رواد مصر الرقمية

وزارة الاتصـــــالات
وتكنولوجيا المعلومات

Chapter One:
Fundamentals Of Testing

❑ What is Testing?

Software systems are an integral part of our daily life.

Software testing assesses software quality and helps reducing the risk of software failure in operation.

Software testing is a set of activities to discover defects and evaluate the quality of software artifacts “test objects”.

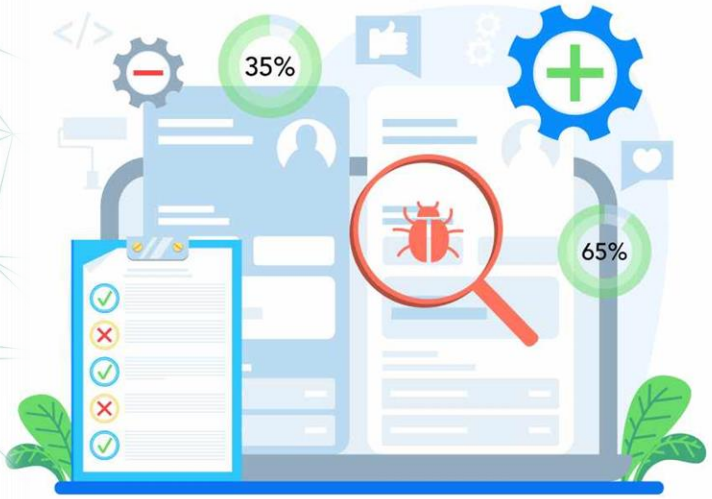
❑ What can Happen if system fails?

- Loss of your job
- Injury or even Death
- Loss of Money
- Loss of Reputation
- Loss of time
- User Inconvenience

❑ Verification vs Validation

Verification: checking whether the system meets specified requirements.
"Are we building the product, right?"

Validation: checking whether the system meets users' and stakeholders' needs.
"Are we building the right product?"



Verification

Validation

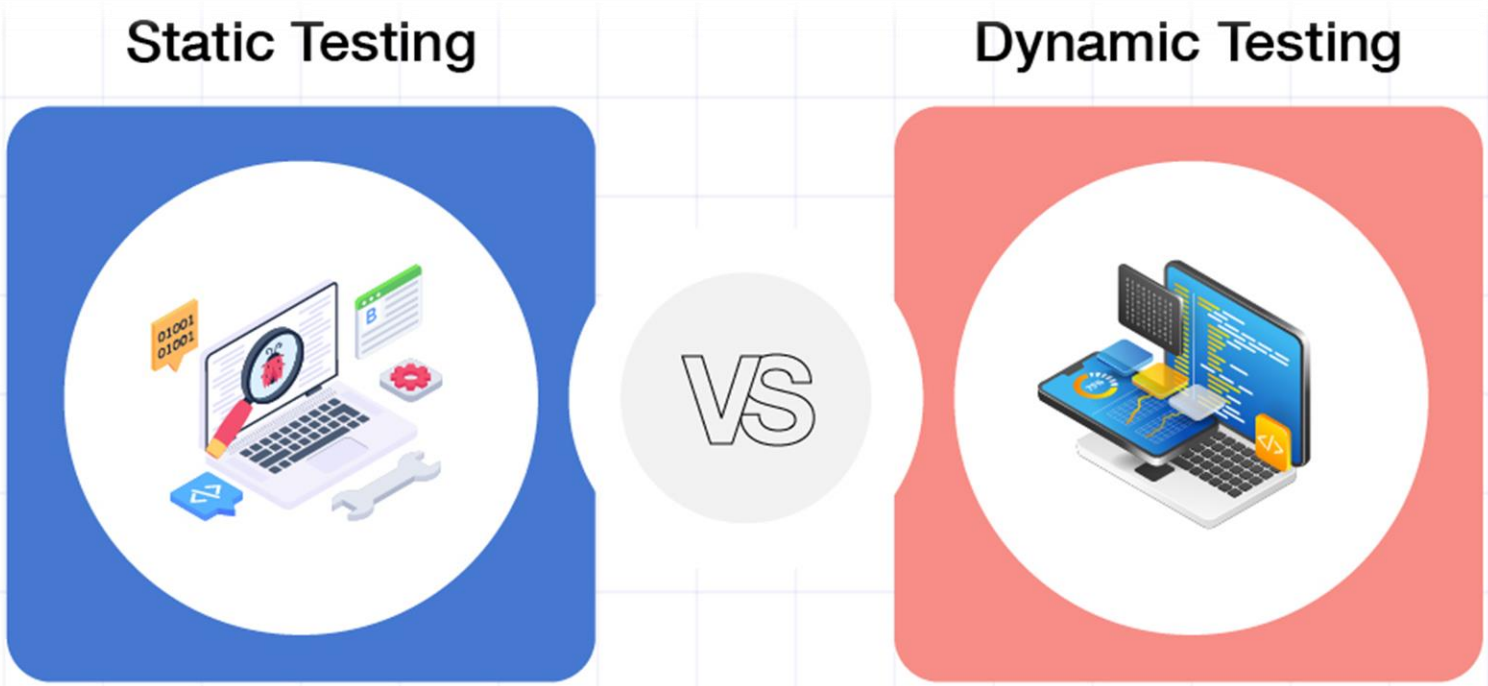
❑ Static vs Dynamic

Static testing: is done without the execution of the software.

<includes review and examine work products>

Dynamic testing: involves the execution of the software.

<uses different types of test techniques and approaches>



Testing Objectives

Evaluating work products, such as requirements

Triggering failures and finding defects

Ensuring required coverage of a test object

Reducing the level of risk of inadequate software quality

Verifying whether specified requirements have been fulfilled

Verifying that test object complies with contractual, legal, and requirement

Provide information to stakeholders to allow them to make decisions

Building confidence in the quality of the test object

Validating whether the test object is complete and works as expected

Testing vs Debugging

Testing can trigger failures that are caused by defects.

Debugging is concerned with finding causes of this failure (defects), analyzing these causes, and eliminating them.

- Reproduction of a failure
- Diagnosis (finding the root cause)
- Fixing the cause

❑ Why is Testing Necessary?

