|  |  |
| --- | --- |
|  | |
| **IBM Peer Review Process** | | |
|  |  | |

Prepared by: Regina Ackerman

IBM

July 22, 2013

**Revision History**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Document # | Date | Description | Author | Company |
| 0.1 | 07/22/2013 |  | Regina Ackerman | IBM |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# Code Checklist

The following checklist is required when reviewing code for a project.

Instructions: If the code being reviewed matches the criterion listed, check the Yes box. If it does not match, the No box is checked. If the criterion is not applicable to the document you are reviewing, the NA box is checked. For all No and NA answers, comments should be provided to indicate the reason for the response.

CR/TFS Number:

Component Name:

Review Date:

Reviewed by:

Functional Role

(SE, Dev., Tester, etc.):

| CRITERIA | RESPONSE | **COMMENTS** |
| --- | --- | --- |
| **Routines** |  |  |
| 1. Each routine’s name describes exactly what the routine does. | Yes  No NA |  |
| 1. Each routine performs one well-defined task. | Yes  No NA |  |
| **Data** |  |  |
| 1. Names are descriptive enough to help document declarations. | Yes  No NA |  |
| 1. Variables are used only for the purpose for which they are named. | Yes  No NA |  |
| 1. Variables are initialized. | Yes  No NA |  |
| 1. Named constants are used instead of magic numbers or magic strings.1 | Yes  No NA |  |
| 1. Naming conventions distinguish among types, named constants, local variables, and global variables. | Yes  No NA |  |
| **Control** |  |  |
| 1. Abstraction is easy to understand and enhances the readability of the code. Implementation details are hidden as much as possible. | Yes  No NA |  |
| 1. The flow of control through the code is straightforward and is not convoluted. All code can be exercised given the right conditions. | Yes  No NA |  |
| 1. Performance was considered when the code was written. (The code is efficient.) | Yes  No NA |  |
| **Layout** |  |  |
| 1. Code layout (formatting) shows its logical structure (e.g., indentation). | Yes  No NA |  |
| 1. The layout of each routine is clearly delineated from the others. | Yes  No NA |  |
| 1. External calls are identified. | Yes  No NA |  |
| 1. Comments are clear (concise, single interpretation and free of spelling or grammatical error). Comments specify both what the code is doing and why. | Yes  No NA |  |
| **Design** |  |  |
| 1. The code is straightforward and without cleverness that makes it obscure (KISS – keep it simple and straightforward). | Yes  No NA |  |
| 1. The code is enumerated and traces back to the Application Design, and all design goals have been satisfied. Ensure that the Traceability Tree Report (TTR) from ReqPro has all required traces from software components (Code) back to the Application Design Elements and that no hanging or orphan requirements exist.  * Ensure all code modules are adequately tagged to the correct items (i.e. No tags to entire document sections, no tags to section headings or tags to documents that do not exist in ReqPro, etc.) | Yes  No NA |  |
| 1. The Configuration Register accurately lists all code CIUs changing for the project/ release as well as documents being created/updated for the project | Yes  No NA |  |
| 1. All coding standards have been followed. | Yes  No NA |  |
| 1. The code is testable. | Yes  No NA |  |
| 1. Code reuse is employed when practical. Repetition and redundancy are avoided. | Yes  No NA |  |
| 1. The code is consistent. Conventions and units are used consistently. | Yes  No NA |  |
| 1. The code is maintainable. | Yes  No NA |  |
| 1. No security vulnerabilities are introduced in the code. | Yes  No NA |  |

*1. A magic number is a number that is hard-coded into the statements of a program whose appearance tells you nothing about its intended purpose or meaning. A magic string is a string that is hard-coded into the statements of a program whose appearance tells you nothing about its intended purpose or meaning.*