**TESTING**

**Definition:** Testing is the process of executing a program to find errors. To make our software perform well it should be error-free. If testing is done successfully it will remove all the errors from the software.

BUG: it is a error code.it may causes defect, fault and flaw.

ERROR: An Error is **a mistake made in the code**; that's why we cannot execute or compile code.

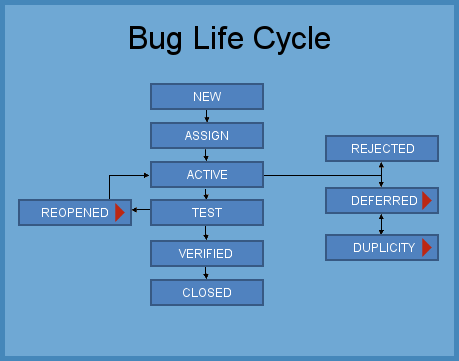
Fault: Software fault is also known as defect, arises when the expected result don't match with the actual results

Failure: A failure is the inability of a software system or component to perform its required functions within specified performance requirements. When a defect reaches the end customer it is called a Failure.

Mistake: A mistake in coding is called Error

Defect: A defect is a system error that doesn't allow the intended action to be completed.

**BUG LIFE CYCLE:**



**TESTING LEVELS:**

**4 Levels of Testing**

There are mainly four **Levels of Testing** in software testing :

1. **Unit Testing**  **(Component Testing)**
2. **Integration Testing**
3. **System Testing**
4. **Acceptance Testing**

## Each Testing Level Details

#### **1.Unit testing:**

A [Unit](https://www.guru99.com/unit-testing-guide.html) is a smallest testable portion of system or application which can be compiled, liked, loaded, and executed. **This kind of testing helps to test each module separately.**

The aim is to test each part of the software by separating it. It checks that component are fulfilling functionalities or not. **This kind of testing is performed by developers.**

#### **2.Integration testing**:

[Integration](https://www.guru99.com/integration-testing.html) means combining. For Example, In this testing phase, different software modules are combined and tested as a group to make sure that integrated system is ready for system testing.

Integrating testing checks the data flow from one module to other modules. **This kind of testing is performed by testers.**

#### **3.System Testing:**

[System testing](https://www.guru99.com/system-testing.html) is performed on a complete, integrated system. It allows checking system’s compliance as per the requirements. It tests the overall interaction of components. **It involves load, performance, reliability and security testing.**

System testing most often the final test to verify that the system meets the specification**. It evaluates both functional and non-functional need for the testing.**

#### **4.Acceptance testing:**

[Acceptance testin](https://www.guru99.com/user-acceptance-testing.html)g is a test conducted to find if the requirements of a specification or contract are met as per its delivery. **Acceptance testing is basically done by the user or customer.** However, other stockholders can be involved in this process.

## Conclusion:

* A level of software testing is a process where every unit or component of a software/system is tested.
* The primary goal of system testing is to evaluate the system’s compliance with the specified needs.
* In Software Engineering, four main levels of testing are Unit Testing, Integration Testing, System Testing and Acceptance Testing.

**TESTING PRINCIPLES:**

**There are seven principles in software testing:**

1. **Testing shows the presence of defects**
2. **Exhaustive testing is not possible**
3. **Early testing**
4. **Defect clustering**
5. **Pesticide paradox**
6. **Testing is context-dependent**
7. **Absence of errors fallacy**

* **Testing shows the presence of defects:** The goal of software testing is to make the software fail**. Software testing reduces the presence of defects.** Software testing talks about the presence of defects and doesn’t talk about the absence of defects. Software testing can ensure that defects are present but it can not prove that software is defect-free. Even multiple testing can never ensure that software is 100% bug-free. Testing can reduce the number of defects but not remove all defects.
* **Exhaustive testing is not possible**: It is the process of testing the functionality of the software in all possible inputs (valid or invalid) and pre-conditions is known as exhaustive testing**. Exhaustive testing is impossible means the software can never test at every test case**. It can test only some test cases and assume that the software is correct and it will produce the correct output in every test case. If the software will test every test case then it will take more cost, effort, etc., which is impractical.
* **Early Testing**: To find the defect in the software, early test activity shall be started. The defect detected in the early phases of SDLC will be very less expensive. For better performance of software, software testing will start at the initial phase i.e. testing will perform at the requirement analysis phase.
* **Defect clustering**: **In a project, a small number of modules can contain most of the defects.** Pareto Principle to software testing state that 80% of software defect comes from 20% of modules.
* **Pesticide paradox**: **Repeating the same test cases, again and again, will not find new bugs.** So it is necessary to review the test cases and add or update test cases to find new bugs.
* **Testing is context-dependent**: **The testing approach depends on the context of the software developed**. Different types of software need to perform different types of testing. For example, The testing of the e-commerce site is different from the testing of the Android application.
* **Absence of errors fallacy**: If a built software is 99% bug-free but it does not follow the user requirement then it is unusable. It is not only necessary that software is 99% bug-free but it is also mandatory to fulfill all the customer requirements.