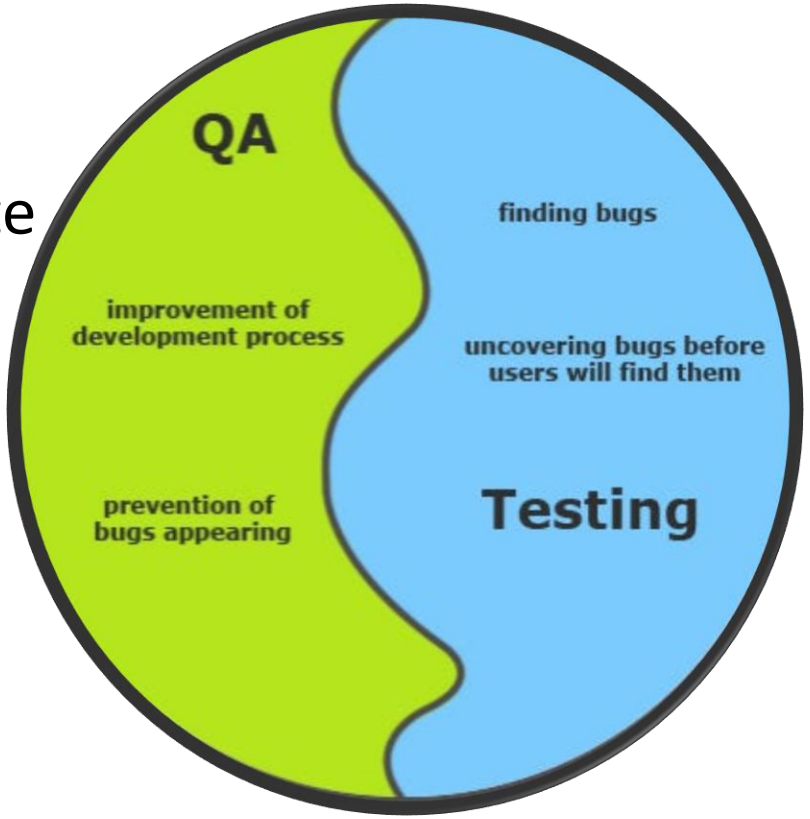
- Testing indirectly contributes to higher quality test objects.
- Evaluating the quality of a test object at various phases.
- Contributing to decisions to move to the next stage of the SDLC, such as the release decision.
- Testing provides users with indirect representation on the development project. Testers ensure that their understanding of users' needs are considered.

Quality Management

Testing vs Quality Assurance

Quality Control - QC

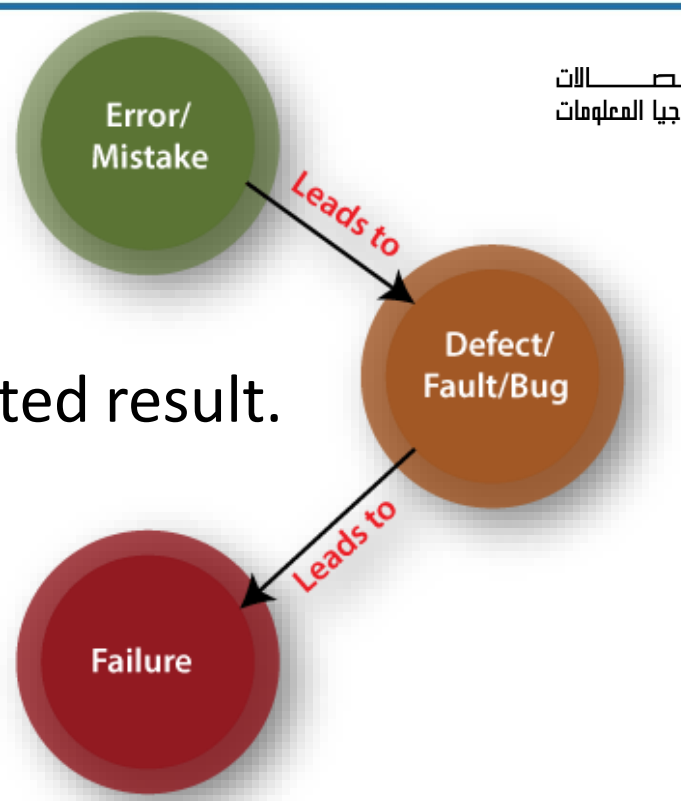
- Product-oriented.
- Corrective approach.
- Activities supporting the achievement of appropriate levels of quality.
- Testing is a major form of quality control.



Quality Assurance - QA

- Process-oriented.
- Preventive approach.
- Implementation and improvement of processes.
- QA applies to both development and testing processes.

Error vs Defect vs Failure



Error: A code mistake.

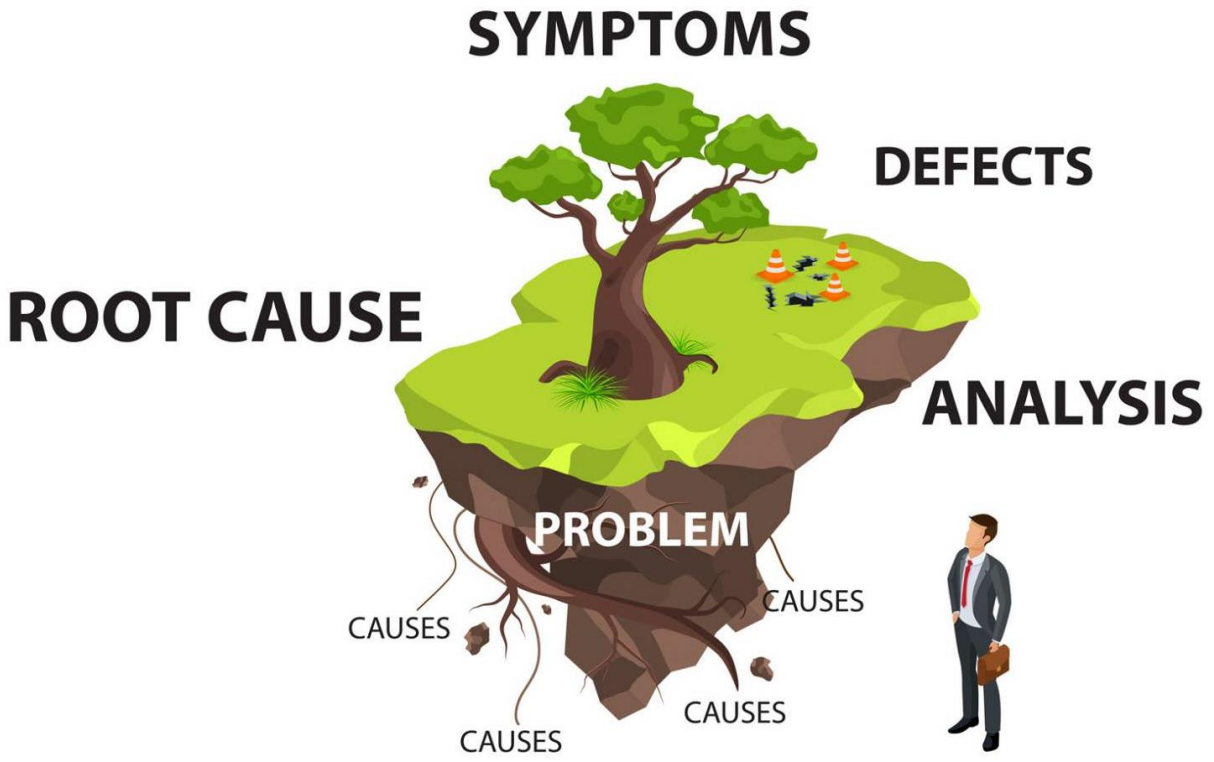
Bug/Defect/Fault: A variation between actual and expected result.

