# Annex C(*informative*)Language Specific Vulnerability Template

Each language-specific Part should have the following heading information and initial sections:

|  |
| --- |
| ISO IEC 24772-X(Informative)Vulnerability descriptions for language [language]Foreword**[**ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.In exceptional circumstances, when the joint technical committee has collected data of a different kind from that which is normally published as an International Standard (“state of the art”, for example), it may decide to publish a Technical Report. A Technical Report is entirely informative in nature and shall be subject to review every five years in the same manner as an International Standard.Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.ISO/IEC TR 24772, was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*.]IntroductionThis document provides guidance for the [language] programming language so that application developers considering [language] or using [language] will be better able to avoid the programming constructs that lead to vulnerabilities in software written in the [language] 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 could lead to vulnerabilities in their software. This document can also be used in comparison with companion standards for other languages and with ISO/IEC 24772-1, to select a programming language that provides the appropriate level of confidence that anticipated problems can be avoided. This document part is intended to be used with ISO/IEC 24772-1, which discusses programming language vulnerabilities in a language independent fashion.It should be noted that 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.1 ScopeThis 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 guidance is applicable to the software developed, reviewed, or maintained for any application.Vulnerabilities described in this document the way that the vulnerability described in the language-independent writeup (in TR 24772-1) are manifested in *[language]*. 2 Normative referencesThe following referenced documents are indispensable for the application 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. [At a minimum, the current version of the language reference manual should be normatively referenced, as well as other international standards that are essential to reading this document.][This sub-clause should list the relevant language standards and other documents that describe the language treated in this Part. It need not be simply a list of standards. It should do whatever is required to describe the language that is the baseline.]3 Terms and definitions, symbols and conventions For the purposes of this document, the terms and definitions given in ISO/IEC 2382–1, in ISO/IEC 24772-1 and the following apply. Other terms are defined where they appear in *italic* type.[Follow the format of ISO/IEC 24772-1 for the specification of language-specific terminology]4 Using this document4.1. Purpose of this documentThis document is intended to be used with ISO/IEC 24772-1 to specify how programming language vulnerabilities arise in the context of programming language [*language*]. The usages specified in ISO/IEC 24772-1 apply transitively to this Part.[Issues raised in ISO/IEC 24772-1 clause 5.1 that are of particular interest to [*language*] should be restated in the terminology specific to language developers of [*language*]. In addition, general language concepts can be explained here.].This document addresses selected application vulnerabilities, which have been experienced in [*language*] or its runtime environment in the past and are likely to happen again. Each vulnerability and its possible mitigations are described in clause 6 of this document.4.2 Applying this document4.3 Structure of this document[Present the organization of the rest of the document, following the approach of ISO/IEC 24772-1 clause 4.3.] |

