Q. P. Code:

(2½ hours)

[Total Marks: 75]

N. B.: (1) **All** questions are **compulsory**.
(2) Make **suitable assumptions** wherever necessary and **state the assumptions** made.
(3) Answers to the **same question** must be **written together**.
(4) Numbers to the **right** indicate **marks**.
(5) Draw **neat labeled diagrams** wherever **necessary**.
(6) Use of **Non-programmable** calculators is **allowed**.

1 Attempt *any two* of the following: **10**

a What is quality? Explain quality viewpoints and expectations in producing and buying software.

- Quality –The degree to which a component, system or process meets specified requirements and/or user/customer needs and expectations
- The right software system should built correctly, the customer wants the project to be within budget and timescale

Viewpoint	**Software**
Quality is measured by looking at the attributes of the product.	e.g.Reliability in terms of mean time between failures (MBTF),
Quality is fitness for use. Quality can have subjective aspects and not quantitative aspects.	Whether users can carry out their tasks;
Quality is based on good manufacturing processes, and meeting defined requirements. It is measured by testing inspection, and analysis of faults failures.	We will use a recognized software development process.We will release the software if there are fewer than five outstanding high-priority defects once the planned tests are complete.
Expectation of value for money. affordability, and a value-based trade-off between time, effort and cost aspects. We can afford to buy this software and we expect a return on investment.	We have time-boxed the testing to two weeks to stay in the project budget.
Transcendent feelings - this is about the feelings of an individual or group of individuals towards a product or a supplier.	We really enjoy working with this software team. So, there were a few problems - they sorted them out really quickly - we trust them.

b State the phases of fundamental test process . Describe test closure activity of the test process.

- The activities within the fundamental test process
  - Planning and control
  - Analysis and design
  - Implementation and execution
  - Evaluating exit criteria and reporting
  - Test closure activities
- Test closure has the following tasks:

- Check which planned deliverables we actually delivered and ensure all incident reports have been resolved through defect repair or deferral(remain open)
- Finalize and archive testware for later reuse
- Hand over testware to the maintenance organization
- Evaluate how the testing went and analyze lessons learned for future releases and projects

c Write note on Psychology of Testing.

Psychological factors influencing testing and success are

- Clear Objectives
- Proper roles
- Independent testing
- Clear and courteous communication
- Feedback on defects

Independent Testing-Different mindset of developer and tester

- Reviews of requirements and design documents by someone other than author
- The degree of independence avoids author bias and is ofter more effective at finding defects and failures.
- Level of independence from lowest to highest
  - Tests by the person who wrote the item under test
  - Tests by another person person within the same team ,such as another programmer
  - Tests by a person from a different organizational group ,such as an independent test team
  - Test designed by a person from a different organization or company,such as outsourced testing or certification by an external body

d What is the difference between software fault and failure?What ae the causes of software failure ?

Fault(bug,defect)-A flaw in a component or system that can cause the component or system to fail to perform its required function.Eg:incorrect statement or data definition.A defect may cause a failure of the component or system

Failure-Deviation of the component or system from its expected delivery ,service or result

Failures can also be caused because of the following reasons :

- Defects may give rise to failure.
- Because of the environmental conditions as well like a radiation burst, a strong magnetic field, electronic field or pollution could cause faults in hardware or firmware. Those faults might prevent or change the execution of software.
- a wrong input value being entered or an output being misinterpreted.

Failures may also be caused by someone deliberately trying to cause a failure in the system-malicious damage. Failures may also arise because of human error in interacting with the software, perhaps

2 Attempt *any two* of the following: **10**

a What is incremental development model?Briefly explain the characteristics of Rapid Application Development.

Incremental development model

- A development life cycle where a project is broken into a series of increments
- Each increment delivers a portion of functionality in the overall project requirements
- The requirements are prioritized and delivered in priority order in the appropriate increment

- Each subproject follows a "mini V-model" with its own design ,coding and testing phases

Rapid Application Development

- Components /functions are development in parallel
- The developments are time-boxed ,delivered and assembled into a working prototype
- Customer can see and provide feedback
- Early validation of technology risks and a rapid response to changing customer requirements
- Dynamic System Development Methodology(DSDM) –refined RAD process that allows control
- Maintain strict configuration management
    - Encourage customer feedback
    - Early visibility of the product
    - What functionality to include in next cycle
    - Decision to whether halt the project

b What is maintenance testing? Explain the different reasons for maintenance testing.

