Contents

[Introduction 4](#_Toc103964663)

[There are two kinds of quality: 4](#_Toc103964664)

[Lecture # 5](#_Toc103964665)

[Software Quality Assurance has: 5](#_Toc103964666)

[Major Software Quality Assurance Activities: 5](#_Toc103964667)

[2. Set The Check Points: 5](#_Toc103964668)

[3. Multi testing Strategy: 5](#_Toc103964669)

[4. Measure Change Impact: 5](#_Toc103964670)

[5. Manage Good Relations: 6](#_Toc103964671)

[Benefits of Software Quality Assurance (SQA): 6](#_Toc103964672)

[Disadvantage of SQA: 6](#_Toc103964673)

[Lecture # Types of Software Testing 7](#_Toc103964674)

[Introduction:- 7](#_Toc103964675)

[Principles of Testing:- 7](#_Toc103964676)

[Types of Testing:- 7](#_Toc103964677)

[Example: 7](#_Toc103964678)

[2. Integration Testing 7](#_Toc103964679)

[Examples: 8](#_Toc103964680)

[3. Regression Testing 8](#_Toc103964681)

[4. Smoke Testing 8](#_Toc103964682)

[5. Alpha Testing 8](#_Toc103964683)

[6. Beta Testing 8](#_Toc103964684)

[7. System Testing 8](#_Toc103964685)

[8. Stress Testing 9](#_Toc103964686)

[Examples: 9](#_Toc103964687)

[9. Performance Testing 9](#_Toc103964688)

[10. Object-Oriented Testing 9](#_Toc103964689)

[This testing is done in the following manner: 9](#_Toc103964690)

[Lecture # Software Testing | Basics 10](#_Toc103964691)

[Software testing can be divided into two steps: 10](#_Toc103964692)

[What are different types of software testing? 10](#_Toc103964693)

[1. Manual Testing: 10](#_Toc103964694)

[2. Automation Testing: 11](#_Toc103964695)

[What are the different techniques of Software Testing? 11](#_Toc103964696)

[What are different levels of software testing? 12](#_Toc103964697)

[1. Unit Testing: 12](#_Toc103964698)

[2. Integration Testing: 12](#_Toc103964699)

[3. System Testing: 12](#_Toc103964700)

[4. Acceptance Testing: 12](#_Toc103964701)

[There are seven principles in software testing: 13](#_Toc103964702)

[1. Testing shows the presence of defects: 13](#_Toc103964703)

[2. Exhaustive testing is not possible: 13](#_Toc103964704)

[3. Early Testing: 13](#_Toc103964705)

[4. Defect clustering: 14](#_Toc103964706)

[5. Pesticide paradox: 14](#_Toc103964707)

[6. Testing is context-dependent: 14](#_Toc103964708)

[7. Absence of errors fallacy: 14](#_Toc103964709)

[Power point slides data 14](#_Toc103964710)

[*Software Quality* 14](#_Toc103964711)

[ Excellence, Superiority, High Value 14](#_Toc103964712)

[ “How good something is?” 14](#_Toc103964713)

[ “Conformance to Requirements” 14](#_Toc103964714)

[Assurance 14](#_Toc103964715)

[ To guarantee that the software is of High Quality 14](#_Toc103964716)

[Software Quality Areas: 14](#_Toc103964717)

[1. Requirements 14](#_Toc103964718)

[2. Design 14](#_Toc103964719)

[3. Implementation / Coding 14](#_Toc103964720)

[4. Testing 14](#_Toc103964721)

[5. User Manuals 14](#_Toc103964722)

[6. Maintenance 14](#_Toc103964723)

[Definitions of Software Quality: 15](#_Toc103964724)

[Why Software Quality? 15](#_Toc103964725)

[Software Defects 15](#_Toc103964726)

[Six Software Defect Origins: 15](#_Toc103964727)

[Categories of Software Defects 16](#_Toc103964728)

[Software Defect Elimination Strategies 16](#_Toc103964729)

[Software Defect Elimination Techniques: 16](#_Toc103964730)

[Non-Testing Defect Removal Techniques 16](#_Toc103964731)

[Inspection 17](#_Toc103964732)

[Testing Defect Removal Techniques 17](#_Toc103964733)

[The V-model of the Systems Engineering Process 18](#_Toc103964734)

a [18](#_Toc103964735)

