**Defects and failures**[[edit](https://en.wikipedia.org/w/index.php?title=Software_testing&action=edit&section=2" \o "Edit section: Defects and failures)]

Not all software defects are caused by coding errors. One common source of expensive defects is requirement gaps, e.g., unrecognized requirements that result in errors of omission by the program designer.[[5]](https://en.wikipedia.org/wiki/Software_testing#cite_note-kolawa-5):426 Requirement gaps can often be [non-functional requirements](https://en.wikipedia.org/wiki/Non-functional_requirements) such as [testability](https://en.wikipedia.org/wiki/Software_testability), [scalability](https://en.wikipedia.org/wiki/Scalability), [maintainability](https://en.wikipedia.org/wiki/Maintainability), [usability](https://en.wikipedia.org/wiki/Usability), [performance](https://en.wikipedia.org/wiki/Computer_performance), and [security](https://en.wikipedia.org/wiki/Computer_security).

Software faults occur through the following processes. A programmer makes an [error](https://en.wikipedia.org/wiki/Human_error) (mistake), which results in a [defect](https://en.wikipedia.org/wiki/Fault_(technology)) (fault, bug) in the software [source code](https://en.wikipedia.org/wiki/Source_code). If this defect is executed, in certain situations the system will produce wrong results, causing a [failure](https://en.wikipedia.org/wiki/Failure).[[6]](https://en.wikipedia.org/wiki/Software_testing#cite_note-ctfl-6) Not all defects will necessarily result in failures. For example, defects in [dead code](https://en.wikipedia.org/wiki/Dead_code) will never result in failures. A defect can turn into a failure when the environment is changed. Examples of these changes in environment include the software being run on a new [computer hardware](https://en.wikipedia.org/wiki/Computer_hardware) platform, alterations in [source data](https://en.wikipedia.org/wiki/Source_data), or interacting with different software.[[6]](https://en.wikipedia.org/wiki/Software_testing#cite_note-ctfl-6) A single defect may result in a wide range of failure symptoms.

**Input combinations and preconditions**[[edit](https://en.wikipedia.org/w/index.php?title=Software_testing&action=edit&section=3" \o "Edit section: Input combinations and preconditions)]

A fundamental problem with software testing is that testing under *all* combinations of inputs and preconditions (initial state) is not feasible, even with a simple product.[[4]](https://en.wikipedia.org/wiki/Software_testing#cite_note-Kaner2-4):17–18[[7]](https://en.wikipedia.org/wiki/Software_testing#cite_note-7) This means that the number of [defects](https://en.wikipedia.org/wiki/Software_bug) in a software product can be very large and defects that occur infrequently are difficult to find in testing. More significantly, [non-functional](https://en.wikipedia.org/wiki/Non-functional_requirements) dimensions of quality (how it is supposed to *be* versus what it is supposed to *do*)—[usability](https://en.wikipedia.org/wiki/Usability), [scalability](https://en.wikipedia.org/wiki/Scalability), [performance](https://en.wikipedia.org/wiki/Computer_performance), [compatibility](https://en.wikipedia.org/wiki/Backward_compatibility), [reliability](https://en.wikipedia.org/wiki/Reliability_(engineering))—can be highly subjective; something that constitutes sufficient value to one person may be intolerable to another.

Software developers can't test everything, but they can use combinatorial test design to identify the minimum number of tests needed to get the coverage they want. Combinatorial test design enables users to get greater test coverage with fewer tests. Whether they are looking for speed or test depth, they can use combinatorial test design methods to build structured variation into their test cases.[[8]](https://en.wikipedia.org/wiki/Software_testing#cite_note-8)

**Economics**[[edit](https://en.wikipedia.org/w/index.php?title=Software_testing&action=edit&section=4" \o "Edit section: Economics)]

A study conducted by [NIST](https://en.wikipedia.org/wiki/NIST) in 2002 reports that software bugs cost the U.S. economy $59.5 billion annually. More than a third of this cost could be avoided, if better software testing was performed.[[9]](https://en.wikipedia.org/wiki/Software_testing#cite_note-9)[[*dubious*](https://en.wikipedia.org/wiki/Wikipedia:Accuracy_dispute#Disputed_statement)*–*[*discuss*](https://en.wikipedia.org/wiki/Talk:Software_testing#NIST_study)]

[Outsourcing](https://en.wikipedia.org/wiki/Outsourcing) software testing because of costs is very common, with China, the Philippines and India being preferred destinations.[[10]](https://en.wikipedia.org/wiki/Software_testing#cite_note-10)

**Roles**[[edit](https://en.wikipedia.org/w/index.php?title=Software_testing&action=edit&section=5" \o "Edit section: Roles)]

Software testing can be done by dedicated software testers. Until the 1980s, the term "software tester" was used generally, but later it was also seen as a separate profession. Regarding the periods and the different goals in software testing,[[11]](https://en.wikipedia.org/wiki/Software_testing" \l "cite_note-11) different roles have been established, such as *test manager*, *test lead*, *test analyst*, *test designer*, *tester*, *automation developer*, and *test administrator*. Software testing can also be performed by non-dedicated software testers.[[12]](https://en.wikipedia.org/wiki/Software_testing#cite_note-12)