Maintenance testing is testing the changes to an operational system or the impact of a changed environment to an operational system.

- Triggered by modification,migration and retirement
    - Modification
        - Planned enhancement changes
        - Corrective and emergency changes
        - Changes of environment
        - Patches to newly exposed or discovered vulnerabilities of OS
        - Two types of modification
            - Planned Modification
                - perfective modifications
                - adaptive modifications
                - corrective planned modifications
            - Ad -hoc corrective Modification
                - Concerned with defects requiring an immediate solution
    - Migration
        - Operational testing of the new environment as well as changed s/w
    - Retirement
        - Testing of data migration or archiving

c What is integration testing ?Explain any two integration strategies with its advantages and disadvantages.

❖ Integration testing tests interfaces between components, Interactions to different parts of a system such as OS,file system and hardware or interfaces between systems.

Top-down integration: The test starts with the top level component of the system that calls other components but is not called itself (except for a call from the operating system). Stubs replace all subordinate components. Successively, integration proceeds with lower level components. The higher level that has already been tested serves as test driver.

Advantage: Test drivers are not needed or only simple ones are required, because the higher–level components that have already been tested serve as main part of the test environment.

Disadvantage: Lower level components not yet integrated must be replaced by stubs. This can be very costly.

Bottom-up integration: The test starts with the elementary system components that do not call further components, except for functions of the operating system. Larger subsystems are assembled from the tested components and then these integrated parts are tested.

Advantage: No stubs are needed.

Disadvantage: Higher-level components must be simulated by test drivers.

Ad hoc integration: The components are being integrated in the (casual) order in which they are finished.

Advantage: This saves time, because every component is integrated as early as possible into its environment.

Disadvantage: Stubs as well as test drivers are required.

Backbone integration strategy

A skeleton or backbone is built into which components are gradually integrated

Advantage: Components can be integrated in any order.

Disadvantage: Labor intensive skeleton or backbone is required.

d Explain the two specific types of tests related to software changes.

- Confirmation testing(re-testing)
  - When a test fails,a new version of s/w is released
  - Test again to confirm defect has been fixed
  - Same input,data and environment
- Regression Testing
  - Unexpected side-effect
  - Executing test cases that have executed before
  - Test cases probably passed the last time they were executed(difference in confirmation and regression testing)
  - Anti-regression because we are executing tests with the intent of checking that the system has not regressed
  - To verify that modifications in the software or the environment have not caused unintended adverse side effects and that the system still meets its requirements
  - Executed whenever software changes
    - As a result of fixes or new or changed functionality
    - Environment changes
    - Eg:new version of DBMS ,new version of compiler…

3 Attempt *any two* of the following: **10**

a Explain the different aspects of code structure to be consider during static analysis ?

- Control flow structure;
  - Addresses the sequence in which the instructions are executed.
  - Reflects the iterations and loops in a program's design
  - Used to identify unreachable (dead) code.
  - Code metrics relate to the control flow structure,
    - e.g. number of nested levels or cyclomatic complexity.
- Data flow structure;
  - Trail of a data item as it is accessed and modified by the code
  - How the data act as they are transformed by the program.
  - Defects can be found such as referencing a variable with an undefined value and variables that are never used.
- Data structure.
  - Refers to the organization of the data, independent of the program.
  - When data is arranged as a list, queue, stack, or other well-defined structure, the algorithms for creating, modifying or deleting them are more likely to be well-defined.
  - Provides information about the difficulty in writing programs to handle the data and in designing test cases to show program correctness.

b Briefly explain the planning and kick-off meeting phases of formal review.

