10/09/22

Software, Quality
Quality Assurance
Software Quality Assurance
Software QA → before.
Quality control → after the product is ~~delivered~~ made.
Verification
Validation
Software Testing - Error → conceptual or mistake made by developer or progra.
fault → specific manifestation of error
failure → inability of system to perform tasks in specified limits.
may occur when a fault is executed
bug → system faults

Attributes of a good test → ① high prob. of finding an error.
② not redundant
③ best of breed.
④ neither simple nor complex.

Principles of software Testing -

① Testing should be based on user requirements.
② " time & resources are limited.
③ impossible to test everything.
④ use effective resources to test.
⑤ test planning should be done early.
⑥ " for invalid & unexpec. input cond. as well as valid cond.
⑦ the prob of existence of more errors in a module or grp of module. α no. of errors already found.

(8) testing should begin at module independent/
(9) " " be done by external party.
(10) assign best person to the task.
(11) testing should not be planned under the implicit assu. that no error will be found.
(12) Keep software static during test.
(13) document test cases and test results.
(14) provide expected result if possible.

Characteristics of Software Testing :-

(1) Testability - how easily a computer program can be tested.
(2) Operability - the better it works the more efficiently it can be tested.
(3) Observability - what you see is what you test.
(4) Controllability - the better we control the soft. the more the testing can be automated & optimized.
(5) decomposibility - modules can be tested indepen-dently.
(6) Simplicity - less there is to test more we can test.
(7) Stability - fewer the changes fewer the disturbance to testing
(8) Understandability - more info. smarter test.

- more principles →
- All tests should be tracable to user requirements.
- exhaustive testing is not possible.
- should start from low to higher level.

Testing characteristics (more) :-

- for effective testing, effective formal technical reviews.
- diff. testing techniques are appr. for diff. point in time.

17/09/2022

- **Unit Testing** - testing individual prog. unit.

  modules - smallest prog. entity.

- **Integration Testing** - testing multiple modules working together.
  - Top-down → top functionality modules will be tested first
  - Bottom-up → error will be discovered early
  - Big-Bang.
    - difficult to find error. (dis)
    - it is quick, development time ↓ (adv)
    - no prototype to show (dis).

  adv - prototype
  dis - error in the basic module is detected at a later stage.

  Top-down.

- **Functional Testing** - each function is tested separately.

  it should do whatever it is supposed to do and not do what it is not supposed to do.

  It is difficult because -

  (1) may consists of lower-level functions each of which may be tested first.

  (2) func. should not coincide with function/module boundaries. This blur the dist. b/w integration & module/unit testing.

- Regression Testing :- is done to ensure that program changes have not degraded the system.
  verifies that problem do not surface again.

Problem arises when -

- System Testing :- ~~check the system~~ concerned with the execution of test cases to check the system inorder to check whether the req. of users are fulfilled or not.

  OR

  series of test, in order to fully test the system.

- Acceptance Testing :- is a process of exe. the test cases agreed with a customer as being an adequate represent-ation of user's requirements.

- Recovery Testing :- is designed to examine how easily and completly the system can recover from a disaster.
  - power shutdown
  - system failure
  - interface failure

  It is desirable that the system is capable of recovering easily with minimal human intervention

bug → system fault

- Security Testing :- It involves testing the system in order to make sure that unauthorized personal or other systems can't gain access to the system and information or resources within it.
  - Soft. Quality, security, reliability are closely coupled
  - It protects from improper penetration.
  - It verifies that protection mechanism built into a system will protect it from improper penetration.

  purpose - identifying & removing soft flaws that may potentially lead to security violation and validating the effectiveness of security measure.

- Performance Testing :- Soft. should not take ∞ time or ∞ resources to execute. It should be free from performance bugs.

  It involves monitoring and recording the performance stats during regular and low and high stress loads.

  It is designed to test the run-time perform. of a software within the context of an integrated system. It occurs throughout all steps in the testing process.

- Reliability Testing :- it refers to the prob. of failure free operation of a system.

- Robustness Testing :-

- Stress Testing :- it encompasses creating unusual loads on the system in attempts to break it.
  - used to test the whole system rather than the software alone.
- Load Testing :- load is weight from min to max level.
- Thread Testing :- it is suitable for testing real-time systems. In this testing
- Seeding Testing :- It is used
- Defect Testing :-
- Interface Testing :- to discover defects in the interface.
  medium
- α - Testing :- developer's side
- β - Testing :- user's side.