[Defect Removal Efficiency 18](#_Toc103964736)

[18](#_Toc103964737)

[Defect Prevention Methods 18](#_Toc103964738)

[Quality Attributes Set 19](#_Toc103964739)

[• Organization-Specific Quality Attributes 19](#_Toc103964740)

[Quality Assurance Organizations 19](#_Toc103964741)

[Quality Measurement Methods 19](#_Toc103964742)

[Root Causes of Poor Software Quality 20](#_Toc103964743)

[Defect Seeding 20](#_Toc103964744)

[Defect Severity Levels 20](#_Toc103964745)

[Test Cases links: 20](#_Toc103964746)

# Introduction

**Quality** is defined as a characteristics or attributes of something where as attributes refer to measurable characteristics-things that we are able to compare to known standards.

## There are two kinds of quality:

* **Quality of design** refers to characteristics that designer’s specify for an item. The grade of materials, tolerances and performance specifications all contribute to the quality of design.
* **Quality of conformance** is the degree to which the design specifications are followed during manufacturing. Greater the degree of conformance, the higher is the level of quality of conformance.

Software quality is defined as conformance to explicitly stated functional and performance requirements, explicitly documented development standards, and implicit characteristics that are expected of all professionally developed software.

In software development, quality of design encompasses requirements, specifications, and the design of the system where as quality of conformance is an issue focused primarily on implementation.

If the implementation follows the design and the resulting system meets its requirements and performance goals, conformance quality is high.

# Lecture #

**Software Quality Assurance (SQA)** is simple a way to assure quality in the software. It is the set of activities which ensure processes, procedures as well as standards are suitable for the project and implemented correctly.

Software Quality Assurance is a process which works parallel to development of software. It focuses on improving the process of development of software so that problems can be prevented before they become a major issue. Software Quality Assurance is a kind of Umbrella activity that is applied throughout the software process.

## Software Quality Assurance has:

1. A quality management approach
2. Formal technical reviews
3. Multi testing strategy
4. Effective software engineering technology
5. Measurement and reporting mechanism

## Major Software Quality Assurance Activities:

1. **SQA Management Plan:**

Make a plan for how you will carry out the SQA throughout the project. Think about which set of software engineering activities are the best for project. Check level of SQA team skills.

## Set The Check Points:

SQA team should set checkpoints. Evaluate the performance of the project on the basis of collected data on different check points.

## Multi testing Strategy:

Do not depend on a single testing approach. When you have a lot of testing approaches available use them.

## Measure Change Impact:

The changes for making the correction of an error sometimes re introduces more errors keep the measure of impact of change on project. Reset the new change to change check the compatibility of this fix with whole project.

## Manage Good Relations:

In the working environment managing good relations with other teams involved in the project development is mandatory. Bad relation of SQA team with programmer’s team will impact directly and badly on project. Don’t play politics.

## Benefits of Software Quality Assurance (SQA):

1. SQA produces high quality software.
2. High quality application saves time and cost.
3. SQA is beneficial for better reliability.
4. SQA is beneficial in the condition of no maintenance for a long time.
5. High quality commercial software increase market share of company.
6. Improving the process of creating software.
7. Improves the quality of the software.

## Disadvantage of SQA:

There are a number of disadvantages of quality assurance. Some of them include adding more resources, employing more workers to help maintain quality and so much more.

# Lecture # Types of Software Testing

## Introduction: -

Testing is the process of executing a program with the aim of finding 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.

## Principles of Testing: -

* All the test should meet the customer requirements
* To make our software testing should be performed by a third party
* Exhaustive testing is not possible. As we need the optimal amount of testing based on the risk assessment of the application.
* All the test to be conducted should be planned before implementing it
* It follows the Pareto rule (80/20 rule) which states that 80% of errors come from 20% of program components.
* Start testing with small parts and extend it to large parts.

## Types of Testing:-

1. **Unit Testing**

It focuses on the smallest unit of software design. In this, we test an individual unit or group of interrelated units. It is often done by the programmer by using sample input and observing its corresponding outputs**.**

## Example:

1. In a program we are checking if loop, method or Function is working fine
2. Misunderstood or incorrect, arithmetic precedence.
3. Incorrect initialization.

## Integration Testing

The objective is to take unit tested components and build a program structure that has been dictated by design. Integration testing is testing in which a group of components is combined to produce output.