Failure: End-user finds any issue.

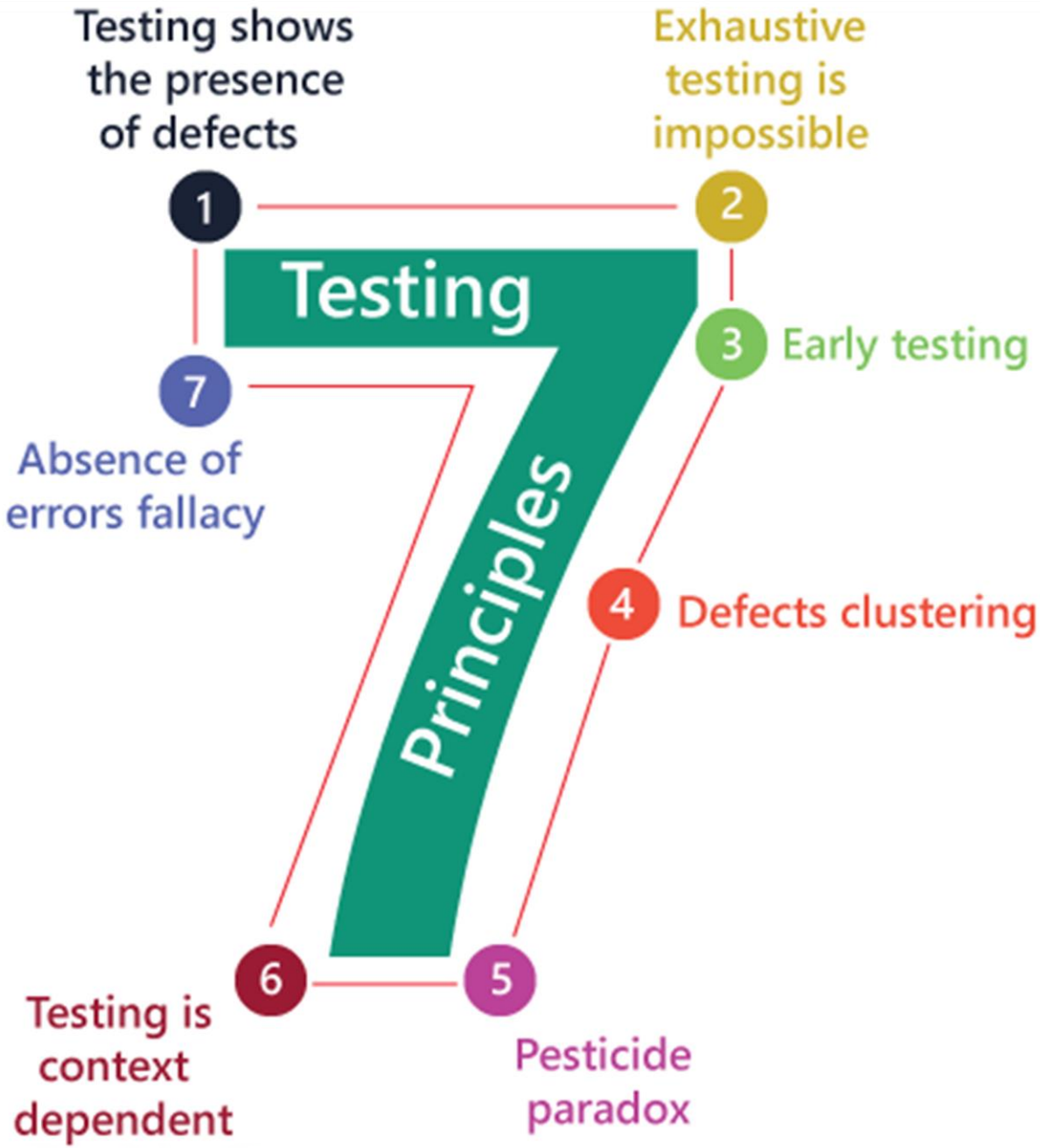
- Human beings make errors, which produce defects, which in turn may result in failures.
- Defects in artifacts produced earlier, if undetected, often lead to defective artifacts later.
- Defect in code is executed, the system may fail to do what it should do.
- Defect shouldn’t, causing a failure.

❑ Root cause analysis

- **Root cause** is a fundamental reason for the occurrence of a problem.
- **Root causes** are identified through root cause analysis which is typically performed when a failure happens or a defect is identified.



Testing Principles



1. Testing shows the presence of defects, not the absence

- Testing can show that defects are present in the test object but cannot prove that there are no defects.
- Testing reduces the probability of defects remaining undiscovered in the test object.
- If no defects are found, testing cannot prove test object correctness.

