BACHELOR OF SCIENCE (COMPUTER SCIENCE)

SOFTWARE TESTING AND QUALITY ASSURANCE

CBCGS(NOV - 2022) Q.P.Code: 14365

Q1. Attempt All

a) Choose the correct alternative from the options given. (10)

I. A deviation from the specified or expected behaviour that is visible to end-users is called **error**.

a. error b. fault
c. failure d. defect

II. How many levels are present in CMM?

c. **5**
a. 3 b. 4
c. 5 d. 6

III. SMI stands for **Software Maturity Index**.

a. Software Mature Indicator b. Software Maturity Index
c. Software Mature Index d. Software Maturity Indicator

IV. Which of the following is not an appraisal cost in SQA?

c. **Quality planning**
a. Inter-process inspection b. Maintenance
c. Quality planning d. Testing

V. Exit criteria in test plan mentions **When to stop testing activities**.

a. When to stop accepting new requirements
b. When to stop the code from moving to production
c. When to stop testing activities
d. When to stop from accepting new pieces of code

VI. **Defect Analysis** takes into account inputs about singular defects as well as defect priorities, product issues, defect resolution history, developers involved and the like.

a. Defect Prevention b. Defect Discovery
c. Defect Resolution d. Defect A nalysis

VII. Cost and schedule are a part of **Project Metrics**.

a. Product metrics b. Process metrics
c. Project metrics d. People metrics

VIII. What is not included in prevention costs?

d. **Equipment calibration and maintenance**

a. Quality planning
b. Formal technical reviews
c. Test equipment
d. Equipment calibration and maintenance

IX. In which of the following type of testing, testing is done without planning and documentation?

c. **Ad hoc testing**

a. Unit testing
b. Retesting
c. Ad hoc testing
d. Regression testing

X. Which document that prescribes the requirements with which the product or service has to comfirm?

b. **Specification**

a. Reliablity
b. Specification
c. Inspection
d. Audit

---

b) Fill in the blanks by selecting the correct option from the pool of options: **(5)**

(reusability, stub, customer, appraisal, fish bone, skeleton, portability, pareto, prevention, employee)

I. **Pareto** chart uses the rule of 20:80?

II. Quality improvement, Quality education, and Quality performance reporting fall under the category of **Prevention** costs.

III. **Portability** is effort required to transfer the program from one hardware and/or software system environment to another.

IV. In Unit Testing, **Stub** serves to replace modules that are subordinate to the component to be tested.

V. The objective of ISO-9000 family of Quality Management is **Customer** satisfaction.

Q2. a) Define Quality and explain software quality attributes. (5)

Quality

Quality is defined as the degree to which a component, system or process meets specified requirements and/or user/customer needs and expectations.

A software quality is defined based on the study of external and internal features of the software.

Software Quality Attributes

There are 6 attributes of Software quality that are:

1. Functionality
2. Reliability
3. Performance
4. Flexibility
5. Usability
6. Securiy

1. **Functionality :-** Functionality is the conformity of the software with actual requirements and specifications.
   An application should provide all the required functionality as specified in the requirements.
2. **Reliability** :- The ability of an application or system to consistently perform its intended or required function on demand and without degradation or failure.
3. **Performance** :- Performance is mostly about response time of the software. This response time should be in acceptable intervals (e.g. max. a few seconds), and should not increase if transaction count increases.
4. **Flexibility** :- Flexibility is the ability of software to add or modify or remove functionality without damaging current system.
5. **Usability** :- Usability is the ease of use and learning ability of an application or system.
6. **Security** :- The degree to which the application is protected against malicious attacks and other potential risks.
   Security is very important especially when the application is dealing with sensitive user data like bank details.

Q2. b) Differentiate between Verification and Validation. **(5)**

Sr.No.	Verification	Validation
1.	Verification is to check whether the software conforms to the specifications.	Validation is to check whether software meets the customer expectations and requirements.
2.	Verifying process includes checking documents, design, code and program.	It is a dynamic mechanism of testing and validating the actual product.
3.	It does not involve executing the code.	It always involves executing the code.
4.	Verification uses methods like reviews, walkthroughs, inspections and desk- checking etc.	It uses methods like Black Box Testing, White Box Testing techniques.
5.	Checks "Are we building the product right"?	Checks "Are we building the right product"?
6.	QA team does verification and make sure that the software is as per the requirement in the SRS document.	With the involvement of testing team validation is executed on software code.

