Errors and updates in Programming in Ada 2012, 1st edition

page line: item
NB line number gives distance in picas from top of text (6 picas = 72 pts = 1 inch)

xvi 10: Corrected typos in Dijkstra and Hoare.

29 14: Changed the exception to Buffer_Error.

29 28: Changed Buffer of Get to B of Get.

29 38: Changed Error to Buffer Error.

39 38: In Exercise 3.3/5; reminded reader that mass is area.

51 44: Noted that we can now write Put(N'Image); see AI12-124.

66 13: Changed reference to 10646:2011.

74 44: After "we will encounter some examples later", added reference to Section 10.5 and Section 11.2.

76 21: Replaced "idea" by "concept".

77 2: Replaced "It follows from the more general principle" by "It is an example of the more general principle".

77 8: Reference to chapter 10 changed to Section 10.7 and mentioned named blocks.

85 42: Changed the number for 1.0/81.0 to be 0.0123456790123... It had been wrong since the first edition of Programming in Ada nearly 40 years ago!

95 9: Added that qualification checks all subtype properties including predicates; added reference to Section 16.4; see AI12-100.

96 32: Changed reference to Section 9.4 to Section 9.1.

99 28: Changed reference to Section 9.4 to Section 9.1.

124 39: Added a remark on nominal subtype and reference to unchecked union. Added nominal subtype to the Index.

157 7: Noted that if the range in a quantified expression is null then the expression is always True in the case of **for all** and always False in the case of **for some**. See AI12-258.

167 1: Changed reference to Section 11.5 to Section 12.5.

168 21: Added note that the result of an expression function never needs double parentheses. Similar rule applies to if expression etc. Added example of function Conjugate and function Piggy with string. See AI12-157.

172 20: Changed reference to Chapter 10 to Chapter 11.

174 27: Results of unconstrained array types might be inefficient, added for example type String.

175/6 34: Deleted sentence about junk values and rewrote the example to introduce the aspect Default_Value and the use of in out parameters. See AI12-74.

190 36: Added reference to downward closure problem explained in Section 11.8.

193 4: In diagram (b) deleted null for variable N.

213 40: Note that the function = is correct; but early compilers were not correct.

223 32: Last sentence mentioned Ada 2102; changed it to Ada 2012.

253 36: Last sentence of last full paragraph regarding use of type as an access parameter should not use the term deferred which is only used for constants. Rewrote it.

258 10: Added **end if** to procedure Action.

265 36: Changed Q to P so that the last bit reads "the user of P does not care about R".

283 28: Changed reference to Section 10.6 to Section 10.7.

294 10: Added discussion about using aspect for Pure rather than pragma. See AI12-154. Changed all uses of library unit pragmas such as Pure and Preelaborate to use aspects. In the appendix moved all relevant pragmas to the list of aspects.

301 30: Changed reference to Chapter 20 to Chapter 21.

318 21: Question 2 assumed that Object is not abstract so referred to Section 14.1 to clarify and then interchanged questions 1 and 2.

319 32: Primitive operations of Root changed to primitive operations of Object.

328 18: Changed reference to Section 21.6 to Section 21.7.

333 25: Changed the phrase "and no concrete types" to "and no concrete operations".

338 20: Clarified why return expression needs **with null record** and added reference to Section 14.1.

346 2: Changed reference to Section 19.3 to Exercise 21.3.

346 5: Changed reference to Chapter 18 to Chapter 20.

346 19: Changed reference to Section 11.5 to Section 12.5.

361 40: Changed reference to Section 11.7 to Section 11.6.

364 20: Noted that exceptions raised in declaration parts of subprograms are propagated; added reference to Section 15.6.

367 18: Changed reference to Exercise 13.7(2) to Exercise 14.7(2).

368 34: Changed remark about Pop being incorrect since functions can now take a parameter of mode **out**. It was OK in Ada 2005.

370 37: Changed in the next chapter to in Chapter 17.

375 42: Changed reference to Section 13.4 to Section 14.4.

382 6: Added note that the example is Section 15.6a Exception Scope on the web.

383 12: Made amount remaining the proper identifier Amount_Remaining.