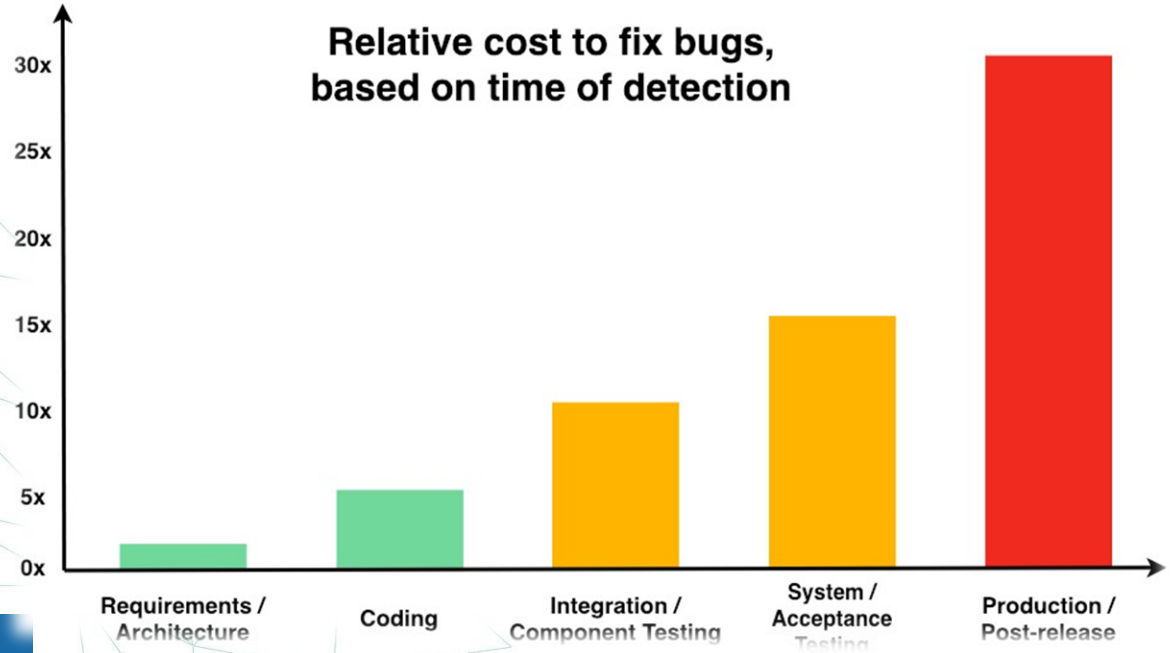
2. Exhaustive testing is impossible

- Testing everything is not feasible except in trivial cases.

- Test case prioritization, and risk-based testing should be used to focus test efforts.

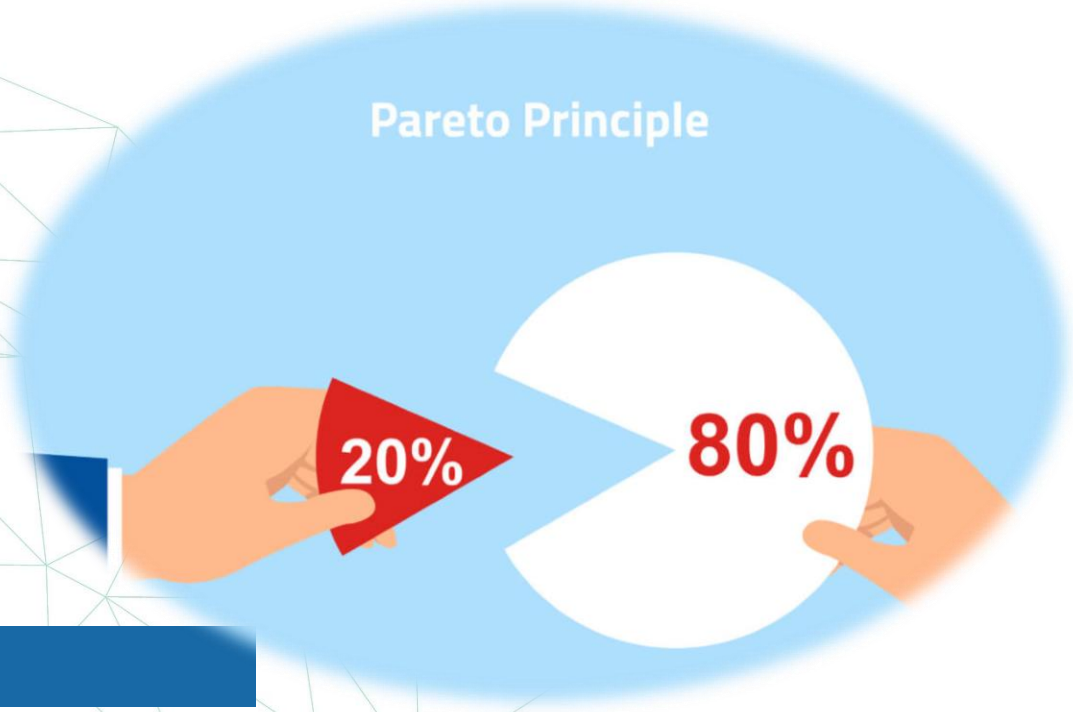
3. Early testing saves time and money

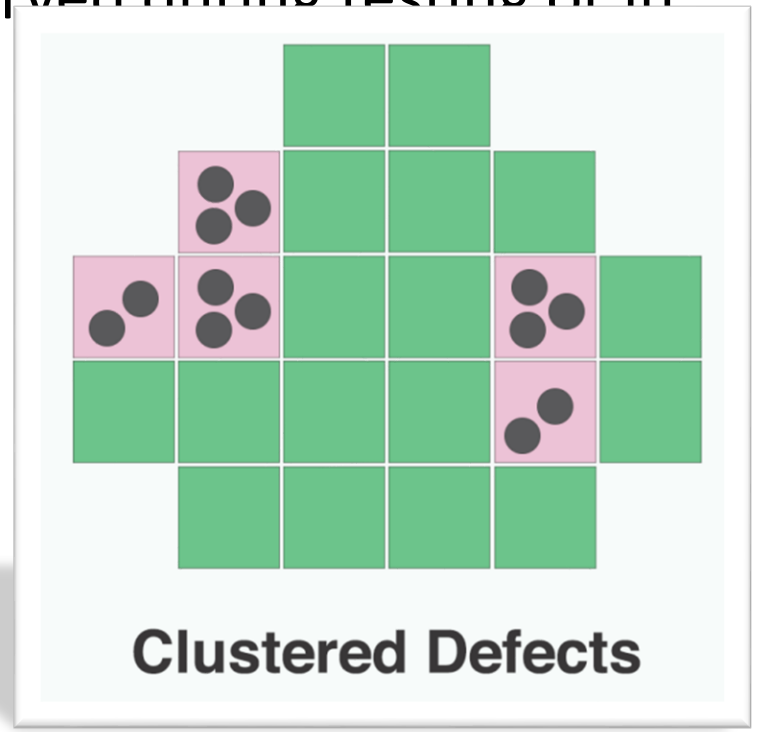
- Defects that are removed early will not cause subsequent defects in derived work products.
- The cost of quality will be reduced since fewer failures will occur later.
- Both static and dynamic testing should be started as early as possible