Q2. c) What is software review? Explain the types of reviews? **(5)**

Software Review

Software review is a process that involves testing the software product and ensuring that it meets the requirements stated by the client.
It is an important part of "Software Development Life Cycle (SDLC)" that assists software engineers in validating the quality, functionality, and other vital features and components of the software.

Types of Reviews

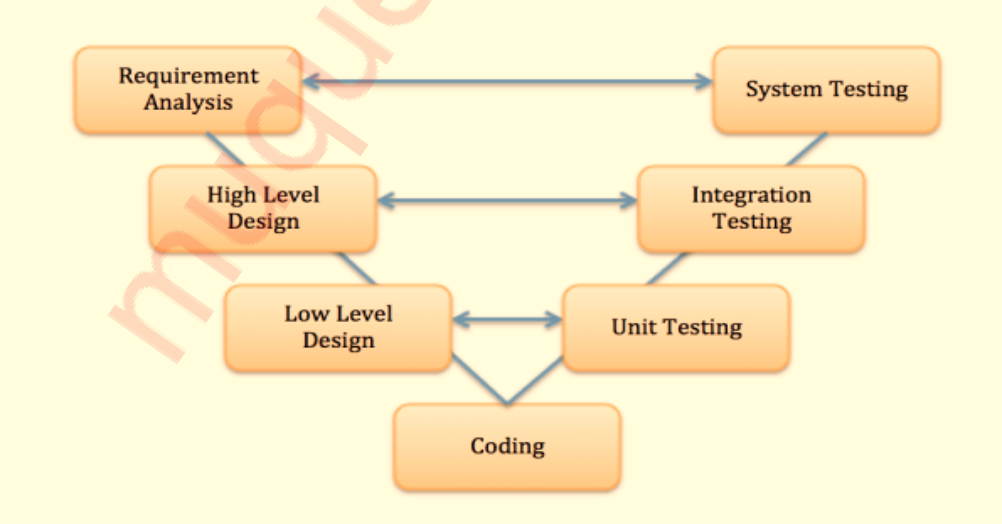
There are 3 types of reviews:

1. Walkthrough
2. Technical Review
3. Inspection

1. **Walkthrough**
    - This is the process where the authors of the document as well as other associates are gathered at one place to discuss the document to achieve a common understandings and gather feedback.
    - In walkthrough the author does most of the study of the document and the participants are not expected to know the document in advance.
    - The participants are selected from different departments and background so that they can bring in different viewpoints.
    - Questions are made, comments are given, answers are given to all the queries people have regarding the software.
    - With all the members satisfaction, conclusions are made.
2. **Technical Review**
    - This review is done to achieve technical consensus about the document.
    - In technical review, defects are found by experts.
    - It is performed by peer review without management.
    - Technical review can be informal or very formal.
3. **Inspection**
    - It is the most formal type of review.
    - During inspection, the documents are prepared and checked thoroughly by the reviewers before the meeting.
    - A separate preparation is carried out during which the product is examined and the defects are found.
    - The defects found are documented in a logging list or issue log.

Q2. d) Write a short note on V-V Model of software testing. (5)

V- Model (V & V Model)



- V-model ia also known as Validation and Verification model.
- V-model was developed to address some of the problems experienced in the traditional software development life cycle.
- V-model provides the guidelines that testing should begin as early as possible in the life cycle; this is also one of the important principles of testing.
- There are many activities related to testing that can be performed before the completion of coding phase.
- These activities can be carried out in parallel with the other development activities.
- Like SDLC, V-model is also sequential model. Each phase must be completed before the next phase begins.
- The various phase of V-model are:
    a) Requirements
    b) High-level Design
    c) Low-level Design
    d) Coding