Planning

- ✓ 'Request for review' by the author to the **moderator**
- ✓ Moderator take care of the scheduling (dates, time, place and invitation) of the review
- ✓ The entry check is carried out to ensure that the reviewers' time is not wasted
- ✓ A document containing too many obvious mistakes is clearly not ready to enter a formal review process

Minimum set for performing the entry check:

• A short check of a product sample by the moderator (or expert) does not reveal a large number of major defects.
• The document to be reviewed is available with line numbers.
• The document has been cleaned up by running any automated checks that apply.
• References needed for the inspection are stable and available.
• The document author is prepared to join the review team and feels confident with the quality of the document

- ✓ Decide which part of the document to review
- ✓ Looking at documents from different perspectives may be more effective than an unfocusedreview.
- ✓ Not to look at the whole document, but to prioritize parts with the highest risk, or to review samples only
- ✓ For a review, the maximum size is usually between 10 and 20 pages.
- ✓ The composition of the review team.
- ✓ The team normally consists of four to six participants, including moderator and author. To improve the effectiveness of the review, different roles are assigned to each of the participants.

Kick-off

- ✓ Get everybody on the same wavelength regarding the document
- ✓ under review and to commit to the time that will be spent on checking
- ✓ Entry check and defined exit criteria are discussed
- ✓ Motivation of reviewers
- ✓ Reviewers receive a short introduction on theobjectives of the review and the documents
- ✓ The relationships between the document under review and the other documents are explained
- ✓ Role assignments, checking rate, the pages to be checked, process changes and possible other questions are discussed
- ✓ The distribution of the document under review, source documents and other related documentation

c Explain the goals and characteristics of inspection.

Goals of Inspection

- help the author to improve the quality of the document under inspection;
- remove defects efficiently, as early as possible;
- improve product quality, by producing documents with a higher level of quality;
- create a common understanding by exchanging information among the inspection participants;
- train new employees in the organization's development process;

• learn from defects found and improve processes in order to prevent recurrence of similar defects;

• sample a few pages or sections from a larger document in order to measure the typical quality of the document, leading to improved work by individuals in the future, and to process improvements.

Characteristics of Inspection

• It is usually led by a trained moderator (certainly not by the author).
• It uses defined roles during the process.
• It involves peers to examine the product.
• Rules and checklists are used during the preparation phase.
• 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.
• A formal follow-up is carried out by the moderator applying exit criteria.
• Agh causal analysis step is introduced to address process improvement issues and learn from the defects found.
• Metrics are gathered and analyzed to optimize the process

d What is static testing?List the objectives and advantages of static testing on software work products.

- Static testing:Testing of a component or system at specification or implementation level without execution of that software.
- Objective
  - To find Deviations from standards, missing requirements, design defects, non-maintainable code and inconsistent interface specifications.
  - Finds defects rather than failures.
  - The objectives of reviews are informational, communicational and educational, whereby participants
  - learn about the content,help them understand the role and to plan for future stages of development.
  - Reviews represent project milestones, and support the establishment of a baseline for a software product.
  - Customer/user communication
- Advantages
  - Early feedback on quality issues can be established
  - Rework costs are most often relatively low and thus a relatively cheap improvement of the quality of software products can be achieved.
  - Development productivity figures are likely to increase.
  - There is an exchange of information between the participants.
  - Static tests contribute to an increased awareness of quality issues.

4 Attempt *any two* of the following: **10**

a Explain cause-effect table testing with the help of an example.

- Decision table
- A table showing combination of i/ps and /or stimuli (causes) with their associated outputs and/or actions (effects) ,which can be used to design test cases
- Making the choice of which combinations to test and which to leave out
- Aid the systematic selection of test cases
- Finding problems and ambiquities in specification
- Work well in conjunction with EP
- Identify a suitable function or subsystem that has a behavior which reacts to combination of inputs or events
- Decision tables can be used for:

a. Specifying complex program logic
b. Generating test cases (Also known as *logic-based testing)*

Conditions - *(Condition stub)*	Condition Alternatives – *(Condition Entry)*
Actions – *(Action Stub)*	Action Entries

- Each condition corresponds to a variable, relation or predicate
- Possible values for conditions are listed among the condition alternatives
    - Boolean values (True / False) – Limited Entry Decision Tables
    - Several values – Extended Entry Decision Tables
    - Don't care value
    - Each action is a procedure or operation to perform

(Any one example)

b Write short note on test case specification and test procedure specification.