|  |
| --- |
| 5 *Language concepts, common guidance*5.1 Language concepts  [This sub-clause should provide an overview of general terminology and concepts of [*language*] that are utilized throughout this Part. 5.2 Top avoidance mechanismsIn addition to the generic programming rules from TR 24772-1 subclause 5.2, additional rules from this section apply specifically to the programming language [*language*]. The recommendations of this section are restatements of recommendations from clause 6 of this document, but represent ones stated frequently, or that are considered as particularly noteworthy by the authors. Clause 6 of this document contains the full set of recommendations, as well as explanations of the problems that led to the recommendations being made.[Following this statement, provide a table that provides the most common (approximately 10) or most important guidance that is not provided in TR 24772-1 subclause 5.2. The format of the table is Rule number (sequential), the rule itself, and references to subclause 6.x.2, where x contains an instance of the rule]6 Language vulnerabilities[Every vulnerability description of clause 6 of the main document should be addressed in this Part in the same order even if there is simply a notation that it is not relevant to the language in question. Each vulnerability description should have the following format:]6.x <Vulnerability name> [<3 letter tag>]6.<x>.1 Applicability to language[This section states the applicability of the vulnerability from ISO/IEC 24772-1 clause 6.X, to [*language*]. The following cases need consideration:1. If [*language*] prevents all facets of the vulnerability as described in ISO/IEC 24772-1 clause 6.X, state that the vulnerability *does not apply* and provide a sound but brief rationale for that statement. In such a case, if there is no further vulnerability, this statement may be the only statement in clause 6.X.
2. If [*language*] provides strong mitigations to help the developer avoid the vulnerability, then the statement that [language] *mitigates* the vulnerability as described in ISO/IEC 24772-1 clause 6.X can be given in 6.X.1, together with rationale for the mitigation statement and an explanation of remaining actions or behaviours needed to eliminate the vulnerability.
3. Otherwise state that the vulnerability as described in ISO/IEC 24772-1 clause 6.X *applies* to [*language*] and explain how it applies. Provide a description of additional required actions to prevent the vulnerability.
4. If the vulnerability described in ISO/IEC 24772-1 clause 6.X does not apply, except for a remaining corner case, you may use the shorter terminology “The vulnerabilities *do not apply except* …
5. Compile-time rejection or run-time checks as a means to prevent vulnerabilities qualify for the “does not apply”-case. In the case-of run-time checks, a standard reference should be added (after adjusting it to the terminology of [*language*]: “The vulnerability associated with X (e.g. unhandled exceptions) is discussed in [reference] (e.g. clause 6.36 “Ignored error status and unhandled exceptions [OYB]”).

Rules to observe:1. Ensure that each vulnerability in Part 1 is addressed by an explicit statement as described above. (Part 1 groups closely related vulnerabilities. Deal with each one.).
2. In considering a vulnerability, abstract Part 1 somewhat to address all facets of the relevant issue (Example: if Part 1 presents a vulnerability about pointers, you cannot deny the vulnerability merely because [language] does not have pointers, but instead provides references or labels that face analogous vulnerabilities).
3. In clause 6, assume that the reader knows [*language*]. If you feel that tutorial text is very important, put terminology in clause 3. and relevant descriptions in clause 5.1. An exception to this rule is the short description for a feature completely dedicated to the vulnerability at hand and of little or no consequence otherwise.
4. Keep rationales short, particularly the ones for non-applicability of a vulnerability.
5. Do not describe language features not directly related to the vulnerability at hand or not related to specific advice for its avoidance.
6. Prefer informal, but comprehensible rationale to precise, but difficult-to-understand reference manual descriptions.
7. The document is not a place to advertise [*language*] and its features. If necessary, describe them briefly and in factual terms. Subjective qualifiers common in marketing literature and subjective statements will be deleted.
8. “Programmers do not do this” is not a valid argument in this document.
9. “Software Engineering principles prevent this” is not a valid argument in this document, unless prevention is strictly enforced by the language. Formulate it as a mitigation, instead.
10. “Tool X prevents this” is not a valid argument in this document, unless use of the tool is mandatory for every program in [*language*]. Formulate tool usage as a mitigation, instead.
11. Avoid comparisons with other languages.
12. Do not cite specific products.

6.<x>.2 Guidance to language users[If the vulnerability is mitigated or if it applies, describes what the programmer or user should do in order to avoid or eliminate the vulnerability.Rules to observe:1. Formulate full sentences in the imperative tense that can be understood in isolation by experts.
2. Use gradations of the imperative to indicate the strength of the advice (from “enforce, prohibit, mandate” via “avoid, prefer” to “consider, examine“).
3. Necessary technical details supporting or explaining the advice should be given in 6.<x>.1.
4. Do not include justifications in the advice itself. ]
 |

Following the final vulnerability description, there should be sub-clauses as follows:

|  |
| --- |
| **7.<y>** Language specific vulnerabilities for [*language]*[This section is where vulnerabilities not covered by TR 24772-1 will be placed. It is possible that there are none for any given language.]8 Implications for standardization or future revision[This section provides the opportunity to discuss changes anticipated for future versions of the language specification. The section may be vacant.] |