Q2. e) Explain the term cyclomatic complexity with example. **(5)**

Cyclomatic Complexity

- Cyclomatic Complexity in Software Testing is a testing metric used for measuring the complexity of a software program.
- It is a quantitative measure of independent paths in the source code of a software program.
- This metric was developed by Thomas J. McCabe in 1976 and it is based on the control flow representation of the program.
- Control flow depicts a program as a graph which consists of Nodes and Edges.
- In the graph, nodes represent processing tasks while edges represent control flow between the nodes.
- Cyclomatic complexity can be calculated by using control flow graphs or with respect to functions, modules, methods or classes within a software program.
- Mathematically, it is set of independent paths through the graph diagram.
- The code complexity of the program can be defined using the formula –
- V(G) = E – N + 2
  Where,
  E – Number of Edges
  N – Number of Nodes

  V(G) = P + 1
  Where, P = Number of predicate nodes (nodes that contain condition)

Example

If (condition 1)
Statement 1

Else
Statement 2

If (condition 2)
Statement 3

Else
Statement 4

- As, V(G) = E – N + 2
  Cyclomatic Complexity for this program will be 8-7+2 = 3.

As complexity has calculated as 3, three test cases are necessary to the complete path coverage for the above example.

Q2. f) What is Coverage Criteria? List and explain any two coverage criteria in short. (5)

Coverage Criteria

A Coverage Criteria is a rule or collection of rules that impose test requirements on a test set. The Coverage Criteria describes test requirements completely and unambiguously.
Two Coverage Criteria in software testing are:

1) Statement Coverage
2) Branch Coverage

1 Statement Coverage

- In a programming language, a statement is nothing but the line of code or instruction for the computer to understand and act accordingly.
- A statement becomes an executable statement when it gets compiled and converted into the object code and performs the action when the program is in a running mode.
- Hence "Statement Coverage", as the name itself suggests, it is the method of validating whether each and every line of code is executed at least once.
- For Example – Consider the sample code

```
int A,B;
if A>B then C=0
End if
```

- To ensure 100% statement coverage only 1 test case is required where the value of A is greater than B, however if the value is A is less than B then statement C=0 will not be executed and hence 100% statement coverage will not be achieved.

2 **Branch Coverage**

- Branch coverage is a requirement that, for each branch in the program (e.g., if statements, loops), each branch have been executed at least once during testing.
- It is sometime also described as saying that each branch condition must have been true at least once and false at least once during testing.
- It aims to ensure that each one of the possible branch from each decision point is executed at least once and thereby ensuring that all reachable code is executed. It helps in validating that every branched code based on decision is executed at least once.
- For Example – Consider the following code

```
Read C
Read D
IF C>D Then
Print "D is Greater"
 ELSE
 Print "C is Greater"
```

Q3. a) Define regression testing? What is the role of capture and playback? (5)

Regression Testing

Regression Testing refers to a type of software testing that is used to verify and modification update in a software without affecting the overall working functionality of the software.

Role of Capture and Playback

- A Capture and Playback tool is a type of test execution tool that records entries during a manual test with the goal of creating automated test scripts that can then be used repeatedly.
- Capture and Playback soetimes referred to as Capture and Replay.
- It is often used to support automated regression testing.
- Capture mode records user interactions with user interface elements.
- Capture and Playback is a script that documents both the test process and the test parameters.
- Capture and Playback were developed to test the applications against graphical user interfaces.

- With a Capture and Playback tool, an application can be tested in which an interactive session can be repeated any number of times without human intervention.
- This saves time and effort while providing valuable insights for further application development.

Q3 b) Write a short note on Smoke Testing. **(5)**

Smoke Testing

- Smoke testing is a type of software testing that verifies the basis functionality of a software or a component before performing more detailed tests.
- It helps to identify any major errors or defects that could prevent the system from working as expected.
- Smoke testing has several benefits for software quality and efficiency.
- It helps to detect any critical issues or bugs that could affect the user experience or the system performance early in the development cycle.
- This reduces the risk of wasting time and resources on testing and debugging a faulty system.
- It helps to ensure that the system meets the basic requirements and specifications before moving on to more complex tests.
- It helps to save time and effort by focusing on the core functionality and features of the system rather than the minor details and enhancements.
- This allows the testers and developers to prioritize the most important aspects of the system and deliver them faster.
- Smoke testing can b performed at different levels and stages of the software development process.
- There are 3 types of smoke testing and that are:
    1. Manual Smoke Testing
    2. Automated Smoke Testing
    3. Hybrid Smoke Testing
- Smoke testing is also called as Build verification testing and confidence testing.
- The goal of smoke testing is to discover simple but severe failures using test cases that cover the most important functionalities of a software.
- Smoke tests are performed by QA teams using a minimal set of tests on each build that focuses on software functionality.