383 23: Changed "the key were free" to "the key was free" in paragraph after Withdraw; subjunctive a bit pedantic here.

389 8: The reference to Section 16.5 was unhelpful since it only described Predicate_Failure. Deleted that paragraph, it was very misleading, we cannot change the exception for pre- and postconditions.

389 14: Added example of the form of Assertion_Policy that gives the aspect or aspects concerned.

389 28: Noted that X'Old has the same type as X even if anonymous. See AI12-32.

390 24: Delete discussion on Assertion_Policy. It is possible for pre to be Check and post to be Ignore.

391 32: This was incorrect since Pinc is not a primitive operation of Integer and so is not inherited. Used My_Integer instead and explained why.

392 2: The sentence "we cannot inherit an operation and change its conditions at the same time" was curious. It seemed obvious or foolish. It was deleted.

392 39: In "And if no new ones are supplied then they are by default taken to be true", clarified that "they" refers to the pre- and post conditions.

392 40: Noted that pre- and post conditions are always given on a specification but are sort of part of the body. And can also be given on an expression function but not if it is just acting as a completion. See AI12-105.

397 24: Changed OP to Op.

399 31: Changed to read "need not always be satisfied".

400 40: Deleted "and" after Is_Unduplicated.

403 27: Noted that checks apply to the initialization of deferred constants. See AI12-49.

404 4: Noted that Type_Invariant'Class can be applied to interface types as well as to abstract types. See AI12-41.

405 11: The type Ring_Pt was incorrect. Must be declared as **with private**. Can only apply invariant to a private type. Added full declaration in private part thus

```
package Places.Inner is
  type Ring_Pt is new Disc_Pt with private
    with Type_Invariant'Class => Check_Out(Ring_Pt);

  function Check_Out(R: Ring_Pt) return Boolean
    with Inline;
private
  type Ring_Pt is new Disc_Pt with null record;

  function Check_Out(R: Ring_Pt) return Boolean is
```

(R.X**2 + R.Y**2 >= 0.25); -- *note no allowance for inaccuracy*

end Places.Inner;

414 18: Changed Real_File_Type to Read_File_Type.

439 29: Added reference to Chapter 22.

452 6: Changed reference to Section 12.7 to Section 13.7.

459 27: Changed reference to be to the function Rev in Section 10.1.

463 8: Changed reference to Section 16.1 to Section 18.1.

467 32: Added reference to 21.4a Multiple Views on the web for garment and other examples.

476 19: Changed reference to Section 3.4 to Section 6.8.

491 24: Changed reference to Exercise 14.3(1) to Exercise 14.3(2).

507 26: Added that entries can also have pre- and postconditions.

515 42: Added reference to aspect Exclusive_Functions. See AI12-129.

538 19; Added note about abort completion points. We can do that by adding Delay 0.0 or Yield. Also added reference to Section 26.2. See AI12-98.

544 12: Added brief discussion on pre- and postconditions and requeue. See AI12-90.

549 41: Added to checklist that entries can have pre- and postconditions.

570 8: Added reference to Section 21.5a Double dispatching on the web.

581 28: Changed to **type** Get_Function **is access function return** Object'Class.

581 32: Corrected to be Register(Get_Circle'Access, 'C');

591 29: Deleted the aspect Preelaborate from the package Dispatching_Domains.

591 36: See AI-33 final version, corrected to use CPU_Range thus

function Create(First: CPU; Last: CPU_Range) **return** D_D;
function Get_Last_CPU(Domain: D_D) **return** CPU_Range;

601 15: Changed the word in to be in bold.

610 24: Changed the parameter to Activity: **not null access** Descriptor'Class.

617 28: Corrected function Time_Of to use semicolon not comma in parameter list.

628 col 2, 40 Changed package Complex_Stuff to use Re and not Rl.

629 follows on from 628.

631 21: Changed the page numbers in Part 4 heading which were incorrect to Playing Pools is 803, Annexes is 807, and Finale is 823.

637 7: Added (the annex is indicated by the code such as RT).