4. Defects cluster together

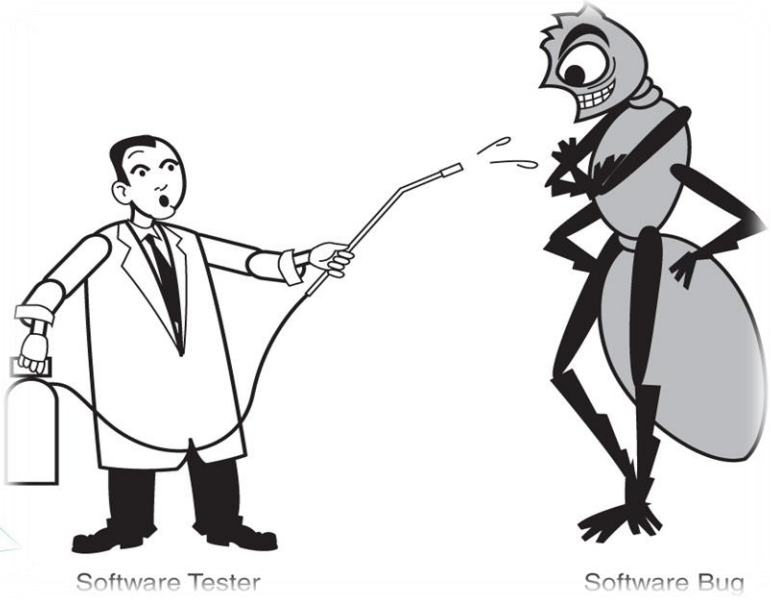
- Small number of system components usually contain most of the defects discovered or are responsible for most of the operational failures.
- Predicted defect clusters, and actual defect clusters observed during testing or in operation, are an important input for risk-based testing.





5. Tests wear out

- If same tests are repeated many times, they become increasingly ineffective in detecting new defects.
- Existing tests and test data may need to be modified, and new tests may need to be written.
- In some cases, repeating the same tests can have a beneficial outcome, e.g. in automated regression testing.

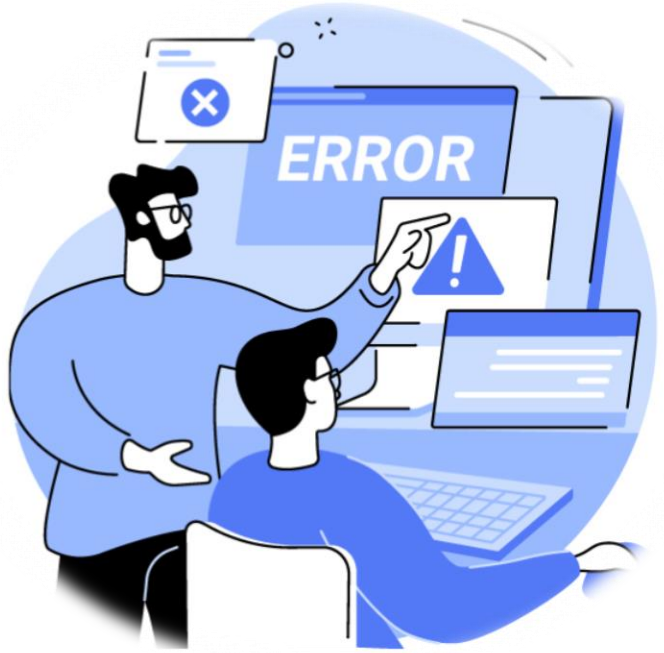


6. Testing is context dependent

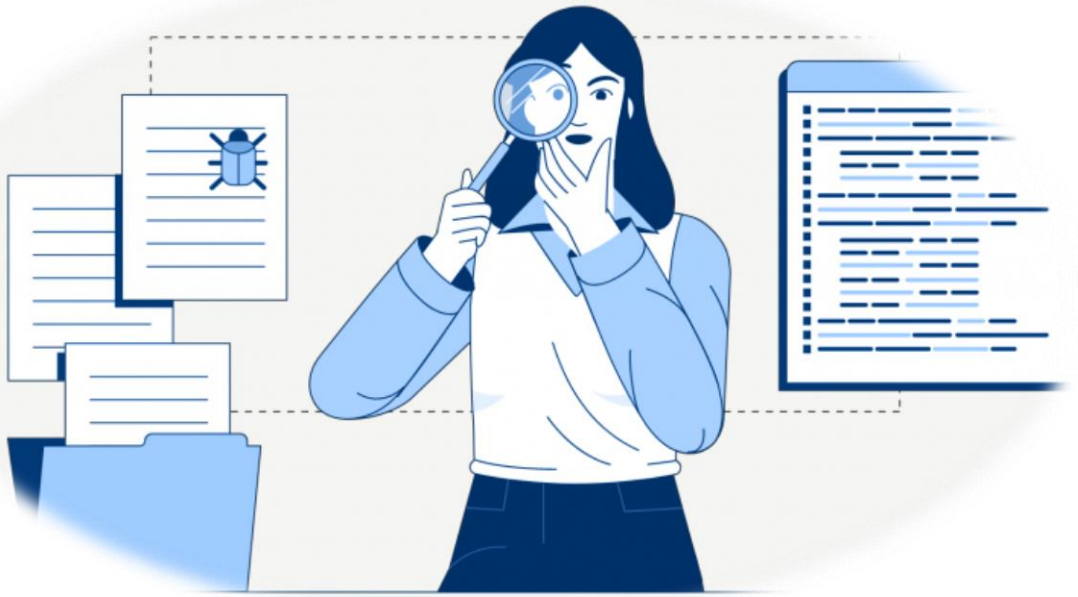
- There is no single universally applicable approach to testing.
- Testing is done differently in different contexts.

