**Naming Conventions**

1. All the names are meaningful.  
2. There are no single-character variables.  
3. Every class named in the method is correctly-written.  
4. Every interface recalled in the method follows the class-naming convention.  
5. Every method’s name begins with a verb in lower-case characters.  
6. Every attribute follows the attribute’s naming conventions.  
7. Every upper-case attribute is referred to class constants declared at the beginning of the file.

**Indention**

8. Every indentation is made of four spaces.  
9. Automatic tabulation has not been used.

**Braces**

10. The chosen bracing convention is the Kernighan and Ritchie one

* 255: the open curly bracket should be moved to line 254

11. No single-line bocks not contained between curly braces are present.

**File Organization**

12. The following blank lines could be removed, because they do not separate any section:

* 276
* 283
* 292
* 300
* 305
* 307
* 311
* 313
* 318
* 320
* 325
* 330
* 332
* 339
* 345

13. Lines exceeding 80 characters that should be divided:

* 256
* 285
* 286
* 289
* 290

14. No line needs to exceed 80 characters.

**Wrapping Lines**

15. In the following lines the line break occurs before the operator, but should appear after that:

* 278
* 288
* 289
* 302
* 315
* 348

16. In the following lines, an high level break would be preferred, but in this way the lines will exceed 80 characters: an evaluation should be made in order to decide which of the two problems solve.

* 276-281
* 288-290
* 303-303
* 315-316
* 348-349
* 353-354

17. Every new statement is aligned with the beginning of the expression of the same level as the previous line.

**Comments**

18. Comments are not unnecessary, and are used only to explain some implementing choices and tricky algorithm’s pieces.  
19. The method does not contains commented out code.

**Java Source Files**

20. The file containing the method provides only one public class; no private classes are present.  
21. The public class is the first class of the file.  
22. *Common part*  
23. Javadoc is not written for this method.

**Package and Import Statements**

24. *Common part*

**Class and Interface Declarations**

25. *Common part*  
26. *Common part*  
27. There is no cloned or replicated code in this method.

**Initialization and Declarations**

28. Variables “pkLevelSequenceNum” (line 268) and “pkLastRefreshedAt” (line 269) should be declared with private visibility, due to the fact that they are used only in this method.  
29. Every variable is declared in the proper scope.  
30. Variable “robInfo” (line 260) should be initialized using the proper constructor, and after that assign the correct value.  
31. Every object is initialized before the use.  
32. Every variable is initialized where is declared.  
33. Every declaration appears at the beginning of the block in which it appears.

**Method Calls**

34. Every method is invoked with the correct order of arguments.  
35. Every invoked method is the right one to invoke.  
36. Every invoked method returns a void value, and the code satisfies this returned type.

**Arrays**

37. No array is used.  
38. No array is used.  
39. No array is used.

**Object Comparison**

40. The use of “equals” is not required: in facts all the comparisons are between “int” type variables, which is a primitive type and can use “==” (or ”!=”).

**Output Format**

41. There are no spelling/grammatical errors in the outputs.  
42. Line 289 contains a non-comprehensive output: the description of the printed variable shouldn’t be the path of the variable.  
‘ “ robInfo.lastRefreshedAt: “ ‘ should be changed into ‘ “ last refreshed at: “ ‘.  
43. In lines 280-281 there is no distinction between “beanLevelSequenceNum” and “robInfo” in the output. A separator should occur between the two outputs.

**Computation, Comparisons and Assignments**

44. The method implementation avoids brute programming.  
45. All the parenthesizing and operators precedence are in the correct order.  
46. Parenthesis are correctly used to avoid operator precedence problems.  
47. There are no denominators to check.  
48. All the integer arithmetic expressions are used appropriately.  
49. Comparisons and Boolean operators are correct.  
50. The possible exception are thrown in the invoked methods. The “finally” block ensures that in case of error the computation continues.  
51. There aren’t implicit type conversions: all the type conversion are made explicitly using a cast.

**Exceptions**

52. No exception is caught.  
53. There are no catch blocks.

**Flow Control**

54. There are no switch statements.  
55. There are no switch statements.  
56. There are no loops.

**Files**

57. There are no files used.  
58. There are no files used.  
59. There are no files used.  
60. There are no files used.