Test case specification

- ❖ A document specifying a set of test cases (objective, inputs ,test actions ,expected results, and execution preconditions) for a test item
- ❖ Comparing "what is" and "what ought to be"
- ❖ What the system should do
    - ❖ Information about correct behaviour of system
    - ❖ Oracle or test oracle
- ❖ After choosing input value,the tester needs to determine expected o/p
- ❖ O/p may be displaying on screen/changes to data and/or states /other consequences

IEEE 829 STANDARD: j
TEST CASE SPECIFICATION TEMPLATE
[

Test case specification identifier	Output specifications
Test items	Environmental needs
Input specifications	Special procedural requirements
	Intercase dependencies

Test procedures specification

- ❖ The document specifying a sequence of actions for the execution of a test
- ❖ Manual test script
- ❖ Automation script
- ❖ Describe the instruction to a test execution tool
- ❖ Written in a programming language that the tool can interpret

IEEE 829 STANDARD:
TEST PROCEDURE SPECIFICATION TEMPLATE

Test procedure specification identifier
Purpose
Special requirements
Procedure steps

c Explain boundary value analysis with the help of an example.

- A black-box test design technique in which test cases are designed based on boundary values.
- Boundary value is an input value which is on the edge of an equivalence partition(minimum and maximum value of a range)

- Both valid boundaries (in the valid partitions) and invalid boundaries (in the invalid partitions)

(Any one example)

d What is decision coverage ?With the help of an example show that 100% decision coverage guarantees 100% statement coverage ,but vice versa is not true.

Decision Coverage is also known as Branch Coverage.

Whenever there are two or more possible exits from the statement like an IF statement, a DO-WHILE or a CASE statement it is known as decision because in all these statements there are two outcomes, either TRUE or FALSE.

With the loop control statement like DO-WHILE or IF statement the outcome is either TRUE or FALSE and decision coverage ensures that each outcome(i.e TRUE and FALSE) of control statement has been executed at least once.

Alternatively you can say that control statement IF has been evaluated both to TRUE and FALSE.

The formula to calculate decision coverage is:

Decision Coverage=(Number of decision outcomes executed/Total number of decision outcomes)*100%

Decision coverage is stronger that statement coverage and it requires more test cases to achieve 100% decision coverage.

Eg:

READ X
READ Y
IF X > Y
PRINT X is greater that Y
ENDIF

To get 100% statement coverage only one test case is sufficient for this pseudo-code.

TEST CASE 1: X=10 Y=5

However this test case won't give you 100% decision coverage as the FALSE condition of the IF statement is not exercised.

In order to achieve 100% decision coverage we need to exercise the FALSE condition of the IF statement which will be covered when X is less than Y.

So the final TEST SET for 100% decision coverage will be:

TESTCASE1:X=10,Y=5
TEST CASE 2: X=2, Y=10

Note: 100% decision coverage guarantees 100% statement coverage but 100% statement coverage does not guarantee 100% decision coverage.

5 Attempt *any two* of the following: **10**

a What is configuration management?How configuration management supports testing?

Configuration management is a discipline applying technical and administrative direction and surveillience to identify and document the functional and physical characteristics of a configuration item ,control changes to those characteristics ,record and report change processing and implementation status ,and verify compliance with specified requirements.

- Configuration management is in part about determining clearly what the items are that make up the software or system
- It allows the testers to manage their testware and test results using the same configuration management mechanisms
- Supports the build process, which is essential for delivery of a test release into the test environment
- Allows us to map what is being tested to the underlying files and components that make it up

- *Against something,* **version controlled**
- We must be able to trace the test results back to what exactly we tested
- when testers receive an organized, version-controlled test release from a change-managed source code repository, it is accompanied by a test item transmittal report or release notes.

b Explain the typical factors that influence the effort related to testing.