7. Absence-of-defects fallacy

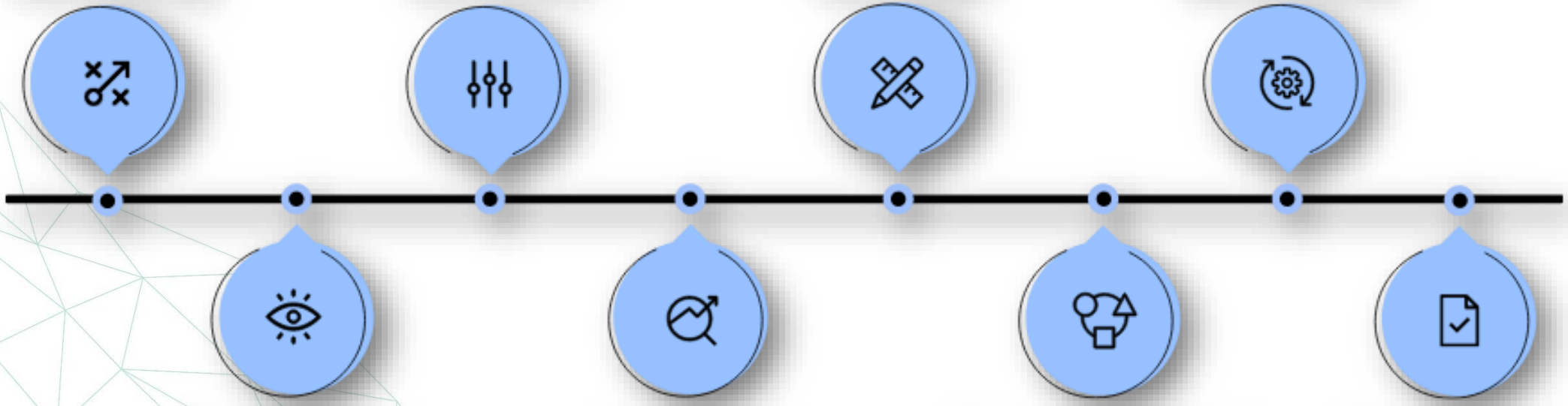
- It is a fallacy (misconception) to expect that software verification will ensure the success of a system.
- Testing all the specified requirements and fixing all the defects found could still produce a system that does not fulfill the users' needs and expectations.
- Validation should also be carried out.



❑ Test Activities and Tasks



- Testing is context dependent but, at a high level, there are common sets of test activities.
- Although many of these activities may appear to follow a logical sequence, they are often implemented iteratively or in parallel.
- These testing activities usually need to be tailored to the system.



It is a detailed strategy that outlines the test strategy, schedule, estimations, and deadlines.

Test Planning

Test Monitoring

Test monitoring involves the on-going comparison of actual progress against planned progress using any test monitoring metrics defined in the test plan.

Taking actions necessary to meet the objectives of the test plan.

Test Control

Test Analysis

It determines “What needs to be tested”.

It describes “How testing should be done”.

Test Design

Test Implementation

It decides When to start test execution.

It is a process of executing the test and comparing the actual and expected results.

Test Execution

Test Completion

It showcases the completed data from test execution.

1. Testing Planning

Defining objectives of testing

Defining the approach for meeting that objectives

Defining the scope of testing

Defining entry and exit criteria

Formulating a test schedule (Time, Cost, Resources)

2. Test Monitoring and Control

Test monitoring: involves the ongoing checking of all test activities and the comparison of actual progress against the plan

Test control: involves taking the actions necessary to meet the objectives of testing

3. Test Analysis

Identify testable features

Define and prioritize associated test conditions

The test basis and the test objects are also evaluated to identify defects

Test analysis answers the question “what to test?”

4. Test Design

Elaborating the test conditions into test cases and other testware

Defining the test data requirements

Designing the test environment

Identifying any other required infrastructure and tools

Test design answers the question “how to test?”

5. Test Implementation

Creating or acquiring the testware necessary for test execution (e.g. test data)

Test cases can be organized into test procedures and are often assembled into test suites

Manual and automated test scripts are created

Test procedures are prioritized and arranged within a test execution schedule for efficient test execution

The test environment is built and verified to be set up correctly

6. Test Execution

Running the tests which may be manual or automated

Test execution can take many forms (continuous testing or pair testing sessions)

Actual test results are compared with the expected results

Test results are logged

Anomalies are analyzed to identify their likely causes

Report the anomalies based on the failures observed

7. Test Completion

Usually occur at project milestones

Testware are archived or handed over to the appropriate teams

Test environment is shut down

Test activities are analyzed to identify lessons learned and improvements for future iterations

Test completion report is created

❑ Test Process in Context

Testing is not performed in isolation.
Test activities are an integral part of the development process.

The way the testing is carried out will depend on a number of contextual factors including:

- Stakeholders (needs, expectations, requirements, willingness to cooperate, etc.)
- Team members (skills, knowledge, level of experience, availability, training needs, etc.)
- Business domain (criticality of the test object, identified risks, market needs, legal regulations)
- Technical factors (type of software, product architecture, technology used, etc.)
- Project constraints (scope, time, budget, resources, etc.)
- Organizational factors (organizational structure, existing policies, practices used, etc.)
- Software development lifecycle (engineering practices, development methods, etc.)
- Tools (availability, usability, compliance, etc.)

