White box testing techniques, also known as structural or code-based testing techniques, focus on examining the internal structure, design, and implementation details of the software system. Testers have access to the source code and use their knowledge of the internal workings to design test cases. Here are some common white box testing techniques:

1. Statement Coverage: Statement coverage aims to ensure that each statement in the source code is executed at least once during testing. Test cases are designed to cover different paths and execution flows within the code.

2. Branch Coverage: Branch coverage focuses on testing all possible branches or decision points within the code. Test cases are designed to ensure that each decision, such as an if-else statement or a switch case, is evaluated both to the true and false conditions.

3. Path Coverage: Path coverage aims to test all possible paths or combinations of statements and branches within the code. Test cases are designed to traverse different paths, including loops, conditional statements, and multiple branches, to achieve maximum coverage.

4. Condition Coverage: Condition coverage ensures that all possible combinations of conditions within a decision are evaluated. Test cases are designed to exercise each condition independently and in combination with others to validate their effects on decision outcomes.

5. Loop Coverage: Loop coverage focuses on testing the different behaviors and scenarios within loops. Test cases are designed to execute the loop body zero times, once, and multiple times to validate the correctness of loop constructs.

6. Data Flow Testing: Data flow testing aims to test how data is used and propagated throughout the code. Test cases are designed to cover different paths where data is defined, used, and modified, ensuring that data dependencies and potential data-related issues are identified.

7. Boundary Value Analysis: Similar to the black box technique, boundary value analysis can also be applied in white box testing. Test cases are designed to exercise the boundaries of variables and data structures within the code to verify their correctness.

8. Code Reviews: Code reviews involve manual examination of the source code by developers or peers to identify potential defects, code smells, and design flaws. Code reviews help in improving code quality, identifying vulnerabilities, and ensuring adherence to coding standards.

9. Static Analysis: Static analysis involves using specialized tools to analyze the source code without executing it. These tools identify potential issues such as coding errors, security vulnerabilities, or performance bottlenecks by analyzing the code structure, syntax, and dependencies.

10. Mutation Testing: Mutation testing is a technique that involves introducing intentional modifications or mutations into the code and running the existing test suite. The objective is to identify the effectiveness of the test cases in detecting these mutations. If the mutations are not detected, it indicates weak or missing test coverage.

These white box testing techniques provide a deep level of insight into the internal structure of the software system, enabling testers to evaluate code coverage, identify potential defects, and assess the robustness of the implementation. By combining these techniques with black box testing, comprehensive test coverage can be achieved, leading to improved software quality.