- Test strategies or approaches
- Product factors
- Importance of non-functional quality characteristics
- Complexity is another major product factor•
    - The difficulty of comprehending and correctly handling the problem the system is being built to solve
    - The use of innovative technologies
    - The need for intricate and perhaps multiple test configurations-rely on arrival of scare software,h/w and other supplies
    - The prevalence of stringent security rules
    - The geographical distribution of the team
- Increases in the project and project team increases the difficulty of predicting and managing them
- Process factors include the availability of test tools
- The life cycle itself is an influential process factor --- V-model
- Process maturity, including test process maturity which reduces test execution cost.
- Time pressure is another factor to be considered
- People factors
- The test results are important in the total amount of test effort during test execution

c What is the purpose and importance of test plan in software testing? Summarize the content of test plan according to 'Standard for Software Test Documentation'.

- The purpose and substance of test plans
    - Writing a test plan guides our thinking
    - Writing a test plan forces us to confront the challenges that await us
    - Focus our thinking on important topics
- Serve as vehicles for communicating with other members of the project team
    - Test scope, objectives and critical areas to test;
    - Project and product risks, resource considerations and constraints;
    - Testability of the item under test.
- The test plan becomes a record of previous discussions and agreements between the testers and the project team
- Written test plans give us a baseline against which to measure revisions and changes

Test Plan according to IEEE 829

1. Test plan identifier
2. Introduction
3. Test items
4. Features to be tested
5. Features not to be tested
6. Approach
7. Item pass/fail criteria (test exit criteria)
8. Suspension criteria and resumption requirements
9. Test deliverables
10. Testing tasks
11. Environmental needs
12. Responsibilities
13. Staffing and training needs
14. Schedule
15. Risk and contingencies
16. Approvals

d What is risk?Explain the different classifications of risk in software testing.

- Risk
  - Possibility of a negative or undesirable outcome
  - It is a possibility, not a certainty
  - Classifications-Product risk and project risk
- Product risk
  - System or software might fail to satisfy some reasonable customer, user, or stakeholder expectation
  - Unsatisfactory software might omit some key function that the customers specified
  - Unsatisfactory software might be unreliable and frequently fail to behave normally.
  - Unsatisfactory software might fail in ways that cause financial or other damage
  - Unsatisfactory software might have problems related to a particular quality characteristic
- Project risk
  - A risk related to management and control of the project
    - Direct risks such as the late delivery of the test items to the test team or availability issues with the test environment.
    - Indirect risks such as excessive delays in repairing defects found in testing or problems with getting professional system administration support for the test environment.
- For any risk, product or project, you have four typical options:
  - Mitigate
  - Contingency
  - Transfer
  - Ignore

6 Attempt *any two* of the following: **10**

a Explain the characteristics of test management tools.

Features or characteristics of test management tools include support for:

- management of tests (e.g. keeping track of the associated data for a given set of tests, knowing which tests need to run in a common environment, numberof tests planned, written, run, passed or failed);
- scheduling of tests to be executed (manually or by a test execution tool);
- Management of testing activities (time spent in test design, test execution,whether we are on schedule or on budget)
- Interfaces to other tools, such as:
  - test execution tools (test running tools);
  - incident management tools;
  - requirement management tools;
  - configuration management tools;
- Traceability of tests, test results and defects to requirements or other sources;

- Logging test results (note that the test management tool does not run tests, but could summarize results from test execution tools that the test management tool interfaces with);
- Preparing progress reports based on metrics (quantitative analysis), such as:
    - tests run and tests passed;
    - incidents raised, defects fixed and outstanding.

b Explain the different levels of scripting used in test execution tools.

There are different levels of scripting. :

• linear scripts (which could be created manually or captured by recording a manual test);
- The script doesn't know what the expected result is until you program it in -it only stores inputs that have been recorded, not test cases.
- • A small change to the software may invalidate dozens or hundreds of scripts.
- • The recorded script can only cope with exactly the same conditions as whenit was recorded. Unexpected events (e.g. a file that already exists) will not be interpreted correctly by the tool.

• structured scripts (using selection and iteration programming structures);
• shared scripts (where a script can be called by other scripts so can be re-used- shared scripts also require a formal script library under configuration management);
• **data-driven scripts** (where test data is in a file or spreadsheet to be read by a control script);
• **keyword-driven scripts** (where all of the information about the test is storedin a file or spreadsheet, with a number of control scripts that implement the tests described in the file).

c List the benefits and risks of test automation and tool support for testing?

