What is Static Testing?

Static testing is a **software testing technique** that involves examining code, requirements, or documentation **without executing the program**. It focuses on:

- Finding bugs **early in development**
- Ensuring compliance with coding standards
- Improving code quality and maintainability

Static testing includes **manual activities** (like walkthroughs and reviews) and **automated tools** (like linters and analysers).

Key Activities: Walkthroughs, Reviews, and Inspections

Activity	Description	Participants	Purpose
Walkthrough	Informal review where the author leads others through the document/code	Author + peers	To get feedback and learn collectively
Technical Review	Structured discussion focusing on technical content	Reviewers (peers + experts)	To validate technical accuracy
Inspection	Formal and rigorous analysis with predefined roles and process	Moderator, author, reviewers	To find defects and improve quality

Static Analysis Tools: How They Boost Software Quality

Static analysis tools automatically examine source code or compiled code **without execution** to detect:

- Syntax errors
- Unreachable code
- Security vulnerabilities
- Potential bugs and performance issues

Benefits of Using Static Analysis Tools

- **Early Detection** of issues before testing begins
- **Reduces debugging time** and cost
- **Improves code quality** by enforcing standards
- Helps ensure **security and performance**
- Encourages consistent **documentation and readability**

Popular Static Analysis Tools & Their Features

Tool	Key Features
SonarQube	Detects bugs, code smells, security vulnerabilities
PMD	Scans for programming flaws in Java code
FindBugs/SpotBugs	Analyzes Java bytecode for bug patterns
ESLint	Finds problems in JavaScript code
Cppcheck	Detects bugs in C/C++ code
Coverity	Enterprise-grade tool for deep static analysis

.