Integration testing is of four types:

(i) Top-down (ii) Bottom-up (iii) Sandwich (iv) Big-Bang

## Examples:

1. **Black Box testing: -** It is used for validation. In this we ignore internal working mechanism and Focused on what is the output?
2. **White Box testing: -** It is used for verification. In this we focus on internal mechanism i.e. how the output is achieved?

## Regression Testing

Every time a new module is added leads to changes in the program. This type of testing makes sure that the whole component works properly even after adding components to the complete program.

**Example :** In school record suppose we have module staff, students and finance combining these modules and checking if on integration these module works fine is regression testing.

## Smoke Testing

This test is done to make sure that software under testing is ready or stable for further testing It is called a smoke test as the testing an initial pass is done to check if it did not catch the fire or smoke in the initial switch on.

**Example:** If project has 2 modules so before going to module make sure that module 1 works properly.

## Alpha Testing

This is a type of validation testing. It is a type of acceptance testing which is done before the product is released to customers. It is typically done by QA people.

**Example**: When software testing is performed internally within the organization.

## Beta Testing

The beta test is conducted at one or more customer sites by the end-user of the software. This version is released for a limited number of users for testing in a real-time environment.

**Example**: When software testing is performed for the limited number of people.

## System Testing

This software is tested such that it works fine for the different operating systems. It is covered under the black box testing technique. In this, we just focus on the required input and output without focusing on internal working.

In this, we have security testing, recovery testing, stress testing, and performance testing.

**Example:** This include functional as well as non-functional testing.

## Stress Testing

In this, we give unfavorable conditions to the system and check how they perform in those conditions.

## Examples:

1. Test cases that require maximum memory or other resources are executed.
2. Test cases that may cause thrashing in a virtual operating system.
3. Test cases that may cause excessive disk requirement.

## Performance Testing

It is designed to test the run-time performance of software within the context of an integrated system. It is used to test the speed and effectiveness of the program. It is also called load testing. In it we check, what the performance of the system is in the given load.

**Example**: Checking number of processor cycles.

## Object-Oriented Testing

This testing is a combination of various testing techniques that help to verify and validate object-oriented software.

## This testing is done in the following manner:

* Testing of Requirements
* Design and Analysis of Testing
* Testing of Code
* Integration testing
* System testing
* User Testing

# Lecture # Software Testing | Basics

Software testing can be stated as the process of verifying and validating that software or application is bug-free, meets the technical requirements as guided by its design and development, and meets the user requirements effectively and efficiently with handling all the exceptional and boundary cases.

The process of software testing aims not only at finding faults in the existing software but also at finding measures to improve the software in terms of efficiency, accuracy, and usability. It mainly aims at measuring the specification, functionality, and performance of a software program or application.

## Software testing can be divided into two steps:

1. **Verification:** it refers to the set of tasks that ensure that software correctly implements a specific function.
2. **Validation:** it refers to a different set of tasks that ensure that the software that has been built is traceable to customer requirements.

Verification: “Are we building the product right?” Validation: “Are we building the right product?”

## What are different types of software testing?

Software Testing can be broadly classified into two types:

## Manual Testing:

Manual testing includes testing software manually, i.e., without using any automated tool or any script. In this type, the tester takes over the role of an end- user and tests the software to identify any unexpected behavior or bug. There are different stages for manual testing such as unit testing, integration testing, system testing, and user acceptance testing.

Testers use test plans, test cases, or test scenarios to test software to ensure the completeness of testing. Manual testing also includes exploratory testing, as testers explore the software to identify errors in it.

## Automation Testing:

Automation testing, which is also known as Test Automation, is when the tester writes scripts and uses another software to test the product. This process involves the automation of a manual process. Automation Testing is used to re-run the test scenarios that were performed manually, quickly, and repeatedly.

Apart from regression testing, automation testing is also used to test the application from a load, performance, and stress point of view. It increases the test coverage, improves accuracy, and saves time and money in comparison to manual testing.

