Q.1 Attempt any TWO questions. [10]
Q.1(a) What is an error, failure and fault? [5]

(A) • A failure is non-fulfillment of requirements, A discrepancy between actual result or behavior (Identified in Requirement Specification)
• Failure in a software means the software doesn’t perform functional requirements, the failure occur in S/W because of fault. Every fault, or Defect or Bug in S/w is present since it was developed or changed.
• A failure has its roots in a fault in the software. These fault is also called a defect or internal error programmer generally use a term Bug for fault e.g. of fault e.g. of fault can be the programmer might be wrongly programmed or forgotten code in application.
• It is possible that a fault is hidden by one or more other faults in different parts of application. In that case, A failure only occurs after the masking defects have been corrected.
  
  The cause of fault or defect is an error or mistakes by person. E.g. wrong programming by developer or A mescindersfanding of commands in programming language.
  
  However faults may be even cause by environmental conditions like radiation, magnetism etc. that introduce Hardware problem.

Q.1(b) Explain success factors of review process. [5]

(A) Implementing formal reviews is not easy as there is no one way to success and there are no. of ways to fail. Critical success factor for formal reviews are
  1. Find a champion
  2. Pick things that really counts
  3. Explicitly planned & track review activities
  4. Trained participants
  5. Manage peoples issues
  6. Follow the rules but keep it simple
  7. Continuously improve process & tools.
  8. Report results
V.1(c) Explain V-model.

(A)

V-model was developed to address some of problem experience using traditional waterfall approach defects were being found to late in life cycle testing was not involved in until end of project testing also added load time due to its late involvement.

The V-model provides guidance that testing needs to begin as early as possible in life cycle.

V-model shows testing is not only execution base activity there are variety of activity that need to perform before end of coding phase.

These activities should be carried out in parallel with development activities and tester needs to work with developer, so they can perform these activity and produce a good activity result.

Therefore in V-model s/w tester is involved in development team from day-1.

V-model was uses 4 test levels, each with there own objectives.

Q.1(d) Explain decision table Based testing with example.

(A)

Decision Table base Testing:

It has been use to represent & analyze & complex logical relationship decision table are Idle for describing situation in which no of combinations of actions are taken under verifying set of conditions.

A decision table has 4 portions.

The leftmost column is stub portion, and to the right of it is entry portion.

The condition portion is denoted by condition & action portion is denoted by expected o/p.

Consider e.g. of triangle function.

Condition stab
Q.2 Attempt any TWO questions. [10]

Q.2(a) What is testing? Explain fundamental principles of testing. [5]

(A) Testing is an umbrella activity conducted throughout life cycle of software to identify bugs or defects or faults, present in software.

Testing can be identify both syntactical as well as logical error present in software.

General principles of testing are:
1) Testing shows the presence of defects, not their absence.
2) Exhaustive testing is not possible.
3) Testing activity should start as early as possible.
4) Defects tend to cluster together (Group of similar elements)
5) The pesticide paradox
6) The test is context depended.
7) False of assuming that no failure means useful system.
8) An efficient testing will conducted by independent third party i.e. professional software testor groups.
9) Test cases require for testing a software must be planned long before testing begins and it must be traces sable to costumer requirement which extend from requirement document.

Q.2(b) Define software. Explain the characteristic of software. [5]

(A) Instruction: Any programmable excitable statement is called Instructions.

A set of instruction which are executable sequentially to get desired output is called program.

A set of programs which when executed sequentially to get desired result could software.
Characteristics of Software:
1) Software cannot be manufactured, it is always developed in classical sense.
2) Software does not wear out i.e. software can be upgraded or modified.
3) Availability: Software should be easily available to all users.
4) Reliability
5) Flexibility
6) Security
7) Usability
8) Maintainability
9) Portability

Q.2(c) Explain psychology of software testing.

Psychology of Testing:

1) The people make mistakes but they do not like to admit them. One goal of testing software is uncover disturbing between software and specification or customer needs, therefore failure found must be reported to developer.
2) Can developer test his own program?, it is an important & frequently asked question. The universal valid answer does not exist, if tester also author of program, they must examine their own work critically.
3) If the developer implements a fundamental design error e.g. if they misunderstand the conceptual formation then it is possible that he or she may not find these using their own test.
4) On the other hand it is an advantage to have a good knowledge of own test object it is not necessary to learn test object and therefore time is saved, Management has to decide when it is the advantage to save time even with the disadvantage of blindness for their own errors.
5) Independent testing team tends to increase quality & comprehensiveness of the test tester can take at the test object without Bias (partiality). It is not their product and possible assumption & misunderstanding of the tester. The tester must acquire necessary knowledge of the test object in order to create a test cases with corresponding time and cost. The tester comes along with deeper testing knowledge, developer which doesn't have or must first acquire.
6) It is job of the tester to report failure and disturbances observe to management. The manner of the reporting can contribute the cooperation between developer and tester.
7) There is often problem that failures found during testing are not reproducible for developers in development environment.
8) Mutual knowledge of their respective task encourages co-operation between tester and developer. Developer should know basics of testing and tester should have basic knowledge of software development.