Potential Benefits

• reduction of repetitive work;
• greater consistency and repeatability;
• objective assessment;
- Using a tool means that subjective bias is removed and the assessment is more repeatable and consistently calculated.

• ease of access to information about tests or testing.

Potential Risks

• unrealistic expectations for the tool;
- have clear objectives for what the tool can do and that those objectives are realistic

• underestimating the time, cost and effort for the initial introduction of a tool;
- technical problems and resistance from other people need to be addressed

• underestimating the time and effort needed to achieve significant and continuing benefits from the tool;
• underestimating the effort required to maintain the test assets generated by the tool;
• over-reliance on the tool.

d Define the following terms

i)Stub ii)Driver iii)Static analyzer iv)Debugging v)Coverage tool

i. Stub-A skeletal or special purpose implementation of a software component used to develop and test a component that calls or is otherwise dependent on it.It replaces a called component
ii. Driver-A software component or test tool that replaces a component that takes care of the control and /or the calling of a component or system
iii. Static analyzer-A tool that carries out static analysis

iv. Debugging-The process of finding, analyzing and removing the causes of failures in software

v. Coverage tool-A tool that provides objective measures of what structural elements have been exercised by a test suite.

7 Attempt *any three* of the following: **15**

a State the principles of software testing.Explain any two in detail.

Principle 1 - Testing shows presence of defects

Testing reduces the probability of undiscovered defects remaining in the software but, even if no defects are found, it is not a proof of correctness.

Principle 2: Exhaustive testing is impossible

Testing everything (all combinations of inputs and preconditions) is not feasible except for trivial cases.

Principle 3: Early testing

Testing activities should start as early as possible in the software or system development life cycle and should be focused on defined objectives.

Principle 4: Defect clustering

A small number of modules contain most of the defects discovered during prerelease
testing or show the most operational failures.

Principle 5: Pesticide paradox

If the same tests are repeated over and over again, eventually the same set of test cases will no longer find any new bugs. To overcome this 'pesticide paradox', the test cases need to be regularly reviewed and revised, and new and different tests need to be written to exercise different parts of the software or system to potentially find more defects.

Principle 6: Testing is context dependent

Testing is done differently in different contexts

Principle 7 - Absence of errors fallacy

Finding and fixing defects does not help if the system built is unusable and does not fulfill the users' needs and expectations

b Define the following characteristics of software
i)Reliability ii)Interoperability iii)Portability iv)Usability v)Maintainability

i. Reliability
   - Perform its required function under stated conditions for a specific period of time ,or for a specified number of operations

ii. Interoperability
   - The capability of software product to interact with one or more specified components of system

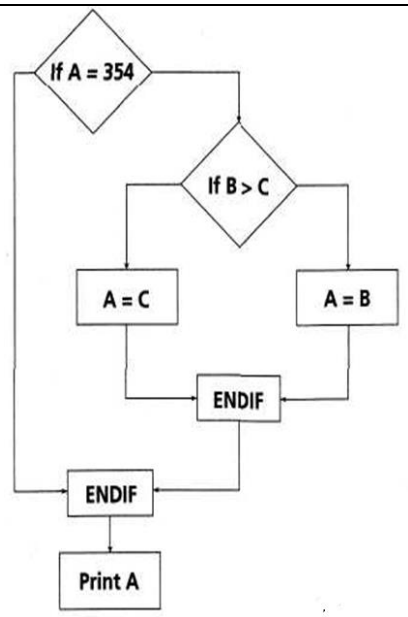
iii. Portability
   - Transferred from one hardware or software environment to another

iv. Usability
   - To be understood ,learned and used and to be attractive to the user when used under pecified conditions

v. Maintainability
   - Can be modified to corret defects,modified to meet new requirements,modified to make future maintenance easier,or adapted to a changed environment

c What is cyclomatic complexity?With the help of an example, explain two ways to calculate cyclomatic complexity.