Q3. c) What is Validation testing? Write 7 validation test criteria. (5)

Validation Testing

Validation Testing is a type of software testing which ensure that each function or performance characteristic conforms to its specification.
Configuration review or audit is used to ensure that all elements of the software configuration have been properly developed, catalogued, and documented to allow its support during its maintenance phase.
The main goal of validation testing is to verify whether a software product meets its acceptance criteria.
It is the static practice of studying and verifying the specific requirements of a particular stage in development.
It does not require executing code.

Validation Test Criteria

1 Software Validation is achieved through tests for conformity with requirements.
2 A test plan ensure all functional requirements are satisfied.
3 All behavioral charactristics are achieved.
4 All content is accurate and properly presented.
5 Performace requirements are also attained.
6 Documentation is correct.
7 Usability, transportability, compatibility, error recovery, maintainability has to be checked.
8 Deficiency list is created for any deviation from specification is uncovered.

Q3. d) What is software metric? Explain its importance. (5)

Software metric

A Software metric is a measure of software characteristics which are masurable or countable.
The standard of measure for the estimation of quality, progress of the software testing effort is called Software metrics.
Software metrics are valuable for many reasons, including measuring software performance, planning work items, measuring productivity, and many other uses.
Software metrics are measurable characteristics of a software program.
The three types of metrics are:

1. Source code metrics
2. Development metrics
3. Testing metrics

Importance of Software metric

- Software metrics can be used to guage the status, effectiveness and efficiency of processes, customer satisfaction, product quality and as a tool for management concepts.
- It helps to identify the particular area for improvising.
- Reduction in overall time to produce the product.
- It helps to determine the complexity of the code and to test the code with resources.
- It helps in providing effective planning, controlling and managing of the entire product.
- The goal of tracking and analyzing software metrics is to determine the quality of the current product or process, improve that quality once the software development project is complete.
- Managers can use software metrics to identify, prioritize, track and communicate any issues to faster better team productivity.
- Software development team can use software metrics to communicate the status of software development projects, pinpoint and address issues, and monitor, improve on, and better manage their workflow.
- You can use software metrics to measure several aspects of the software development progress.
- The softwae quality specifications can be prepared using software metrics.

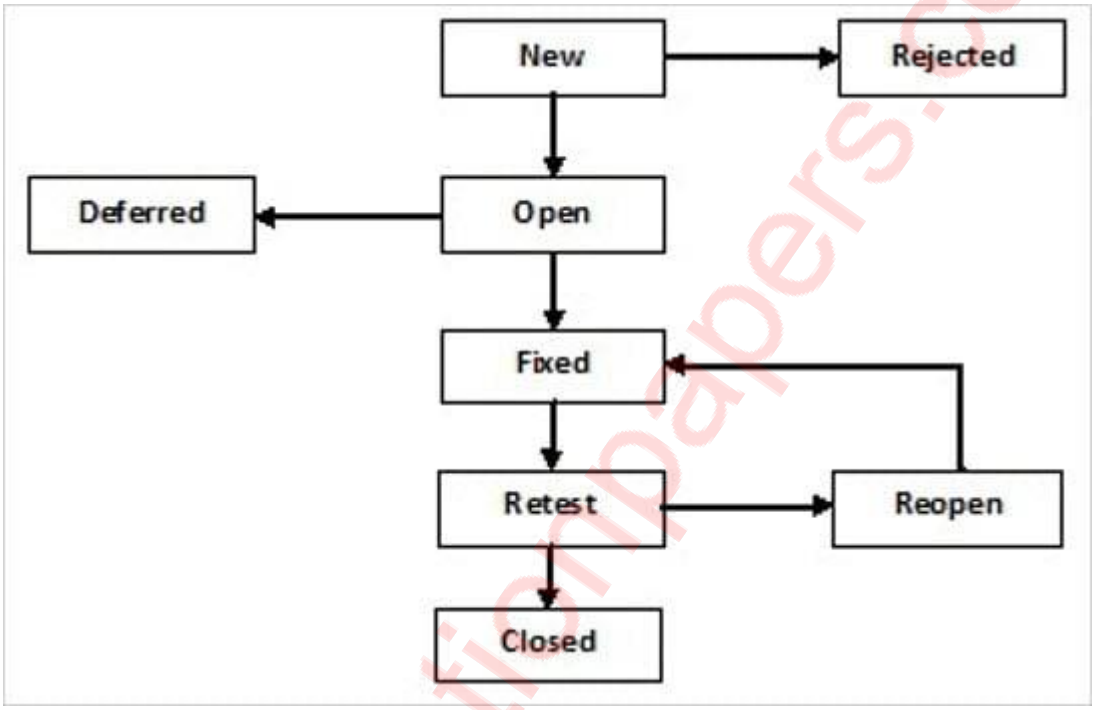
Q3. e) Explain defect life cycle. **(5)**