Q. 2(d) Explain fundamental test process. [5]

(A) Fundamental test process:

- Planning and Analysis and design

Implementation and
Execution

Evolution of the test exit criteria

Post testing activity

Control

Begin

end
Q.3 Attempt the following (any TWO) [10]
Q.3 (a) Explain waterfall model with its advantages and disadvantages. [5]

(A) The WATERFALL MODEL was one of the earliest model to be designed. It is one of the systematic & sequential approach towards software development. The waterfall model develops software using Analysis, Analysis Design, Coding, Testing, Deployment & maintenance.

The arrows in these model are unidirectional. It assumes that developer shall get all requirements in a single attempt. Therefore, these model is suitable only for static requirements.

Once all requirements are analyzed and gathered from customer, the requirements are converted into Low Level and High Level Design. The design can be Logical Flow, Diagramatic Represent Flowchart, Data Flow Diagram of software. The actual conversion of software design into program code is done in software coding. Coding is directly depended on system Design. As the design is simple, the coding is also easier. The programmer selects appropriate programming language and start software coding.
Executable are tested as per the test plan in system testing. The software are executed to find bugs defects present if any
Finally successful software is deployed to the costume and future activities are handled through maintain advantages:
1) It is one of the oldest systematic & sequential approach towards software development.
2) All the phases of software development are clearly defined i.e., Each team member should know when his/her responsibility in development process.
3) It is one of the time consuming model because ones requirements are gathered from costumer, the work model is available only after the deployment. Costumer needs to specify all their requirements software development but it is often difficult for any costumer to give all their requirements explicitly.

Q.3 (b) Explain different levels of testing. [5]

(A) According to V–model of Software Development there four different levels of testing
1) Unit/Component/Module Testing :
   - A component testing is also known as Unit, module & program testing, it searches for defects and verifies functionality of software that are separately testable.
   - In unit testing, the entire software is divided into smallest workable module called as unit and each a every unit is tested separately to ensure that it is working properly.
   - Advantages of Unit Testing are
     a) Error identification become simple
     b) Error propagation gets minimize
     c) Overall testing time will reduce
   - The following test are performed on an individual module during unit testing.

     Unit Testing Consideration
     Module or Unit or Component
     BBT { Boundary Condition Testing
     WBT { Local Data Structure
     Independent Path Testing
     Event Handler

     a) In Boundary Condition Testing Individual module is tested on their extremes boundaries, because maximum number of errors do occurs on boundaries rather than its operation bound.
b) Local data structure testing ensures the validity of data structure in software module.

c) In independent path testing tester tests all the independent path in module are working properly or not.

d) In interface testing, tester tests the interface of module with the system.

e) Finally all error handling paths are tested to ensure that the module is working properly as an individual.

- Component testing may include testing of functionality & specific non-functional characteristics such as resource behavior, performance testing, & structural testing test cases are derived from the driver class to test their stuffs.

Unit testing is also called as Grey Box Testing (GBT)

2) Integration Testing :

- Once all the modules are tested properly. Tester needs to integrate them as per the design of software to ensure that it will work properly as whole.

- There are two types of Integration testing i.e. Top down Integration, Bottom up Integration.

- In top down integration one by one module gets added in system where as in bottom up integration is conducted cluster by cluster integration.

- Regression testing is perform along with integration testing to ensure that system will perform properly whenever there is any change in software.

- Integration testing test interfaces between components, interaction, to different parts of system such as an o. s., file system & interfaces between software & hardware.

- Integration testing is after carried out by integrator but preferably by specific integration tester or testor team depending on structural requirement of system. Integration testing can be categorized as follows

a) Top Down Integration:

In these, testing takes place from top to better following the control flow or architectural structure e.g. starting from GUI or main menu.

Components or system are substituted by staff.

b) Bottom Up Integration

Bottom up testing takes place form bottom of control flow upwards.
Components or system are substituted by drivers. (Cluster by cluster testing)
c) Functional Incremental : Integration & testing takes place on the basis of functional specification.

