**Static Testing**

**Static Testing** is a method of finding bugs or issues in software **without actually running the program**. Reviewing the content to ensure it’s correct.

**Analogies:**

* **Proofreading a Document**: Imagine you’ve written a report or an essay. Before you print it out, you check it for any spelling or grammar mistakes. You don’t need to read the entire report aloud to find these mistakes—you simply review the content and identify issues. This is similar to static testing where you check code, documents, or requirements for errors without executing the program.
* **Blueprint Review**: Think of designing a house. Before you build it, you’d review the blueprint to ensure the design is correct. You don’t need to start constructing the house to see if the plan is flawed. This is like static testing where you look at the "blueprint" (code or design) for potential issues before building (running the program).

**What Happens in Static Testing?**

In static testing, testers or developers review code, documents, requirements, or designs to find errors before the software is run. Static testing doesn’t involve running the program at all. It is all about reviewing and analyzing the static elements.

**Steps in Static Testing:**

1. **Review Requirements/Design**:
   * The team examines the **requirements** or **design documents** to ensure they are clear, complete, and correct. They look for misunderstandings or missing information.
2. **Code Review**:
   * Developers might **review the code** written by other developers to ensure it is written well, follows coding standards, and is free of common errors or bugs. This is like peer proofreading for code.
3. **Static Code Analysis**:
   * Tools are used to automatically **scan the code** for potential issues such as unused variables, possible security vulnerabilities, or violations of coding guidelines. It's like using a grammar checker to find mistakes in a document.
4. **Walkthroughs**:
   * A team might conduct a **walkthrough** where they go through the design or code together, discussing and identifying possible issues. This is like having a group meeting to go over a report before it is finalized.
5. **Inspection**:
   * An **inspection** is a more formal, detailed review of code or documents by a team of people, where each person focuses on different aspects of the material to identify bugs or issues. It’s like an in-depth proofreading session with multiple reviewers.

**Benefits of Static Testing:**

* **Find Errors Early**: Like proofreading a document before submitting it, static testing helps find mistakes before the code is even run, which saves time and effort later.
* **Improves Quality**: Ensures that the foundation (code, design, requirements) is solid before building or running the program.
* **Reduces Cost**: It’s easier and cheaper to fix errors in the early stages than after the software is already running.

**Example:**

Let’s say you are developing a program that checks if a number is prime .

* **Static Testing**: Before running the program, you check the code to ensure that it follows proper logic, looks clean, and follows coding standards. You might use a static analysis tool to check for mistakes or unused variables.
* **Dynamic Testing** (after static testing): Once the code is correct and ready, you’d run the program to test if it works correctly with different inputs.

Note : **Static Testing** is about finding issues early, before running the software, by carefully reviewing and analyzing code, design, and documentation. It's like proofreading your work or double-checking a blueprint—ensuring things are right before moving on to the next step!