Defect

A Software Defect or Bug is a condition in a software product which does not meet a software requirement or end-user expectation.

A defect ia an error in coding or logic that causes a program to malfunction or to produce incorrect or unexpected results.

Defect Life Cycle



Defect Life Cycle has 6 states:

1) **New**
   This is the first state of defect life cycle.
   When any defect is identified logged, it is in the "New" state.
2) **Open**
   The newly logged defect is then assigned to the development team for solving.
   This is generally done by the project manager of the development team.
   In open state the developer goes through the defect to analyze it, if the developer feels that defect is not valid or similar defect have been logged then the defect is transferred to reject state.
   The defect can also be transferred to deffered state for future solving.
3) **Fixed**
   If the developer has worked on the defect and resolved it then the defect moves to fixed state.
   The testing team later needs to test and verify whether the defect is actually fixed.

4) **Retest**
In this test the development is waiting for the testing team to test the resolved defect to check whether the defect have been resolved accurately.
The team also checks whether fixing the defect or issue has led to any other new defect.

5) **Open**
If the tester while retesting feels the defect is not resolved or partially resolved then the defect state is changed to reopen and the development team will have to rework on the issue to resole it.

6) **Closed**
Once the testing team has checked and verified the defect and are sure that the defect does not exist any longer, the defect is then moved to closed state.

Q3. f) List the types of System Testing and explain any 2 types in short. (5)

System Testing

System Testing of software or hardware is testing conducted on a whole, integrated system to estimate the systems compliance with its specified set of requirements.

Types of System Testing

Types of System Testing are:

1) Unit Testing
2) Integration Testing
3) Validation Testing
4) Acceptance Testing

1) **Validation Testing**

- Validation Testing is a type of software testing which ensure that each function or performance characteristic conforms to its specification.
- Configuration review or audit is used to ensure that all elements of the software configuration have been properly developed, catalogued, and documented to allow its support during its maintenance phase.
- The main goal of validation testing is to verify whether a software product meets its acceptance criteria.
- It is the static practice of studying and verifying the specific requirements of a particular stage in development.
- It does not require executing code.
- Deviations must be negotiated with the customer to establish a means for resolving the errors.

2) **Acceptance Testing**

- Acceptance Testing is a type of software testing to make sure that the software works correctly for intended user in his or her normal work environment.
- It involves Alpha test and Beta test.
- Alpha test is a version of a complete software is tested by customer under the supervision of the developer at the developers site.
- Beta test is a version of a complete software is tested by customer at his or her own site without the developer being present.
- User is completely involved in Acceptance testing.
- Acceptance Testing determine the customer for satisfaction with software product.
- Acceptance testing only involves the Functional testing based on the requirement given by client or user.