3) **System testing** :
- It is concerned with behavior of whole system or product as defined by the scope of development project or product
- System testing may include test based on requirement specification, business process, use cases, system behavior, interactions with operation system & system resources.
- System testing is most often the final test on behalf of development to verify that the system to be delivered meets specification and its purpose may be to find as many defects as possible.
- Most oftenly system testing is curried out by specialist testor from independent third party
- System testing should investigate both functional & Non–functional requirements of the system. Typically nonfunctional requirement includes performance & reliability system testing of functional requirement starts by using most appropriate specification based technique (BBT) for aspect of the system to be tested.

- The following are testing to be conducted during system testing
  1. Performance testing
  2. Load/Stress
  3. Recovery
  4. Security testing

4) **Acceptance Testing** :
- When development organisation has perform its system test & has corrected or all most defects, the system will delivered to user or costumer for acceptance testing these test should answer following question such as
  a) Can system be released?
  b) What if any are the outstanding risk?
  c) Has development meet their expectations?
- It is most often the responsibility of user & costumer, although other stoke holder may be involves as well. The goal of acceptance testing is to be established a confidence in the system, part of the system or specific non–functional characteristics.
  e.g. Useability of system
Acceptance testing is most then focused on a validation type of testing, where by we are trying to determine whether the system is fit for the purpose. Finding defects should not be main focus of acceptance testing if generally perform on s/w to decide whether s/w is simply accepted or rejected.

Q.3 (c) Explain functional and non functional testing. [5]

(A) A test type is focused on a particular test object which could the testing of function to be performed component or system, A non–functional quality characteristic such as reliability useability; A structure or architecture of system or component; or related to change i.e. confirming the defects have been fixed and looking for unintended changes. i.e. Regression Testing. Depending on test objectives, a different testing will be organised.

1. Testing of function [functional testing] The function of system is “what is does.” This is typically describe in requirement specification, functional specification & in use cases. There may be some functions that are assumed to be provided that are not documented as well as not the part of requirement for system functional tests are based on these functions, describe in documents or understood by the tester. Functional testing considers the specified behaviour & is often also referred as Black Box testing or Behavioural or Performance testing. Testing functionality can be done from two prospective Requirement based or Business Process Based.

A) Req. Based testing uses a specification of the functional Requirements for the system as the basis for designing test cases. A good way to start is to use the table of content of the requirement specification as an initial test inventory or list of items to test. As a tester we should also priorities a requirement based on risk criteria and use these to prioritize the test. These will ensure that the important & critical test are included in testing efforts.

(B) Business Process Base testing uses knowledge of business processes. Use cases are very useful basis for test cases for business prospective. The technique use for functional testing are often specification based but experience based technique can be used as a part of test design in functional testing, a model may be developed such as state transition model re use case model.

1. A second target for testing is testing of the quality characteristics or non-functional attributes of the system here we are interested in “How well or fast something is done”. We are testing something that we need to measure on scale of measurement e.g. time to respond. Non-functional testing, as functional testing is performed at all test levels. Non-Functional testing includes performance testing, load, stress usability, maintaence, reliability & portability testing. It is the testing of “How well” the system work.

To the characteristics & sub characteristics of the software to be tested under non-functional testing are as follows:

1. Functionality Sub-characteristics:
   - Suitability, Accuracy, Security, Interoperability, complaints. These deals with functional testing.
2. reliability – Sub characteristics –
   - Maturity, facett Tolerance, Recoverability & complaints.
3. Usability – Sub characteristics –
   - Understandability, Learnability, Operability, attractiveness.
4. Efficiency – Which is divided into Behavior & resource Utilisation.
5. Maintainability : Sub characteristics :
   - Analyzability, Changeability, Stability, Testability,
6. Portability : Sub characteristics :
   - Adaptability, install ability, Co-existence, Replaceability.

Q. 3(d) Explain maintenance testing. [5]

(A) Maintenance testing:

1) Once deployed, system is often in service for year or even decades during this time the system & it’s operational environment is often corrected, change or extended. Testing i.e. executing during these life cycle period is called Maintainance Testing.

2) Maintainance testing is different from maintainability testing, which defines how easy it is to maintain the system.

3) The development and test process applicable to new developments doesn’t change fundamentally for maintainance purpose the same test process steps will be apply & depending on size & risk of the changes made, several levels of testing are carried out during maintainance testing. A component test, system test, Acceptance Test.

4) A maintainance test process useally begins with the receipt of an application for a change. The test manager will use these as basis for producing test plan.
On the receipt of new or change the specification, corresponding test cases are specified or adapted. Once the necessary changes have been made, regression testing is performed.