639 5: Clarified that Equal_Case_Insensitive and Less_Case_Insensitive are also subprograms of Fixed, Bounded, and Unbounded.

639 24: Changed reference for Wide_Strings to be 23.3 and not 26.2 RT.

639 37: Added Unchecked_Deallocate_Subpool and reference 25.4.

651 28: Noted that in the case of the explicit pad being A, the intermediate string should be "ABCDEA" so the result is "BCDEA".

653 6: Changed to use named calls of To_Set to avoid ambiguities, thus

```
S := To_Set(Singleton => '?');       -- a single character
S := To_Set(Sequence => "AEIOU");    -- a string
S := To_Set(Span => ('0', '9'));        -- a range
S := To_Set(Ranges => (('A', 'Z'), ('a', 'z')));  --  an array of two ranges
```

664 40: Added procedure Flush to Sequential_IO. See AI12-130.

666 26: Added a mention of Flush. See AI12-130.

682 22: Changed reference to Section 21.6 to Section 21.7.

682 28: Changed reference to Section 21.6 to Section 21.7.

698 24: Changed reference to the CD to the website.

701 32: Corrected to be **pragma** Remote_Types(Doubly_Linked_Lists);

709 2: Changed Anther to Another.

710 22: Added full details of the subtype Extended_Index (in smaller font).

719 24: Added **use** Store_Maps after The_Store: Store_Maps.Map;

736 12: In the function Equivalent_Keys added a space before **return.**

739 30: Reversed the parameters of Find_In_Subtree and Ancestor_Find.

740 21: **for** C **in** The_Tree.Iterate_Subtree(S) **loop** replaced by

```
for C in Iterate_Subtree(S) loop
```

743 10: **for** C **in** Parent.Iterate_Children **loop** was nonsense. Changed to read

This is called if we write

```
for C in The_Tree.Iterate_Children(P) loop
                    -- do something via cursor C
end loop;
```

and iterates over all the children (using cursor C) of the parent node designated by the cursor P from P.First_Child to P.Last_Child.

and so on. Compare with Iterate Subtree which does grandchildren etc as well. Also corrected the reverse one. Noted also that the subtrees cannot be reversed but the children can.

743 43: Added the discriminant Op to the type El to make it definite.

746 23: The symbolic procedure Do_Node was not meant to be true Ada, but it might be improved nevertheless so changed it to

```
procedure Do_Node(N: Node) is
begin
  for CN in First_Child(N) .. Last_Child(N) loop
    Do_Node(CN);
  end loop;
  if not Is_Root(N) then
    Do_Element(N);
  end if;
end Do_Node;
```

That is also not real Ada. If it were an exercise the answer might be

```
procedure Do_Node(N: Cursor) is
begin
  for CN in Iterate_Children(My_Tree, N) loop
    Do_Node(CN);
  end loop;
  if not Is_Root(N) then
    Do_Element(N);
  end if;
end Do_Node;
```

749 9: Clarified to say, two are bounded and two are unbounded; two have priority and two do not. The names are ...

752 3: Changed reference to Ada 2020 to "a future version of Ada".

754 12: Made package bold.

757 5: Added comment "-- *or simply L<R*" to expression Degree'Pos(L) < Degree'Pos (R).

759 22: Deleted bounded twice in this para to say "Lists, ordered maps ... and then in the case of vectors, and hashed maps...".

765 24: This version of Add_Entry does not work, since the container is indefinite. So added explanation that the user has to choose one of the other versions.

765 45: Changed to **if** Indexes.Key(M_Cursor)'Length = 4 **then**.

766 13: Changed Key(M_Cursor)'Length to Indexes.Key(M_Cursor)'Length.

766 15: Changed Element(L_Cursor).Page to Places.Element(L_Cursor).Page.

766 27: Changed M_Cursor.Element to Indexes.Element(M_Cursor).

766 34: Changed **is** to **in**.

767 31: Changed to say the procedure for unconstrained arrays is identical.

773 9: The first procedure Prepend is not for Lists so deleted Y.

776 20: Changed the box for multiway trees for function Iterate to be Y Forward.

778 4: Changed the last box for multiway trees to be Y.