Q4. a) List and explain the goals and objectives of SQA. **(5)**

Goals and Objectives of SQA

i) Customer Satisfaction:- Ensure that software development was made according to their needs, wants and exceeding their expectations.
ii) Well Structured:- SQA ensures that each application are build in an understandable manner. Their applications could easily be transferred from one developer to another.
iii) The software requirements are complete, correct and consistent.
iv) Ensure that software design adheres to established standards and guidelines.
v) Ensure that software products meet functional and non-functional requirements and are robust, reliable, and scalable.
vi) The software configurations, versions and changes are properly managed and tracked.
vii) Ensure that software documentation is complete, accurate and up-to-date.
viii) Identify and correct issues and ensure that software products are of high quality.
ix) Ensure that developers and stakeholders have the necessary skills and knowledge to develop and use high-quality software products.
x) It ensures management of data which is important for product quality.
xi) To improve development and testing processes to prevent defects from arising during the product development lifecycle.

Q4. b) Write a note on statistical Software Quality Assurance. (5)

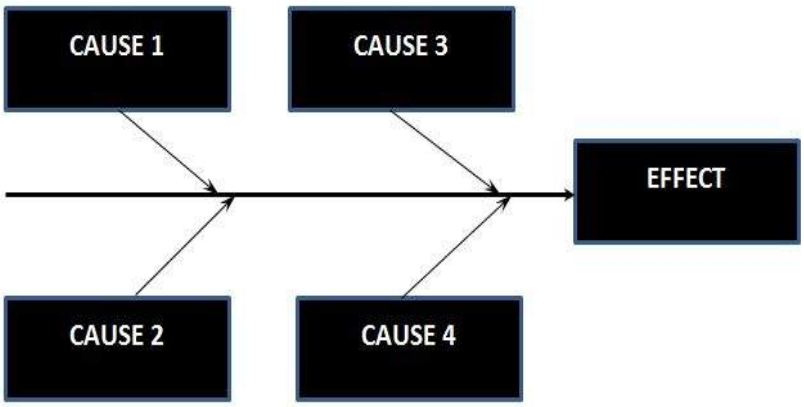
Statistical Software Quality Assurance

- Statistical Software Quality Assurance involves tracing each defect to its underlying cause and making moves to correct them.
- Statistical quality assurance reflects a growing trend throughout industry to become more quantitative about quality.
- For software, statistical quality assurance implies the following steps:
    1. Information about software defects is collected and categorized.
    2. Each defect is traced back to its cause.
    3. Using the Pareto principle (80% of the defects can be traced to 20% of the causes) isolate the 'vital few' defect causes
    4. Move to correct the problems that caused the defects in the 'vital few'.

Q4. c) Discuss how reliability changes over the lifetime of a software product and a hardware product. (5)

- Reliability behaviour for hardware product and software product are totally different.
- Reliability in software is software that has no failure and works in a special time period with a special environment.
- Hardware Reliability is the probability of the absence of any hardware-related system malfunction for a given mission.
- Software Reliability is the probability that the software will provide a failure-free operation in a fixed environment for a fixed interval of time.
- In hardware product, failures occurs due to physical deterioration in wear and tear.
- Hardware components may rust or wear out with time and usage, but software will not rust or wear out during its life cycle.
- Software will not change over time unless intentionally changed or upgraded.
- Software reliability is hard to achieve, because the complexity becomes too high. While any system with a high degree of complexity, including software will be hard to reach a certain level of reliability.
- Reliability of software is maintained until any fault find in hardware which affects the path of the data. Physically errors always find in software system.
- Reliability of a hardware product improves over time and remains fairly constant until end of life when components begin to wear out but reliability of a software product improves piecewise over time but faces periodic down-turns because of the bugs and increasing complexity associated with upgrades.

Q4. d) Write short note on Cause-effect Diagrams. **(5)**



- A cause and effect diagram also known as a "Ishikawa" or "fishbone" diagram.
- It is a graphic tool used to explore and display the possible causes of a certain effect.
- Use the classic fishbone diagram when causes group naturally under the categories of Materials, Methods, Equipment, Environment, and People.
- Use a process-type cause and effect diagram to show causes of problems at each step in the process.
- A cause and effect diagram has a variety of benefits:
    a) It helps teams understand that there are many causes that contribute to an effect.
    b) It graphically displays the relationship of the causes to the effect and to each other.
    c) It helps to identify areas for improvement.
- Cause Effect Graph is a black box testing technique.
- It is also known as Ishikawa diagram because of the wat it looks, invented by Kaoru Ishikawa or fish bone diagram.
- It is a testing technique that aids in choosing test cases that logically relate Causes(inputs) to Effects(outputs) to produce test cases.
- A "Cause" stands for a separate input condition that fetches about an internal change in the system. An "Effect" represents an output condition, a system transformation or a state resulting from a combination of causes.
- Cause-effect diagram can be used to determine the current problem so that right decisions can be taken very fast.
- It can be used to recognize the probable root causes, the cause for a exact effect, problem or outcome.