Usually maintenance testing will consist of two parts

i) Testing Changes
ii) Regression Test to show that rest of system has not been affected by maintenance work. The maintenance testing will perform on software under following condition.

A) If customer or end user requires support or they are not understanding some of the functionality of s/w.
B) When developer wants to enhance / upgrade a software.
C) When any changes are informed by user.

Q.4 Attempt the following (any TWO) [10]
Q.4(a) Explain phases of review process. [5]

(A) Review process vary from informal to formal i.e. (well structured & Regulated)
Although inspection is most documented and formal review technique the review process consist of six main steps.

1) PLANNING
   i) The review process for particular review process begins with “Request for Review” by the author to moderator. The moderator is often assign to take care of scheduling if review that means the time, date, venue, agenda & invitation of review is made by moderator.
   ii) On a project level, the project planning needs to allow time for review & rework activities, providing engineers with time to thoroughly participate in review.
   iii) For more formal reviews, the moderator always perform an entry check & defines formal exit criteria.

2) KICK-OFF
   i) An optional step in review process in kick off meeting the goal of these meeting is to get everybody on same wavelength regarding document under review & to commit to time that will be spent on checking.
   ii) Also the result of entry check & exit criteria are discussed in case of more formal review.
   iii) Roll assignment, checking rate, the pages to be check process changes & possible other questions regarding formal reviews are also discuss during this meeting.
3) The participants who are individually on document under review using the related documents, procedure rules & checklist proper. The individual participant identify defects according to their understanding of document & role.

ii) All issues are recorded preferably using logging form spelling mistakes are recorded on document under review but not mentioned during review meeting.

iii) A critical success factor for thorough preparation is no. of pages checked per hour, these is called Checking Rate.

4) REVIEW MEETING:

   i) Logging Phase:
      - Discussion Phase
      - Decision Phase

   During logging phase, the issues e.g. defects that have been identified during preparation phase are mentioned phase by phase, reviewer by reviewer & are logged either by author or recorder.

   A separate person like recorder or scribe is present in logging phase to log defects encouraged during review process.

   To ensure process & efficiency, no real discussion, the items are logged & then handled in discussion phase. A detailed discussion on whether or not an issue is defects is not meaningful, as it is much more efficient to simply log it & proceed to next defect.

   During logging phase the focus is on logging as many defects as possible within certain time frame.

   To ensure these, moderator tries to keep good logging rate i.e. no. of defects logged per minute.

   As chairperson of discussion meeting the moderator takes care of peoples issues. All the defects logged in previous phase are discussed in detail during discussion phase.

   At the end of meeting a decision on document under review has to be made by participant based on formal exit criteria. The most important exit criteria is no. of critical or major effect is avg. no. of major or critical defect found per page.

5) REWORK:

   Based on defects detected, author will improve document under review step-by-step. Not every defect i.e. found leads to rework. It is the author's responsibility to judge if defects has to be fixed. If no. of
defects per page exceeds the exit criteria then only the rework should be conducted for author.

6) FOLLOW-UP:
The moderator is responsible for ensuring that satisfactory action have been taken on all logged defects, process improvement suggestion & change request. Although the moderator checks to make sure that author has taken action on all defects, it is not necessary for moderator to check all corrections in detail for more formal review type moderator checks only for complaints to exit criteria i.e. during follow-up moderator keeps track on those errors which are encountered during logging phase & their elimination process.

Q.4(b) Explain roles and responsibilities of members present in review process. [5]

(A) The participant in any type of formal review must have adequate knowledge of review process, the best & most efficient review situation occurs when participant gains. Some kind of adv. for their own work during reviewing. The following are the members & responsibilities in review process.

1) Moderator / Manager:
It is a chairperson of the entire review process. He/she determines incorporation with author, type of review, approach & composition of review team. The moderator is responsible of scheduling of review process.
Moderator prepares policy, plan objectives & review process.
The training of reviewer can conducted by moderator the moderator performs entry check & follow up on rework, in order to control, the quality of input & output of review process.
Moderator can also defines exit criteria.

2) Author:
Author is writer of the document under review, the authors goal should be learn to as much possible with regards to improving quality of document but also to improve his or her ability to write futures documents.
The author’s task is to eliminate under areas & to understand the defects found.
If authors document exceeds exit criteria then author must involved in the rework phase.

3) Scriber / Recorder:
During logging phase recorder has to record each defect mentioned and any suggestions for process improvements
Reviewer:
The task of reviewer is to check authors documents for defects the levels of domain knowledge or technical expertise needed by reviewer will depend on type of review & type of document which is under review. The reviews should be chosen to represent different prospective & rates in review process. In addition to document under review the material reviewer reviews includes source, standards, checklist etc. The sole responsibility is to review author's document & try to find as many defects as present in document.

