TECHNICAL REPORT

ISO/IEC TR 24772-2

First edition
2020-04

Programming languages — Guidance to avoiding vulnerabilities in programming languages —

Part 2: Ada

Langages de programmation — Conduite pour éviter les vulnérabilités dans les langages de programmation —

Partie 2: Ada

This document is a preview generated by EVS

Reference number
ISO/IEC TR 24772-2:2020(E)

© ISO/IEC 2020

This document is a preview generated by EVS



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2020

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword ... **vii**
Introduction ... **viii**
1 Scope ... **1**
2 Normative references ... **1**
3 Terms and definitions ... **1**
4 Language concepts ... **6**
4.1 Enumeration type ... 6
4.2 Exception ... 6
4.3 Hiding ... 6
4.4 Implementation defined ... 6
4.5 Type conversions ... 6
4.6 Operational and Representation Attributes ... 7
4.7 User defined types ... 7
4.8 Pragma compiler directives ... 7
4.8.1 **Pragma** Atomic ... 7
4.8.2 **Pragma** Atomic_Components ... 7
4.8.3 **Pragma** Convention ... 7
4.8.4 **Pragma** Detect_Blocking ... 7
4.8.5 **Pragma** Discard_Names ... 7
4.8.6 **Pragma** Export ... 8
4.8.7 **Pragma** Import ... 8
4.8.8 **Pragma** Normalize_Scalars ... 8
4.8.9 **Pragma** Pack ... 8
4.8.10 **Pragma** Restrictions ... 8
4.8.11 **Pragma** Suppress ... 8
4.8.12 **Pragma** Unchecked_Union ... 8
4.8.13 **Pragma** Volatile ... 8
4.8.14 **Pragma** Volatile_Components ... 8
4.9 Separate compilation ... 8
4.10 Storage pool ... 8
4.11 Unsafe programming ... 9
5 General guidance for Ada ... **9**
5.1 Ada language design ... 9
5.2 Top avoidance mechanisms ... 10
6 Specific guidance for Ada ... **11**
6.1 General ... 11
6.2 Type system [IHN] ... 11
6.2.1 Applicability to language ... 11
6.2.2 Guidance to language users ... 11
6.3 Bit representation [STR] ... 11
6.3.1 Applicability to language ... 11
6.3.2 Guidance to language users ... 11
6.4 Floating-point arithmetic [PLF] ... 12
6.4.1 Applicability to language ... 12
6.4.2 Guidance to language users ... 12
6.5 Enumerator issues [CCB] ... 12
6.5.1 Applicability to language ... 12
6.5.2 Guidance to language users ... 13
6.6 Conversion errors [FLC] ... 13
6.6.1 Applicability to language ... 13
6.6.2 Guidance to language users ... 13
6.7 String termination [CJM] ... 14