Steps to proceed on Cause-Effect Diagram:

1) Recognize and describe the input conditions(causes) and actions(effect).
2) Build up a cause-effect graph.

3) Convert cause-effect graph into a decision table.
4) Convert decision table rules to test cases. Each column of the decision table represents a test case.

Q4. e) What is quality cost? Explain the objectives of finding quality cost. (5)

Quality Cost

Cost of quality is one of the most established, effective measures of quantifying and calculating the business value of testing.
Quality cost is a method for calculating the costs companies incur ensuring that products meet quality standards, as well as the costs of producing goods that fail to meet quality standards.

Objectives of finding quality cost

- The goal of calculating the cost of quality is to create an understanding of how quality impacts the bottom line.
- To achieve and maintain quality output.
- The companies determine the cost of quality to derive a competitive advantage in the industry.
- Ensures that the business maintains a positive bottom line.
- Helps the organization to chalk out wrong and poor quality output.
- Helps in problem-solving, wherein it performs cost and benefits analysis on quality and process improvement initiatives.
- Evaluates the costs of failures and appraises them accordingly.
- Provide managers with information about how quality system works and also helps managers to evaluate the success in achieving quality goals.
- Determine the true(full) costs of each of the programs under analysis.
- Increase an organisations productivity and enhance profitability.
- Identify the sources of quality failures.

Q4. f) Explain the outline structure for a quality plan. (5)

A quality plan is a document, or several documents, that together specify quality standards, practices, resources, specifications, and the sequence of activities relevant to a particular product, service, project, or contract.

Quality plans should define:

- Objectives to be attained (for example, characteristics or specifications, uniformity, effectiveness, aesthetics, cycle time, cost, natural resources, utilization, yield, dependability, and so on).

- Steps in the processes that constitute the operating practice or procedures of the organization.
- Allocation of responsibilities, authority, and resources during the different phases of the process or project.
- Specific documented standards, practices, procedures, and instructions to be applied.
- Suitable testing, inspection, examination, and audit programs at appropriate stages.
- A documented procedure for changes and modifications to a quality plan as a process is improved.
- A method for measuring the achievement of the quality objectives.
- Other actions necessary to meet the objectives.

At the highest level, quality goals and plans should be integrated with overall strategic plans of the organization.
At lower levels, quality plan assumes the role of an actionable plan.
Such plans may take many different forms depending on the outcome they are to produce.
Quality plans may also be represented by more than one type of document to produce a given outcome.

Q5. a) Define the terms: error, fault and failure. **(3)**

Error

A mistake made in coding is known as “Error”.
An error is a humn action that produces an incorrect result.

Fault

“Fault” is a condition that causes the software to fail to performed its required function.

Failure

“Failure” is a inability of a system or component to perform required function according to its specification.

Q5. b) Distinguish between White box and Black box testing. **(3)**

Sr.No.	White Box Testing	Black Box Testing
1.	It is also known as glass box testing, clear box testing or structural testing.	It is also known as specification based testing or input/output driven testing.
2.	White Box Testing is a software testing method in which the internal structure/design/implementation of the item being tested is known to the tester.	Black Box Testing is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester.
3.	White box testing requires knowledge of programming languages.	Black box testing does not require knowledge of programming languages.
4.	It is done by software developers.	It is done by software testers.
5.	It is most time consuming.	It is least time consuming.
6.	In this type of testing, testers mainly focuses on the funtionality of the system.	In this type of testing, developers mainly focuses on the structure means program/code of the system.