Manager:
The manager is involved in reviews as he/she decides on execution of reviews, allocates time in project schedules & determines whether review process object have been met. The manager is also responsible for providing readymade tools, review training if requested by participants.

Q.4(c) Explain goals and key characteristics of walkthrough and [5] Inspection.

(A) Walkthrough:
1. Walkthrough is characterized by author of document under reviewing guiding participant through document and his or her thought process, to achieve common understanding & to gather a feedback.
2. Walkthrough is specially useful if people from outside s/w discipline are present, who are not used to or cannot easily understand s/w documents. The content of document is explain step by step by author, to rich the consequences on changes or together information. Within a walkthrough the author does most of the preparation the participants who are selected. From different departments & document in advance.
3. Walkthrough is specially useful for higher level document such as requirement specification & architectural documents.
4. In walkthrough the reviewer should go through all requirements & specification of product to be develop & if require any modifications or suggestion or new ideas will be given to another as feedback
   • The goals of walkthrough are
      1. To present document to stack holder both within an outside the s/w discipline in order to gather information regarding topic under documentation.
      2. To explain [knowledge transfer] & evaluate content of document
      3. Establish a common understanding of document.
      4. To examine & discuss validity of proposed solution &
      5. Viability of alternatives.
- Key characteristics of walkthrough
  1. The meeting is led by author often separate scribe is present.
  2. The senioria may use to validate content
  3. Separate permeating preparation for reviews is optional
  4. The current requirement of product compare with similar app.

Inspection:
1. Inspection is most formal review type, the document under inspection is prepared & check thoroughly before meeting, comparing work product with its sources & other referenced documents, & using rules & checklist as inspection is validation criteria (Acceptance testing)
   - Checklist mechanism is must
2. In inspection meeting defects found are logged & any discussion is postponed until discussion phased, these makes inspection meeting very efficient meeting
3. In inspection meeting all defects are discussed, solve, & make a decision on them whether to accept or reject inspection is performed by an inspector.
   Using checklist mechanism
- The goals of Inspections are
  1. To help author to improve the quality of documentation under inspection
  2. To remove defects efficiently, as early as possible.
  3. To remove defects efficiently, as producing documents with higher level of quality
  4. To train new employer in organisations development process.
- Key Characteristics of Inspection are
  1. It is usually led by trained moderator
  2. It uses define goal using process
  3. If involves peer to peer common to examines product
  4. Rules & checklist are use during preparation phase
  5. Defects found are documented in logging list or issued log
  6. Follows up is carried out by applying exit criteria.

Q.4(d) Explain static analysis by tools.
(A)  - Coding standards
     - Code Metric
     - Code Structure
Static analysis is an examination of req., design & code which differs from dynamic testing in no. of following ways
1. Static analysis is perform on requirement, design or code without actually executing s/w being examine
2. It is ideally perform before types of formal reviews
3. Static analysis unrelated to dynamic properties of requirements, design & code such as test coverage
4. The basic goal of static analysis is to find defects whether or not they may cause failures

The are three diff. methods to perform static analysis
1. Coding standard :
Checking of coding standards is the most wellknown feature during static analysis. The first action to be taken is to define a coding standard
The coding standards are platform dependent, usually a coding standard consist of set of programming rules, naming conventions & layout specification. It is recommended for s/w tester to adopt existing standards. The main advantage of coding standard is that it saves lot of time & efforts during testing. Main reason for adopting standard is that the readymade tools are available which supports that standard, so overall testing efforts will reduced.
2. Code metric :
When performing static code analysis useally information is calendared about structural attributes of code, frequency of comments in program, depth of nesting, cyclomatic complexity & LOC. Cyclomatic complexity metric is based on no. of decision in program. It is very important for testor because it provides indication of amount of testing necessary to practically avoid defects using cyclomatic complexity we can also get no. of independent paths in program the e.g. of code metric [refer notes]
3. Code structure :
There are many different kinds of structure used in static analysis but frequently used code structure are
1. Control Flow Structure :
   It addressed sequence of instruction during execution it provides how the program structure is executed when a tester run the program. It can also be used to identify unreachable (Dead) code.
2. Data flow structure
   It shows how the data acts as they are transform by programs.
3. Data structure
It refers to organization of data itself, independent of program when data is arranged as list, queue, stack or other well-defined data structure, algorithm for creating modifying or information about the data while it provides lot of information about the data while designing a test case.