Testware


Partners who make jobs better and faster.
35%
TESTWARE
TESTER
65%

Testware: output work products from the test activities

Test planning work products

- test plan
- test schedule
- list of risks
- entry and exit criteria
- risk register

Test monitoring and control work products

- test progress reports
- documentation of control directives and risk

Test analysis work products

- test conditions
- defect reports: defects in the test basis

Test design work products

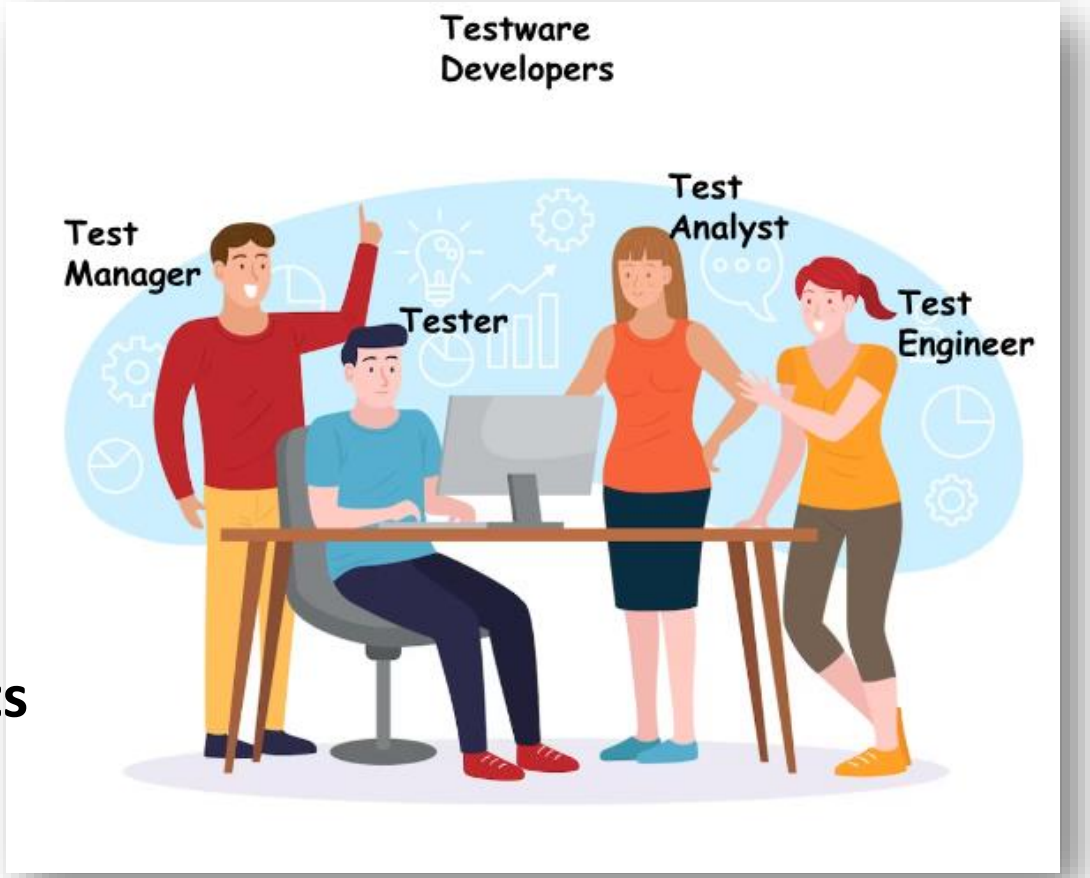
- test case
- test charters
- coverage items
- test data requirements
- test environment requirements

Test implementation work products

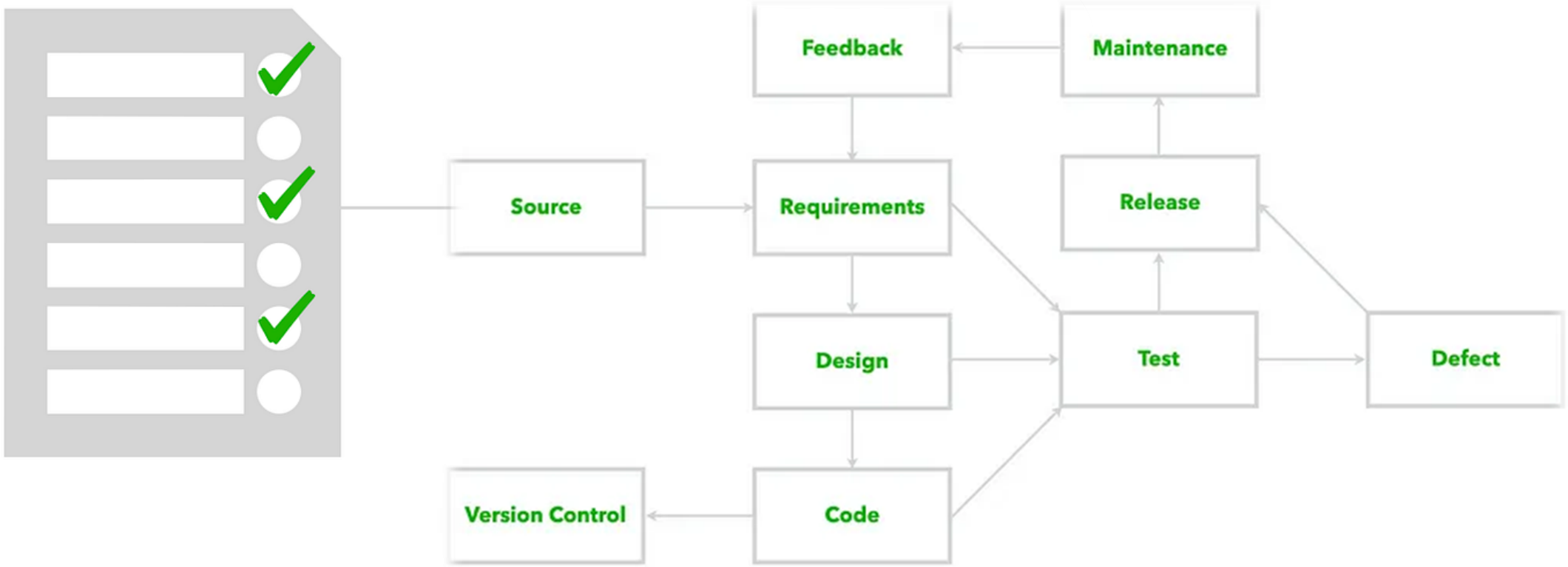
- test procedures
- automated test scripts
- test suites, data
- test execution schedule
- test environment elements

Test execution & completion work products

- Test logs
- defect reports
- test completion report



❑ Traceability



Traceability

❑ Roles in testing

- **Two principal roles:**

1. Test management role:

- Overall responsibility for the test process, test team and leadership of the test activities.
- Focus on test planning, test monitoring and control and test completion.

2. Testing role:

- Responsible for the engineering (technical) aspect of testing.
- Focus on test analysis, test design, test implementation and test execution.

❑ Generic Skills Required for Testing

- Testing knowledge.
- Thoroughness, carefulness, curiosity, attention to details, being methodical.
- Good communication skills, active listening, being a team player.
- Analytical thinking, critical thinking, creativity.
- Technical knowledge.
- Domain knowledge.