This document is a preview generated by EVS

© ISO/IEC 2020 – All rights reserved

6.8 Buffer boundary violation (buffer overflow) [HCB] 14
6.9 Unchecked array indexing [XYZ] 14
6.9.1 Applicability to language 14
6.9.2 Guidance to language users 14
6.10 Unchecked array copying [XYW] 14
6.11 Pointer type conversions [HFC] 14
6.11.1 Applicability to language 14
6.11.2 Guidance to language users 15
6.12 Pointer arithmetic [RVG] 15
6.13 Null pointer dereference [XYH] 15
6.13.1 Applicability to the language 15
6.13.2 Guidance to language users 15
6.14 Dangling reference to heap [XYK] 15
6.14.1 Applicability to language 15
6.14.2 Guidance to language users 16
6.15 Arithmetic wrap-around error [FIF] 16
6.16 Using shift operations for multiplication and division [PIK] 16
6.17 Choice of clear names [NAI] 16
6.17.1 Applicability to language 16
6.17.2 Guidance to language users 17
6.18 Dead store [WXQ] 17
6.18.1 Applicability to language 17
6.18.2 Guidance to language users 17
6.19 Unused variable [YZS] 17
6.19.1 Applicability to language 17
6.19.2 Guidance to language users 17
6.20 Identifier name reuse [YOW] 18
6.20.1 Applicability to language 18
6.20.2 Guidance to language users 18
6.21 Namespace issues [BJL] 18
6.22 Initialization of variables [LAV] 18
6.22.1 Applicability to language 18
6.22.2 Guidance to language users 19
6.23 Operator precedence/order of evaluation [JCW] 19
6.23.1 Applicability to language 19
6.23.2 Guidance to language users 19
6.24 Side-effects and order of evaluation [SAM] 20
6.24.1 Applicability to language 20
6.24.2 Guidance to language users 20
6.25 Likely incorrect expression [KOA] 20
6.25.1 Applicability to language 20
6.25.2 Guidance to language users 21
6.26 Dead and deactivated code [XYQ] 21
6.26.1 Applicability to language 21
6.26.2 Guidance to language users 21
6.27 Switch statements and static analysis [CLL] 21
6.27.1 Applicability to language 21
6.27.2 Guidance to language users 22
6.28 Demarcation of control flow [EOJ] 22
6.29 Loop control variables [TEX] 22
6.30 Off-by-one error [XZH] 22
6.30.1 Applicability to language 22
6.30.2 Guidance to language users 23
6.31 Unstructured programming [EWD] 23
6.31.1 Applicability to language 23
6.31.2 Guidance to language users 23
6.32 Passing parameters and return values [CSJ] 23
6.32.1 Applicability to language 23

This document is a preview generated by EVS

 © ISO/IEC 2020 – All rights reserved

6.32.2 Guidance to language users 23
6.33 Dangling references to stack frames [DCM] 23
6.33.1 Applicability to language 23
6.33.2 Guidance to language users 24
6.34 Subprogram signature mismatch [OTR] 24
6.34.1 Applicability to language 24
6.34.2 Guidance to language users 24
6.35 Recursion [GDL] 25
6.35.1 Applicability to language 25
6.35.2 Guidance to language users 25
6.36 Ignored error status and unhandled exceptions [OYB] 25
6.36.1 Applicability to language 25
6.36.2 Guidance to language users 25
6.37 Type-breaking reinterpretation of data [AMV] 25
6.37.1 Applicability to language 25
6.37.2 Guidance to language users 26
6.38 Deep vs. shallow copying [YAN] 26
6.38.1 Applicability to language 26
6.38.2 Guidance to language users 26
6.39 Memory leak and heap fragmentation [XYL] 26
6.39.1 Applicability to language 26
6.39.2 Guidance to language users 27
6.40 Templates and generics [SYM] 27
6.41 Inheritance [RIP] 27
6.41.1 Applicability to language 27
6.41.2 Guidance to language users 27
6.42 Violations of the Liskov substitution principle or the contract model [BLP] 28
6.42.1 Applicability to language 28
6.42.2 Guidance to language users 28
6.43 Redispatching [PPH] 28
6.43.1 Applicability to language 28
6.43.2 Guidance to language users 28
6.44 Polymorphic variables [BKK] 29
6.44.1 Applicability to language 29
6.44.2 Guidance to language users 29
6.45 Extra intrinsics [LRM] 29
6.46 Argument passing to library functions [TRJ 29
6.46.1 Applicability to language 29
6.46.2 Guidance to language users 30
6.47 Inter-language calling [DJS] 30
6.47.1 Applicability to language 30
6.47.2 Guidance to language users 30
6.48 Dynamically-linked code and self-modifying code [NYY] 30
6.49 Library signature [NSQ] 30
6.49.1 Applicability to language 30
6.49.2 Guidance to language users 31
6.50 Unanticipated exceptions from library routines [HJW] 31
6.50.1 Applicability to language 31
6.50.2 Guidance to language users 31
6.51 Pre-processor directives [NMP] 31
6.52 Suppression of language-defined run-time checking [MXB] 31
6.52.1 Applicability to Language 31
6.52.2 Guidance to language users 32
6.53 Provision of inherently unsafe operations [SKL] 32
6.53.1 Applicability to Language 32
6.53.2 Guidance to language users 32
6.54 Obscure language features [BRS] 32
6.54.1 Applicability to language 32