Static analysis by tools is useful because of following reasons.
1. Early detection of defects prior to test execution
2. Early warning about suspicious aspects of the code, design or requirements.
3. Identification of defects not easily found in dynamic testing.
4. Improved maintainability of codes & design seen as engineers work according to documented standards & rules
5. Prevention of defects, provided that's engineers are willing to learn from their errors & continuous improvement is practiced.

Q.5 Attempt the following (any TWO) [10]

Q.5(a) Explain White Box Testing. [5]

(A) The basis for white box technique is the source code of the test object. These techniques are often caused code-based testing technique or structural testing technique.

Generic idea of white box technique is to execute every part of the code of test object at least once flow oriented test cases are identified, analyzing program logic & then executed. However the expected result should determine using requirement or specifications, not the code. These is done in order to decide if the execution resulted in failure.

The focus of examination of white box technique is on the statements of the test objects.

The primary goal of white box testing is then to achieve a previously defined coverage of the statements, e.g. executes are possible statements in the program.

The basic white box test design technique are as follows:
1) Statements Coverage
2) Branch Coverage
3) Path Coverage

Consider a e.g. of triangle problem (refer notebook)

1) Statement Coverage:
These analysis focuses on each statement of test object the test cases shall execute a predefined minimum quota or even all statements of test
objects. The first step is to translate source code into control flow graph. The graph makes it easier to specify in detail the control elements that must be covered in a graph the statements are represented as node a control flow between statement are represented by edges if sequences of unidirectional statement appear in program fragment then they are illustrated as one single node. Conditional statements like if….else & loops like while, do …..while have more than one edges going out from them. After execution of test cases it must be verified which of the statements have been executed. Statement coverage must ensure that each & every statement of a program must be executed at least once. The test completion criteria for s.c. can define as

\[ \text{State Coverage} = \frac{\text{No.of executable Stat.}}{\text{Total no.of stat.}} \times 100 \]

2) **Branch Coverage:**
A more advance criteria for white box testing is branch coverage of control flow graph.

e.g. Edges in the graph are Centre of tension. Execution of each statement is not consider during branch coverage but rather the execution of decisions are more important. The result of decision determines which statement is executed next. Branch coverage testing should make sure every decision with both possible outcomes i.e. true & false.

Test completion criteria for branch coverage is defined as

\[ \text{Branch Coverage} = \frac{\text{No.of executable branches}}{\text{Total no.of Branches}} \times 100 \]

3) **Path Coverage:**
Until now test case determination focused on statement or branches of control flow as well as complexity repetitions, the previous deliberations are not sufficient or not adequate test, path coverage requires execution of all different paths through test objects. The test completion criteria for P.C. cannot define mathematically because it will depends on no. of repetition of loop.

No. of path coverage of test object can be determine using cyclomatic complexity, which gives total no. of maximum paths available in control flow.
Q.5(b) Explain boundary value analysis and equivalence class partitioning. [5]

(A) 1) Equivalence class Partitioning:

It is very difficult to test entire software as whole because software is collection of set program. In equivalence partitioning s/w tester divide entire software into set of equivalence classes & each & every equivalence program is tested by using set of valid & invalid i/p condition. The domain of possible I/P data for each I/P data element is divided into equivalence classes.

[Equivalence class Partitioning]

An equi. Class is group of data values where tester assumes that test objects processes them in same way. The test of one representative of equivalence class is such as sufficient because it is assume that for any other I/P value for same equi. Class the test object will not show different variables.

Besides equi. Classes for correct I/P, those for incorrect I/P values must be tested as well. To test equi. Classes we have four diff. types of equivalence classes.

1) If continuous numerical domain is specified then create 1 valid & 2 invalid equivalence classes.

<table>
<thead>
<tr>
<th>I/P → Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 ≤ x ≤ 50</td>
</tr>
</tbody>
</table>

Valid → x = {20,21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50} |

Invalid → i) x < 20 |

ii) x > 50

2) If no. of values should entered then create / valid (all possible correct values) & 2 invalid equi. Classes (less & more than correct no.)

<table>
<thead>
<tr>
<th>I/P ...... Specific value</th>
</tr>
</thead>
<tbody>
<tr>
<td>a = 5</td>
</tr>
</tbody>
</table>

Valid → x ∈ 5 |

Invalid → x ∈ 5 or x < 5 |

iii) x > 5

3) If set of values is specifies where each value may possibly be treated differently then create one valid equivalence class for each value of set (containing exactly these values) & one additional invalid equi. Class (containing all possible other value).