- Installation Testing :-

24/09/2022 Date

(4) Testing and Debugging

planned systematically — occurs as a consequence of successful testing.

Debugging :-

→ is a process that results in removal of error.

→ It begins with the execution of test cases.

→ Two steps →
① Identified caster (correct it)
② Suspected cause. (test again)

Testing	Debugging
① Testing is a process using which we find errors & bugs.	① It is a process using which we correct the bugs that we found during testing process.
② It is the display of errors.	② It is a deductive process.
③ It is a process to identify the failure of implemented code.	③ It is a process to give the absolution to code failure.
④ It is done by testers.	④ It is done either by programmer/ developer.
⑤ There is no need of design knowledge.	⑤ It is done only by insider.
⑥ It can be manual or automated.	⑥ It is always manual.
⑦ It is based on different levels of testing i.e unit, integration etc.	⑦ It is based on diff. types of bugs.
⑧ It is a stage of SDLC.	⑧ It is not an aspect of SDLC. It occurs as a conseq. of testing.
⑨ Testing is initiated after the code is written.	⑨ It commences with the execution of test cases.

→ Why is debugging a difficult task? (2)

(1) the symptom and cause may be geographically remote.
(2) the symptom may disappear when another error is corrected
(3) the symptom may actually be caused by a non-error (eg. round off)
(4) the symptom " be caused by human error that is not easily traced.
(5) the symptom may be a result of timing prob. rather than processing problems.
(6) It may be difficult to accurately reproduce input conditions.
(7) the symptom may be intermittent.
(8) " " " " due to causes that are distributed across a number of tasks running on different

processors.

Debugging Techniques :-

(1) Brute-force debugging :- It is applied when all else fails.
(a) It is the most common technique but least efficient.
(b) In this technique the program is loaded with print statements to print the intermediate value with the hope that some of the printed values will help to identify the statement with error.

(2) Backtracking Debugging :- This technique involves backtracking the incorrect results through the logic of the program. The source-code is traced backwards until the error is discovered. It becomes unfit when the no. of source-lines to be traced back increases and the no. of potential backward paths becomes unmanageable.

(3) Debugging by Induction :- It is a process that locates the data organises it and derives a hypothesis. This hypothesis must then be proved.

(4) Debugging by Deduction :- In this process determine the possible causes and use data to eliminate causes. Then refine the remaining cause into a hypothesis & prove it.

(5) Cause elimination Debugging :- In this technique, a list of causes that could have contributed to the error symptom is prepared and tests are carried out to eliminate each cause.

(6) Debugging by Testing :- It involves using two types of test cases — (1) cases that expose a previously undetected error.
(2) and cases that provide useful information in debugging to locate an error.

(7) Debugging by program slicing :- In this technique the overall search space is first divided in the program slices so that the search is confined to the program slice only. It is similar to backtracking.

④ Test case:- It is a document which has a set of test data, pre-conditions, expected results and post-conditions developed for a particular test scenario in order to verify compliance against a specific requirement.

→ It is a set of actions, executed to verify a particular feature or functionality of your software application. It is required to check if a particular application or software is working or not.

⇒ It defines how to test a system, software or applications.

→ It is a set of actions, performed on a system to determine if it satisfies the software requirements and functions correctly.

Purpose :- ① Guaranteed good test coverage. ② reduce maintenance & software support costs ③ Improve quality of soft. and user experience. ④ more reliable products are produced ⑤ higher quality products. Spiral leads to more satisfied customer. ⑥ More satisfied customer.

increase company profit. (7) clients are more satisfied.

Date ________

Test case should be —

① simple & clear ② traceable ③ minimal description
④ zero assumptions ⑤ maintains uniqueness ⑥ concise
⑦ transparent ⑧ easy to implement

Input of test case should be —

① different ② cover various conditions ③ ~~Factual~~ result
④ repetative result.

parameters of test case :-

1. module name
2. test case id
3. Test scenario
4. Test case description.
5. Test steps
6. pre requirement
7. Test data
8. Actual result.
9. comments