Q5. c) State the objectives of testing. **(3)**

Objectives of Testing

- Finding defects in the software product that may have occurred during the development of the software due to human error.
- Gaining confidence and concluding the quality of the software developed.
- To prevent future defects from occuring.
- To ensuring that the developed product satisfies Business Requirement Specification (BRS) and System Requirement Specifications (SRS).
- To gain customers confidence by delivering them a high quality product that has minimum defects possible.
- To find bugs as early as possible and fix bugs and make sure that the software is bug-free.
- Enhancement of scalability and reliability.
- Identify the correctness, completeness and quality of developed software.

Q5. d) What is the difference between function-point metrics and feature-point metrics. **(3)**

Sr.No	Function-point Metrics	Feature-point Metrics
1.	Function Point is the most widespread functional type metrics suitable for quantifying a software application.	Feature Point is the superset of function point measure that can be applied to systems and engineering software applications.
2.	For a given software application, each of these elements is quantified and weighted.	Feature points are weighed by only single weight.
3.	The Function Point metric is calculated from software complexity assessment.	The Feature Point metric computed by counting the informaion domain values.
4.	It can be used for measuring the size of Management Information System(MIS) software.	It can be used in those areas where there is a level of complexity, is comparitavely very high.
5.	A function point metric overcomes the shortcoming of the LOC metric.	A function point extension is called Feature point.
6.	The function point measure accommodates applications in which algorithm complexity is low.	The feature point measure accommodates applications in which algorithm complexity is high.

Q5. e) Discusses the differences between Alpha and Beta testing. **(3)**

Sr.No	Alpha Testing	Beta Testing
1.	Alpha testing is a type of software testing performed to identify bugs before releasing the product to real users or to the public.	Beta Testing is performed by real users of the software application in a real environment.
2.	Alpha testing involves both Black-box testing and white-box testing.	Beta Testing commonly uses black-box testing.
3.	Alpha testing is performed by testers who are usually internal employees of the organization.	Beta testing is performed by clients who are not part of the organization.
4.	Alpha testing is performed at the developer's site.	Beta testing is performed at the end-user of the product.

5.	Reliability and security testing are not checked in alpha testing.	Reliability, security and robustness are checked during beta testing.
6.	Alpha testing requires a testing environment or a lab.	Beta testing doesn't require a testing environment or a lab.

Q5. f) What are the guidelines for review. **(3)**

Guidelines for review

1. Review the product not the producer.
2. Set an agenda and maintain it.
3. Limit debate and contradiction.
4. Pronounce problem area but don't attempt to solve every problem noted.
5. Take written notes.
6. Limit the number of participants and insist upon advance preparation.
7. Develop a checklist for each product that is likely to be reviewed.
8. Allocate resources and schedule time for FTRs.
9. Conduct meaningfull training for all reviewers.
10. Review your early reviews.

Q5. g) What is Run chart? How to create and interpret it? **(3)**

Run Chart

A Run Chart is a graph of data in chronological order that displayes changes and trends in the central tendency(average).
Run charts are often used to monitor and quantify process outputs before a control chart is developed.
Run charts an be used as input for establishing control charts.

There are seven steps to creating a run chart.

1. Decide on the measure to be analyzed.
2. Gather the data - have a minimum of 10 data points.
3. Draw a graph with a vertical line and a horizontal line.
4. On the vertical line, or the y-axis, draw the scale relative to the variable you are measuring.
5. On the horizontal line, or the x-axis, draw the time or sequence scale.
6. Calculate the mean/median (whichever the data set indicates to be appropriate) and draw a horizontal line at that value- going across the graph.
7. Plot the data in the sequence or the time order, in which the data was collected.

Q5. h) Discuss the concept of software safety. **(3)**

Software Safety

- Software Safety defined as a software quality assurance activity that focuses on identifying potential hazards that may cause a software system to fail.
- Early identification of software hazards allows developers to specify design features can eliminate or at least control the impact of potential hazards.
- Software reliability involves determining the likelihood that a failure will occur, while software safety examines the ways in which failures may result in conditions that can lead to a mishap.
- It optimizes system safety in the design, development, use, and maintainence of software systems and their integration with safety-critical hardware systems in an operational environment.
- Software is only hazardous when operating or controlling hardware.
- This provide another key to software safety focus on the safety critical hardware and hazardous system operations and modes.