<table>
<thead>
<tr>
<th>I/P → Member of set</th>
</tr>
</thead>
<tbody>
<tr>
<td>x = {a, e, i, o, u}</td>
</tr>
</tbody>
</table>

valid → x ∈ {a, e, i, o, u} |

Invalid → x ∈ {b, c, d, f, ..., z} |

4) If there is condition that must be fulfill then create 1 valid and 1 invalid equivalence class to test the condition fulfill or not fulfill.
I/P → Boolean Value
Valid → TRUE
Invalid → FALSE

in t a = 5, b = 3, c = 2
if (a > b & b & b > c)

Test Completion Criteria:
For equi. Partitioning can be define as
Equ. Class coverage = \( \frac{\text{No. of tested EC}}{\text{Total no. of EC}} \times 100 \)

2) Boundary Value Analysis:
It delivers a very reasonable addition to the test cases that have been identified by equi. Classes.
Faults often appears at the boundaries of equi. Classes. In boundary value analysis s/w tester emphasizes on boundary condition because maximum no. of errors are generally occurs at boundary rather that within operational bound. Testers assume that if software works properly on the boundary then it will definitely work within boundary condition.
The most of the errors in software lies on boundary because boundaries are often not defined clearly or programmers misunderstand them.
A test with boundary values usually discovers the failure the technique can only be applied if set of data which is in one equi. Class has identifiable boundaries.
Boundaries value Analysis checks borders of equi. Classes on every border, the exact boundary value & both nearest adjacent value (Inside / Outside Equi. Class) are tested.
If the I/P condition for the program is the range within close internal ([a , b]) then using B VA we have six test cases as follows.
1) \([a,b]\)
   \(a - 1, a, a + 1, b - 1, b, b + 1\)
2) An I/P file has restricted number of data records between 1 & 100 the test values should be 1,100,2,99.
3) If permitted no. of O/P values is to be rested, proceed just as with the no. of input values:
   If O/P of 1–4 data values are, test accounted, test cases are 1, 4, 0, 5
   Test completion criteria for Boundary value is define as
   \[ \text{BV coverage} = \frac{\text{No. of tested BV}}{\text{Total No. of BV}} \times 100 \]
Q.5(c) Explain experience based testing. [5]

(A) In experienced based technique, peoples knowledge, skills & backgrounds are prime contributor to test cases & test condition, the experience of both technical & business people is important, as they bring different prospective to test analysis & design process due to previous experience with similar system they may have insights into what could go wrong, which is very useful in testing.

There are two types of experience based technique.
1) Error guessing
2) Exploratory testing

1) **Error Guessing:**
   It is a technique that should always be use as complement to other more formal technique the success of error guessing is very much depend on skill of tester, as good tester knows where the defects are most likely to occur.
   Some people seem to be naturally good at testing as they have experience in testing of variety of software. Error guessing approach is used after more formal techniques that have been applied to same extent can be very effective.
   In these technique the tester is likely to gain better understanding of system what is does & how it does, & with these better understanding the tester is likely to be better at guessing ways in which the system may not work properly.
   There are no rules for error guessing.
   The tester is encourage to think of situations in which s/w may not be able to code.
   Typical conditions to try includes division by zero, blank I/P , empty files and wrong data I/P if anyone even says a system or environment in which s/w is not operate then error guessing is first methodology to operate software.
   By using error guessing tester can find no. of errors by his own experience or common sense approach. Error guess methods saves a lot of time during testing.

2) **Exploratory Testing:**
   It is hands on approach in which testers are involve in minimum planning & maximum test execution planning involves creation of test cases, a short declaration of scope, objectives & possible approaches to be used, where as test design & execution activities are performed in parallel without formally documenting test condition or test cases.
It is most useful when there are no or poor. Specifications & when time is severally limited it can be use as check on the formal test process by helping by ensure that most serious defects have been found.
In exploratory testing the cases prepared by s/w tester based on their past experience of testing similar application based software.

Q.5(d) Explain use case based and state transition testing. [5]

(A) Used Case Based Technique:
In order to detect requirements, use cases or business cases are describe these are then compiled into use case diagram, it is used to describe behavioral model of the system diagram consist of actors and uses cases where actors are active elements present in system and use cases are simply task or responsibilities of actor in given system.
Use case diagram serve the purpose of defining requirement on a relatively abstract level & describe typical user system interaction. UCD mainly serve to show external view of system it shall explain external view of system from the viewpoint of user or relation to neighbouring system.
UCD serve as basis for determining test case testing as external view is model technique is useful for both system testing & acceptance testing.
The following is necessary for determining test cases for UCBT.
1) Start Situation and Preconditions
2) Other Possible Condition
3) Expected Results
4) Post Conditions
F.g. for use case diagram for use case diagram notation.