- Complexity metrics identify high risk, complex areas.
- The cyclomatic complexity metric is based on the number of decisions in a program.It is the number of independent paths through the program.
- Independent path is defined as a path that has at least one edge which has not been traversed before in any other paths.
- Cyclomatic complexity=sum the number of binary decision statements (e.g. if, while, for, etc.) +1
- Cyclomatic complexity=L-N+2P where L is the number of edges in a graph, N is the number of nodes a graph and P is the number of disconnected pats of a graph

IF A = 354 THEN IF B > C THEN A = B ELSE A = C ENDIF ENDIF Print A cyclomatic complexity is 8-7 + 2 = 3	 cyclomatic complexity is 2 + 1 = 3.

d Briefly explain the different factors that influence the selection of appropriate test design technique in software testing.

- ✓ The internal factors that influence the decision about which technique to use are:
  - *Models used*
    - *State transition diagram -state transition testing*
  - *Tester knowledge I experience*
  - *Likely defects*
    - *Each technique is good at finding a particular type of defects*
  - *Test objective*
    - *Gain confidence-use case*
    - *Thorough testing-structure based testing*
  - *Documentation*
  - *Life cycle model*
    - *Sequential life cycle model-formal techniques*
    - *Iterative life cycle model-exploratory testing*
- ✓ The external factors that influence the decision about which technique to use are:
  - *Risk*
    - *Greater the risk ,mor e thorough and formal testing*
  - *Customer I contractual requirements*
  - *Type of system*

        - *Financial application –boundary value analysis*
    - *Regulatory requirements*
        - *Aircraft industry-BVA,EP,STT, together with statement and decision or modified condition decision coverage*
    - *Time and budget*

e List the major types of test approaches.Explain any two in detail.

- Analytical:
    - start with analysis as a foundation
    - Includes two of the most common strategies
        - Requirements-based testing
        - Risk-based testing
    - Analyze the test basis to identify the test conditions
    - The risk-based strategy involves performing a risk analysis using project documents and stakeholder input, then planning, estimating, designing, and prioritizing the tests based on risk.
    - The requirements-based strategy, where an analysis of the requirements specification forms the basis for planning, estimating and designing tests.
- Model-based:
    - Build mathematical models for loading and response for e-commerce servers, and test based on that model.
    - If the behavior of the system under test conforms to that predicted by the model, the system is deemed to be working
- Methodical:
    - Have a checklist industry-standard for software quality, such as ISO 9126,
    - Methodically design, implement and execute tests following this outline.
    - Developed in-house, assembled from various concepts developed inhouse and gathered from outside, or adapted significantly from outside ideas
- Process- or standard-compliant
    - Have in common reliance upon an externally developed approach to testing, often with little - if any – customization
- Dynamic (Reactive)
    - Exploratory testing,
    - Have in common concentrating on finding as many defects as possible during test execution and adapting to the realities of the system under test as it is when delivered, and they emphasize the later stages of testing.
- *Consultative Test Strategies*
    - Rely on the input of one or more key stakeholders
    - External stakeholders determine test conditions to cover
    - Stakeholders have complete control over conditions
- Regression-averse
    - Have in common a set of procedures - usually automated - that allow them to detect regression defects.

f. What are the important factors to be considered in selecting a tool? State the objectives for a pilot project for a new tool.

The following factors are important in selecting a tool:

• assessment of the organization's maturity (e.g. readiness for change);

• identification of the areas within the organization where tool support will help to improve testing processes;

• evaluation of tools against clear requirements and objective criteria;
• proof-of-concept to see whether the product works as desired and meets the requirements and objectives defined for it;
• evaluation of the vendor (training, support and other commercial aspects) or open-source network of support;
• identifying and planning internal implementation (including coaching and mentoring for those new to the use of the tool).

The objectives for a pilot project for a new tool are:

- to learn more about the tool (more detail, more depth);
- to see how the tool would fit with existing processes or documentation, howthose would need to change to work well with the tool and how to use the tool to streamline existing processes;
- to decide on standard ways of using the tool that will work for all potential users (e.g. naming conventions, creation of libraries, defining modularity, where different elements will be stored, how they and the tool itself will bemaintained);
- to evaluate the pilot project against its objectives (have the benefits been achieved at reasonable cost?).