782 4: Noted that null procedures never freeze anything and an expression function acting as a completion just freezes the expression. See AI12-157 and AI12-103.

785 26: Changed the word tue to true.

785 28: Noted that Has_Same_Storage is false in the case of zero bits. See AI12-77.

786 36: Noted that alignments need not be identical; alignment of source can be a multiple of that of the target, as in AI05-78, so changed to compatible.

788 9: Changed reference to Section 11.6 to Section 11.5.

797 33: Noted that the default storage pool can be changed back to Standard. See AI12-3. Noted also the interaction with Storage_Size. See AI12-43.

800 15: Added note that the nominal subtype of X is unconstrained whereas that of Y is constrained. Added reference to Section 8.2. Pointed out that Program Error is only raised when comparing objects as a whole. Changed the example to read

```
if X.SP_Value = 45.7 then          -- OK, but False
if X = Y then                      -- raises Program_Error
```

800 38: Corrected to say has the value zero of type Float.

808 38: Noted that Discard_Names is an aspect not a pragma. See AI12-72. Added Greek example.

817 1: Changed reference to Section 17.4 to Section 19.4.

824 44: Changed reference to Section 20.5 to Section 18.5.

825 12: Operands of **in** and **not in** needed clarification. Made it simply "any combination", deleted tab before combination.

849 18: Noted that the web address www.sparkada.com is no longer valid; one should use www.adacore.com/about-spark.

854 15: Noted that S'Image(X) can be abbreviated to X'Image. See AI12-124.

860 28: Added aspect Exclusive_Functions. See AI12-129. Also added aspect Discard_Names.

862 41: Deleted pragma Discard_Names. It is now an aspect.

863 36: Preeleborable_Initialization does not only apply to private types. See AI05-28. So deleted "Applies to a private type."

865 27: Added real-time annex restriction No_Dynamic_CPU_Assignment (AI12-55) and No_Tasks_Unassigned_To_CPU (AI12-117).

876 6: Changed syntax for relation. See AI12-152. Use *tested*_simple_expression with membership_choice.

880 36: Changed syntax for expression_function_declaration. Added same again with (expression) replaced by aggregate.

884 24: Changed syntax for raise_expression. See AI12-152. Use *string*_simple_expression.

885 9: Changed syntax for digits_constraint and delta_constraint to use *static*_simple_expression.

889 14: Changed syntax for protected_operation_item. See AI12-147. Added null_procedure_declaration and expression_function_declaration.

891 20: Changed syntax for storage_pool_indicator. Add Standard. See AI12-3.

908 col 1, 21: Answer 12.1/2 used Rl for real part whereas question used Re. Changed answer to match question. Same applies to Answer 12.2/1 and Answer 12.2/2.

912 col 1, 20: Similarly changed Answer 13.3/1 to use Re_Part and not Rl_Part.

914 22: In Answer 16.3/1 added **end if**; after **return** False;

914 29: Answer 16.3/2 was hopelessly wrong. As soon as we find a pair that are equal we are done, so should be **some** not **all** and of course we need a **no**t as well giving

```
function Is_Unduplicated(S: Stack) return Boolean is
(not (for some I in 1 .. S.Top-1 =>
        for some J in I+1 .. S.Top => S.S(I) = S.S(J))));
```

Note that inserting **not** requires yet another pair of parentheses!

930 20: Noted that the web address www.sparkada.com is no longer valid; one should use www.adacore.com/about-spark.

931 Added the new AIs to the Index; also a few other items such as: abort completion points, Discard_Names, nominal subtype, statically unevaluated, subtype predicates.

947 Added further examples to the examples index, such as: Greek, Indexes, Grim_Reaper, Make_Process, Signal, Store_Maps.

Web answers

21 14.3(5) Added comment: *Select_Seat etc as before*.

The confusion regarding complex numbers probably arose because in Programming in Ada in 1981, it used complex numbers as a question and in the answer it used Rl. Remember that Ada 83 did not have a numeric library. When Ada 95 came along it used Re for the complex number package in the Numerics annex. The text of examples and questions were changed to match but the answers were overlooked.

February 2021