❑ Whole Team Approach

- In the whole-team approach any team member with the necessary knowledge and skills can perform any task.
- Everyone is responsible for quality.
- Team members share the same workspace, as co-location facilitates communication.
- Improves team dynamics, enhances communication and collaboration within the team
- Allowing the various skill sets within the team.
- Testers work closely with other team members to ensure that the desired quality levels are achieved.

❑ Independence of Testing

A certain degree of independence makes the tester more effective at finding defects due to differences between the author's and the tester's cognitive biases.

Independence is not, a replacement for familiarity.

Work products can be tested:

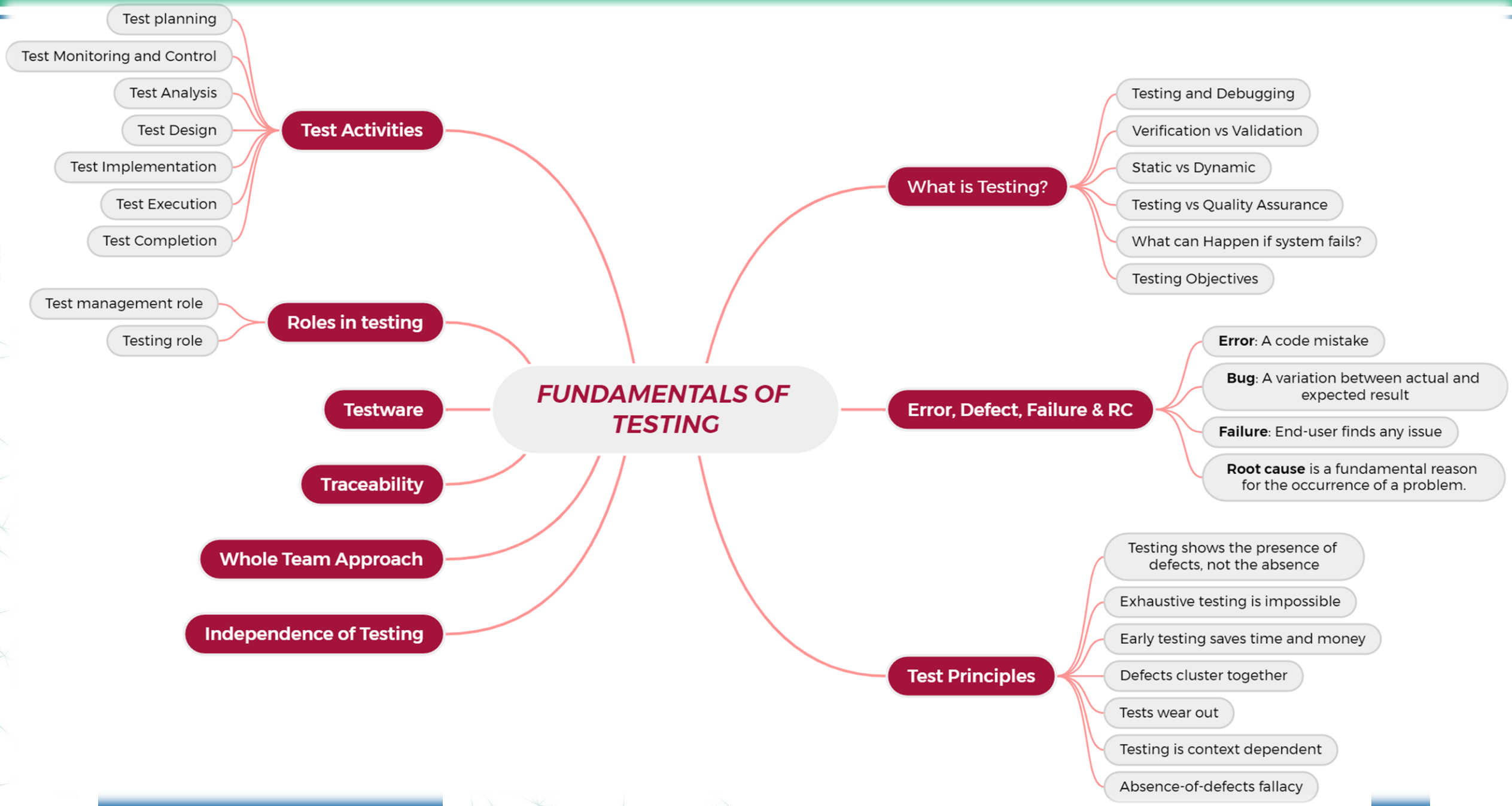
1. by their author (no independence).
2. by the author's peers from the same team (some independence).
3. by testers from outside the author's team but within the organization.
4. by testers from outside the organization.

For most projects, it is usually best to carry out testing with multiple levels of independence

❑ Benefits and drawbacks of Independent Testing

- **Benefits:**
  1. Independent testers are likely to recognize different kinds of failures and defects
  2. Independent tester can verify, challenge, or disprove assumptions made by stakeholders during specification and implementation.

- **Drawbacks:**
  1. Independent testers may be isolated from the development team.
  2. Lack of collaboration, communication problems and an adversarial relationship.
  3. Developers may lose a sense of responsibility for quality.
  4. Testers may be seen as a bottleneck or be blamed for delays in release.


FUNDAMENTALS OF TESTING
Test Activities
Test planning
Test Monitoring and Control
Test Analysis
Test Design
Test Implementation
Test Execution
Test Completion
Roles in testing
Test management role
Testing role
Testware
Traceability
Whole Team Approach
Independence of Testing
What is Testing?
Testing and Debugging
Verification vs Validation
Static vs Dynamic
Testing vs Quality Assurance
What can Happen if system fails?
Testing Objectives
Error, Defect, Failure & RC
Error: A code mistake
Bug: A variation between actual and expected result
Failure: End-user finds any issue
Root cause is a fundamental reason for the occurrence of a problem.
Test Principles
Testing shows the presence of defects, not the absence
Exhaustive testing is impossible
Early testing saves time and money
Defects cluster together
Tests wear out
Testing is context dependent
Absence-of-defects fallacy

رواد مصر الرقمية

وزارة الاتصـــــالات
وتكنولوجيا المعلومات

THANK YOU!