This document is a preview generated by EVS

© ISO/IEC 2020 – All rights reserved

6.54.2 Guidance to language users 32
6.55 Unspecified behaviour [BQF] 32
6.55.1 Applicability to language 32
6.55.2 Guidance to language users 33
6.56 Undefined behaviour [EWF] 33
6.56.1 Applicability to language 33
6.56.2 Guidance to language users 34
6.57 Implementation-defined behaviour [FAB] 34
6.57.1 Applicability to language 34
6.57.2 Guidance to language users 35
6.58 Deprecated language features [MEM] 35
6.58.1 Applicability to language 35
6.58.2 Guidance to language users 35
6.59 Concurrency — Activation [CGA] 35
6.59.1 Applicability to language 35
6.59.2 Guidance to language users 35
6.60 Concurrency — Directed termination [CGT] 36
6.60.1 Applicability to language 36
6.60.2 Guidance to language users 36
6.61 Concurrent data access [CGX] 36
6.61.1 Applicability to language 36
6.61.2 Guidance to language users 36
6.62 Concurrency — Premature termination [CGS] 36
6.62.1 Applicability to language 36
6.62.2 Guidance to language users 36
6.63 Protocol lock errors [CGM] 37
6.63.1 Applicability to language 37
6.63.2 Guidance to language users 37
6.64 Reliance on external format strings [SHL] 37
7 Language-specific vulnerabilities for Ada 37
8 Implications for standardization 37
Bibliography 39
Index 41

This document is a preview generated by EVS

© ISO/IEC 2020 – All rights reserved

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*.

This first edition cancels and replaces ISO/IEC TR 24772:2013, which has been split into several parts.

This document is intended to be used with ISO/IEC TR 24772-1, which discusses programming language vulnerabilities in a language independent fashion.

A list of all parts in the ISO/IEC 24772 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

This document is a preview generated by EVS

© ISO/IEC 2020 – All rights reserved

Introduction

This document provides guidance for the programming language Ada so that application developers considering Ada or using Ada can better avoid the programming constructs that lead to vulnerabilities in software written in the Ada language and their attendant consequences. This guidance can also be used by developers to select source code evaluation tools that can discover and eliminate some constructs that can lead to vulnerabilities in their software. This document can also be used in comparison with companion documents and with the language-independent ISO/IEC TR 24772-1, to select a programming language that provides the appropriate level of confidence that anticipated problems can be avoided.

It should be noted that this document is inherently incomplete. It is not possible to provide a complete list of programming language vulnerabilities because new weaknesses are discovered continually. Any such report can only describe those that have been found, characterized and determined to have sufficient probability and consequence.

This document is a preview generated by EVS

© ISO/IEC 2020 – All rights reserved

Programming languages — Guidance to avoiding vulnerabilities in programming languages —

Part 2: Ada

This document is a preview generated by EVS

1 Scope

This document specifies software programming language vulnerabilities to be avoided in the development of systems where assured behaviour is required for security, safety, mission-critical and business-critical software. In general, this document is applicable to the software developed, reviewed or maintained for any application.

Vulnerabilities described in this document present the way that the vulnerability described in ISO/IEC TR 24772-1 are manifested in Ada.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 2382, *Information technology — Vocabulary*

ISO/IEC 8652, *Information technology — Programming languages — Ada*

ISO/IEC TR 24772-1, *Programming languages — Guidance to avoiding vulnerabilities in programming languages — Part 1: Language-independent guidance*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 2382, ISO/IEC 8652, ISO/IEC TR 24772-1 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
abnormal representation
representation of an object that is incomplete or that does not represent any valid value of the object's subtype

3.2
access-to-object
pointer to an object

3.3
access-to-subprogram
pointer to a subprogram (function or procedure)

© ISO/IEC 2020 – All rights reserved