Java Code Review Checklist

Reviewer		Program		Date	

General	- Use one column for each class reviewed; enter the name of the class on the first row. - Review each code segment against the applicable category. - As you complete each review step, check off in the box at the right (✓ok, × errors).

Category	What to verify			
Headers and declarations (class design)				
Package decl.	• Name reflects intent and follows conventions			
Imports	• The correct packages are imported.			
Class header	• Describes class responsibilities and dependencies. • Has author and date			
Class declaration	• Name according to responsibilities and naming conventions (e.g., MyTable). • Class parameters (generics) are appropriately typed.			
Field declarations	• Self-explanatory or documented meaning. • Correct type and initialization (may need to check constructors). • Properly encapsulated (least visibility).			
Method headers	When not self-explanatory, short specification of: • Intent, including side effects and call constraints • Meaning of parameters and return value • Handling of exceptional conditions			
Method signatures	• Name reflects intent and follows naming conventions (e.g., getSize) • Correct parameter and return types.			
Method bodies (class implementation)				
Pre-conditions	• Call constraints (on parameters and object state) are checked. • Appropriate error messages and exceptions are produced.			
Literals	• No hard-coded/magic constants (use final static fields instead).			
Variables	• Self-explanatory (by name or usage) or documented meaning. • Declared near first use. • Correct type and initialization.			
Assignment	• Assigned values/objects conform to type/range/constraints.			
References	• Doesn't try to follow null references. • Comparison of values (with .equals) or references (with ==) as appropriate. • Copy of values or references as appropriate.			
Conditional statements	• Boolean conditions are correct. • All relevant cases are considered.			
Loop statements	• Loop header is correct (no off-by-one errors). • Loop always terminates.			
I/O	• I/O strings are correctly spelled. • Input values are appropriately checked.			
Calls	• The correct methods are used. • Parameters are of the correct type and order. • Call constraints are not violated. • Return value is used consistently with spec. • Exceptions thrown are handled or appropriately ignored. • Recursive calls terminate.			
Post-conditions	• Return values and side effects produced as specified (in method header).			
Efficiency	• Time efficiency is adequate. • Memory and other resource utilization is minimized.			
Final overall checks				
Completeness	• All class responsibilities are fulfilled.			
Maintainability	• No duplicate code.			
Traceability	• Consistent with previous-phase specs (requirements, design).			