke a religious war. Some argue that object-oriented programming *requires* the use of inheritance. I have found that incorporating inheritance into data structures is very challenging.

It is clearly impossible to ignore inheritance, if for no other reason than the political reality of the competition (C++). The natural concepts of inheritance do not easily apply in a data structures course because technical details get in the way. The most natural application would be to derive a balanced search tree from an unbalanced search tree, but this seems to be a mess (I did it in the C++ version of my text, and I now regret my decision). Similarly, deriving a sorted linked list from a general linked list does not make sense in Ada.

It turns out that the data structures themselves do not need inheritance, but rather, the objects on which the data structures act use inheritance.

The 84 pages of supplements that I have written, combined with the revised Ada code, can be used in conjunction with my textbook *Data Structures and Algorithm Analysis in Ada* (Benjamin/Cummings, 1993) to teach data structures using Ada95.

The supplements are of two types. First, we list a collection of teaching materials, such as a syllabus, assignments, and examination questions. Then we list some of the handouts that can be given to students.

An important practical issue is to show that Ada95 has as much power as C++. Several of the handouts deal with the Ada95 parallels of C++ concepts such as command line arguments, constructors and destructors, and inheritance. This may well be Ada95's most significant advantage over Ada83 in a data structures course.

2 For the Instructor

Several supplements are available for the instructor. This includes:

1. *Annotated Syllabus for Ada95*

   A list of twenty-five lectures with readings and a brief overview of material to cover is provided. (6 pages)

2. *Examination Questions*

   Organized by chapter, roughly 140 multiple choice questions are provided with answers. Combined with end-of-chapter questions and the obvious "write the code to implement..." or "show the results of the following operations" there is more than enough material for several semesters of unique exams. This material is available in LaTeX form so it can be easily used. (28 pages)

3. *Assignments*

   A collection of sixteen assignments that I have used in the last year is provided. Combined with the assignments suggested in the end-of-chapter exercises, this yields plenty of material. (17 pages)

3 For the Students

Most of the code in the text has been revised to make use of Ada95 features. Each compilation unit is placed in a separate file.

In addition, I have written several handouts for my students. At FIU, we use *gnat* under SunOS 4.1.3, so I am assuming basic familiarity with UNIX. Again, these handouts are available in LaTeX and are easily customizable.

1. The handout *Using GNAT on Unix* describes how *gnat* is used and briefly summarizes some differences between Ada83 and Ada95. (5 pages)

2. The handout *Command Line Arguments in Ada95* describes how command line arguments are processed in Ada95. The mechanism appears to be somewhat cleaner than the C/C++ equivalent. (2 pages)

3. The handout *Using Inheritance for Heterogeneity* shows how inheritance can be used to achieve polymorphism and dynamic binding in Ada95. Once again, the mechanism is similar to what is seen in C++, though arguably cleaner. (8 pages)

4. The handout *Using Finalization in Ada95* illustrates how objects are initialized and finalized automatically in Ada95. (11 pages)

5. The handout *Summary of Ada95 Code* discusses some of the changes in the on-line code and lists, on a chapter-by-chapter basis, all compilation units. The new code includes some additional data structures: top-down splay trees, top-down red black trees, AA-trees, and pairing heaps, making the Ada95 supplement in agreement with the second editions of the text. (7 pages)

6. A complete listing of the online source code is included. (152 pages)