## What are the different techniques of Software Testing?

Software techniques can be majorly classified into two categories:

1. **Black Box Testing:** The technique of testing in which the tester doesn’t have access to the source code of the software and is conducted at the software interface without concern with the internal logical structure of the software is known as black-box testing.
2. **White-Box Testing:** The technique of testing in which the tester is aware of the internal workings of the product, has access to its source code, and is conducted by making sure that all internal operations are performed according to the specifications is known as white box testing.

|  |  |
| --- | --- |
| Black Box Testing | White Box Testing |
| Internal workings of an application are not required. | Knowledge of the internal workings is a must. |
| Also known as closed box/data- driven testing. | Also known as clear box/structural testing. |
| End users, testers, and developers. | Normally done by testers and developers. |
| This can only be done by a trial and error method. | Data domains and internal boundaries can be better tested. |

## What are different levels of software testing?

Software level testing can be majorly classified into 4 levels:

## Unit Testing:

A level of the software testing process where individual units/components of a software/system are tested. The purpose is to validate that each unit of the software performs as designed.

## Integration Testing:

A level of the software testing process where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units.

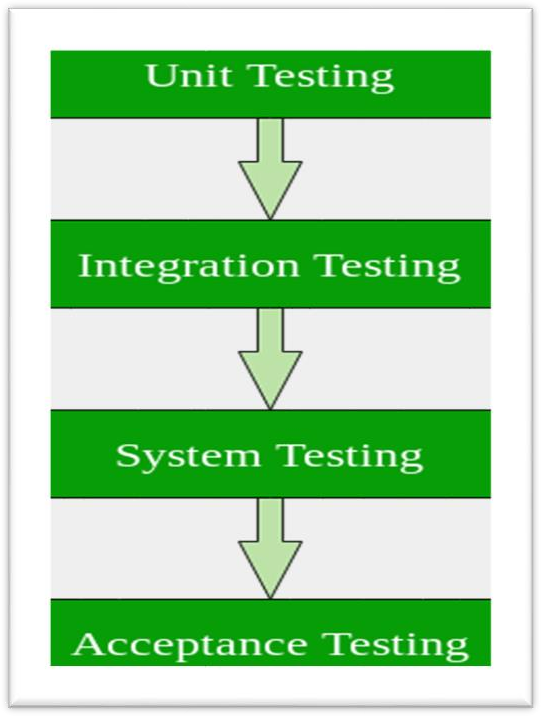
## System Testing:

A level of the software testing process where a complete, integrated

system/software is tested. The purpose of this test is to evaluate the system’s compliance with the specified requirements.

## Acceptance Testing:

A level of the software testing process where a system is tested for acceptability. The purpose of this test is to evaluate the system’s compliance with the business requirements and assess whether it is acceptable for delivery.



**Lecture # Seven Principles of software testing**

Software testing is the process of executing a program with the aim of finding the error. 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.

## 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 cannot 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.

# Power point slides data

# *Software Quality*

# Excellence, Superiority, High Value

# “How good something is?”

# “Conformance to Requirements”

# Assurance

# To guarantee that the software is of High Quality

# Software Quality Areas:

# Requirements

# Design

# Implementation / Coding

# Testing

# User Manuals

# Maintenance

# Definitions of Software Quality:

1. **Low levels of defects** when deployed, ideally zero
2. **High level of reliability**, so as running without crashes or strange results
3. A **majority of highly satisfied clients** when surveyed
4. A **structure that can minimize “bad fixes”** or insertion of new defects during repairs
5. **Effective customer support** when problems occur
6. **Rapid repairs for defects**, especially for high severity defects

# Why Software Quality?

* **Minimizes scrap and rework** expenses
* **Reduces time to market** new products
* **Enhances market share** compared to direct competitors
* **Attracts venture capitalists**
* **Minimizes the risk of litigation**
* **Minimizes the risk of operating failures**
* **Minimizes the risk of business failures**

# Software Defects

“Errors, problems or issues that stop the user to use the software effectively are the software defects”

# Six Software Defect Origins:

1. Errors in Requirements
2. Errors in Design
3. Errors in Source code
4. Errors in Test Cases
5. Errors in User Manuals
6. Errors due to “Bad fixes”

**Bad Fixes:** Mistakes made during repairs

# Categories of Software Defects

1. Errors of **commission**
   * something is ***done***, but that is wrong
   * Y2K Issue
2. Errors of **omission**
   * something ***left out*** by accident
   * lack of domain knowledge
3. Errors of **clarity** & **ambiguity**
   * different interpretations
   * روکو مت جانے دو
4. Errors of **speed** & **capacity**

