**What is Testing?**

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. In simple words, testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

According to ANSI/IEEE 1059 standard, Testing can be defined as - A process of analyzing a software item to detect the differences between existing and required conditions (that is defects/errors/bugs) and to evaluate the features of the software item.

Who does Testing?

It depends on the process and the associated stakeholders of the project(s). In the IT industry, large companies have a team with responsibilities to evaluate the developed software in context of the given requirements. Moreover, developers also conduct testing which is called **Unit Testing**. In most cases, the following professionals are involved in testing a system within their respective capacities −

* Software Tester
* Software Developer
* Project Lead/Manager
* End User

Different companies have different designations for people who test the software on the basis of their experience and knowledge such as Software Tester, Software Quality Assurance Engineer, QA Analyst, etc.

It is not possible to test the software at any time during its cycle. The next two sections state when testing should be started and when to end it during the SDLC.

When to Start Testing?

An early start to testing reduces the cost and time to rework and produce error-free software that is delivered to the client. However in Software Development Life Cycle (SDLC), testing can be started from the Requirements Gathering phase and continued till the deployment of the software.

It also depends on the development model that is being used. For example, in the Waterfall model, formal testing is conducted in the testing phase; but in the incremental model, testing is performed at the end of every increment/iteration and the whole application is tested at the end.

Testing is done in different forms at every phase of SDLC −

* During the requirement gathering phase, the analysis and verification of requirements are also considered as testing.
* Reviewing the design in the design phase with the intent to improve the design is also considered as testing.
* Testing performed by a developer on completion of the code is also categorized as testing.

When to Stop Testing?

It is difficult to determine when to stop testing, as testing is a never-ending process and no one can claim that a software is 100% tested. The following aspects are to be considered for stopping the testing process −

* Testing Deadlines
* Completion of test case execution
* Completion of functional and code coverage to a certain point
* Bug rate falls below a certain level and no high-priority bugs are identified
* Management decision

Verification & Validation

These two terms are very confusing for most people, who use them interchangeably. The following table highlights the differences between verification and validation.

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Verification** | **Validation** |
| 1 | Verification addresses the concern: "Are you building it right?" | Validation addresses the concern: "Are you building the right thing?" |
| 2 | Ensures that the software system meets all the functionality. | Ensures that the functionalities meet the intended behavior. |
| 3 | Verification takes place first and includes the checking for documentation, code, etc. | Validation occurs after verification and mainly involves the checking of the overall product. |
| 4 | Done by developers. | Done by testers. |
| 5 | It has static activities, as it includes collecting reviews, walkthroughs, and inspections to verify a software. | It has dynamic activities, as it includes executing the software against the requirements. |
| 6 | It is an objective process and no subjective decision should be needed to verify a software. | It is a subjective process and involves subjective decisions on how well a software works. |

**Software Testing**

***Software bugs*** can potentially cause monetary and human loss, history is full of such examples. To make it right, you first need to identify what’s wrong. And when it’s about finding the wrong in software, we call it “***Software Testing***”! Testers don’t break software, software is already broken. As simple as that. To tell somebody that they are wrong is called criticism. To do so officially is called ***Software Testing***

it’s the software analysis in order to find defects to be corrected. ***Software Testing*** provides customers, client, and stakeholders with information about the quality of the product or service under test – whether it meets the requirements that guided its design and development.

***“Software testing is a process used to identify the correctness, completeness, and quality of developed computer software. It includes a set of activities conducted with the intent of finding errors in software so that it could be corrected before the product is released to the end users.”***

***Software testing*** is primarily a broad process that is composed of several interlinked processes – verifying software completeness in regards to functional/business requirements, identifying technical defects and assessing software usability, performance, security, localization, compatibility, installation, etc. As the number of possible tests for even simple software components is practically infinite, all ***software testing*** uses some strategy to select tests that are feasible for the available time and resources.

**Quality Assurance**

What if you follow the best practices during design & implementation of a software, in order to prevent defects once it is released in production? That’s called ‘***Software Quality Assurance***’ – a set of administrative and procedural activities (e.g. process implementation, training, auditing, etc.) implemented in ***software engineering processes*** so that requirements and goals for a software will be fulfilled. This ***defect prevention****in quality assurance* differs subtly from defect detection and rejection in ***quality control***, and has been referred to as a *shift left* as it focuses on quality earlier in the process. A ***quality assurance system*** is said to increase customer confidence and a company’s credibility, to improve work processes and efficiency, and to enable a company to better compete with others.

The ***quality management system*** under which the software system is created is normally based on one or more of the following models/standards:

* CMMI
* Six Sigma
* ISO 9000

Note: There are many other models/standards for quality management but the ones mentioned above are the most popular.

***Fun Fact***: Quality assurance was initially introduced in World War II when munitions were inspected and tested for defects after they were made. Today’s quality assurance systems emphasize catching defects before they get into the final product.

***Software Testing is an infinite process of comparing the invisible to the ambiguous in order to avoid the unthinkable happening to the anonymous.***Say you want to transfer 10K INR to your friend via Net banking. You make the transaction & inform your friend. But when you check your SMS, it says 11K INR has been debited from your account. How do you feel? Frustrated? Angry? Resolution to change your Bank? I.e. if you don’t like testing your product, most likely your customers won’t like to test it either.