# Software Defect Elimination Strategies

1. Predict Defects
2. Prevent Defects
3. Remove Defects
4. Track Defects during development
5. Useful quality measurements
6. Ensuring high levels of user-satisfaction

# Software Defect Elimination Techniques:

* Defect **Prevention**
* Defect **Removal**
  + Non-Testing Defect Removal Techniques
  + Testing Defect Removal Techniques

# Non-Testing Defect Removal Techniques

* Requirement inspections
* Design inspections
* Code inspections
* Test-case inspections
* Usability review
* *Validation: Are we building the right system?*
* *Verification: Are we building the system right?*
* *In other words, validation is concerned with checking that the system will meet the customer’s actual needs, while verification is concerned with whether the system is well-engineered, error-free, and so on.*

# Inspection

* Inspection is a **formal** & **manual** procedure of **examining** software artifacts for finding errors.
* Inspection process has:
  + **Number** of inspectors
  + **Role** of inspectors
  + **Defects** found

# Testing Defect Removal Techniques

* **Unit Testing**
  + Individual program units, such as procedure, methods in isolation
* **Function Testing**
  + Black-box testing
* **Regression Testing**
  + To uncover new software bugs in existing functional of a system after changes
* **Performance Testing**
* **Integration Testing**
  + Modules are assembled to construct larger subsystem and tested

### System Testing

* + Includes wide area of testing such as functionality, and load
* **Acceptance Testing**
  + Customer’s expectations meet from the system

# The V-model of the Systems Engineering Process

# 

# Defect Removal Efficiency

# 

**DRE =**

# Defect Prevention Methods

* Prototyping
* JAD (Joint Application Development)
* QFD (Quality Function Deployment)
* Reuse of Certified components
* Change-control Boards
* Configuration Control Tools

# Quality Attributes Set

* **Product-Specific** Quality Attributes
  + Ease of Use
  + Documentation
  + Defect Tolerance
  + Defect Frequency
  + Defect Impact
  + Packaging
  + Price versus Reliability
  + Performance

# Organization-Specific Quality Attributes

* + Service and Support

# Quality Assurance Organizations

* **No Quality Assurance**
  + **No staff** for Quality Assurance
* **Token Quality Assurance**
  + **One manager** with *job title* at least of Quality Assurance Manager
* **Passive Quality Assurance**
  + QA team size **1% to 2% of development team**
* **Active Quality Assurance**
  + QA team size **5% of development team**

# Quality Measurement Methods

* **Complexity** measurements
* **Cost per defect** measurements
* **Defects per function point** measurements
  + A **function point** is a unit of measurement to express the amount of business functionality.
    - **Input :** Form Size
    - **Output :** Screen Size
    - **Logical files :** tables, text files
* **Defects per KLOC** measurements
* **Defect aging** measurements
* **Defect severity** measurements

# Root Causes of Poor Software Quality

* Inadequate **training** of managers and staff
* Inadequate **defect measurement**
* Insufficient **defect removal**
* Excessive **schedule pressure**
* High **complexity** levels
* **Ambiguous and creeping requirements** and design

# Defect Seeding

* Defect seeding is the process of introducing known defect in the build which is ready to go for testing
* **Purpose** of defect seeding is to determine whether software is properly tested or not?

# Defect Severity Levels

* Severity level 1: **total failure** of application
* Severity level 2: failure of **major function(s)**
* Severity level 3: **minor problem**
* Severity level 4: **cosmetic problem**

# Test Cases links:

Topic Test Case:

<https://www.guru99.com/test-case.html>

*Different Test cases Examples:*

<https://www.loginradius.com/blog/engineering/test-cases-for-registration-and-login-page/>

------------------------

<https://www.softwaretestinggenius.com/four-test-cases-frequently-discussed-in-interviews-on-software-testing/>

<https://www.qaacharya.in/2021/12/test-cases-for-add-to-cart-button.html>

---------------

<https://www.zyxware.com/articles/3762/what-are-the-test-cases-to-be-considered-while-testing-the-print-out-of-a-webpage>

------------------------

<https://www.qaacharya.in/2019/11/test-cases-scenarios-for-captcha.html>

<https://www.qaacharya.in/2019/10/test-cases-scenarios-for-search-box.html>

**End of Mid Term Course**

--------------------------------------------------------------------------------------------------------------