1) Actor stickman name of order
2) Use cases (Name of use cases...)
3) Association →
4) System Boundary Box →
5) Rotation
   → include « include »
   → Extend « extends »
Q.6 Attempt the following (any TWO) [10]
Q.6(a) Explain roles and responsibilities of test leader in an organization. [5]
(A) Roles of test leader in test organisation:
Test leaders tend to be involved in planning, monitoring, & control of testing activity & tasks. At the outset of project, test leaders, in collaboration with other stakeholders, device test objectives, organizational test policies, test strategies, & test plans.
They estimate testing to be done & negotiate with management to acquire necessary resources they recognize when test automation as appropriate & it is they plan the effort, select the tools & ensure training of the team.
They may consult with other groups e.g. programers, to help them with their testing. They lead guide & monition the analysis, design, implementation & exception of test cases, test procedures & test suites. They ensure proper configuration management of the test ware produced & traceability of tests to test basis.
As test execution comes near, they make sure the test environment is put into place before test execution & manager during test execution. They monitor, control & report on the test progress, the product quality states & the test result.
Daring text execution & as project winds down they write summary reports on test states. Sometime test leaders wear diff., titles, such as test management or test Co-ordinator.

Q.6(b) What are the skills required for test staff. [5]
(A) Doing testing properly requires more than defining the right position & number of people for those position. Good test teams have the right mix of skill based on tasks & activities they need to carry out & people outside team who are in charge of test task need the right skills too. People involved in testing need basic professional & social qualification such as literary, the ability to prepare & communicated effectively & so on. Going beyond that when we think of the skill that test need three main areas to me to main.
Application or business domain : A tester must understand the intended behavior, the problem the system will solve, the process it will automate.
Technology : A tester must be aware of issues, limitation & capabilities of chose implementation technology, in order to effectively and efficiently locate problem & recognize the "likely to fail" functions & features.
A tester must know testing in order to efficiently carry out the test tasks assigned.

Q.6(c) Explain test strategies and approaches used in testing. [5]
(A) The choice of test approaches or strategies is one powerful factor in success of test effort & accuracy of test plan & estimates. This factor is under control of testor & test leaders
1. Analytical
2. Model-based
3. Methodical
4. Process or standard compliant
5. Dynamic
6. Consultative or directed
7. Risk
8. Skill
9. Objectives
10. Regulations
11. Product
Q.6(d) What are the factors affecting test effort. [5]

(A) Testing is complex procedure on many projects & variety of factors can influence it.
When creating test plan & estimating test effort & schedule you must keep these factors in mind or you plans & estimate will deceive you at beginning of project & betray you at middle or end.
The test strategies or approaches you pick will have a major influent on testing effort.
Product factor start with presence of sufficient project documentation so that testers can figure out what system is, how is opposed work & what correct behaviour look like.
The importance of non–functional quality characteristic such as usability, reliability, security, performance, & so forth influences the testing effort.
These test target can be expensive & time consuming.
Complexity is another major product factor. The difficulty of comprehending & correctly handling the problem they system is being built to solve use of innovative technologies.
The need for intricate & perhaps multiple test configuration especially when these rally on the timely arrival of scarce software, hardware & other supplies.
The privalece of stringent security rules. The privalece of stringent security rule, strictly regimented process or other regulation geographical distribution of team especially if team croseses time zones.

Q.7 Attempt the following (any THREE)  [15]
Q.7(a) Explain potential benefits and risks of using tools. [5]
(A) Tool can be any device which reduces human efforts & makes the task easier
e.g. of tools are calculator, hammer, etc.

The potential benefits of tools are
1. Tool reduces the human efforts
2. Using tools user can save their time
3. Using tool, it reduces repetitive work
4. Greater consistency repetitive work
5. Objective assessment
6. Ease of access to information about test or testing

Risk of Using tools :
1. Unrealistic expectations for the tool
2. Underestimating time, cost & effort initial introduction of tool
3. Underestimating time & effort needed to achieve significant & continuing benefits from the tool
4. Underestimating effort require to maintain test assets generated by the tool
5. Over reliance on the tool

Q.7(b) Explain test management and requirement management tools. [5]

(A) Requirement management tools are used to gather requirement form end users.
e.g. of requirement management tools is are we portals.
Set of questionnaire
Features of characteristics of requirements management tools including support for:
1. Storing requirements statements.
2. Storing information about requirement attributes.
3. Checking consistency of requirement
4. Identify undefined, missing or to be defined later requirements.
5. Prioritizing requirements for testing purposes.
6. Traceability through levels of requirements
7. Interfacing to test management tools
8. Coverage of requirements by set of test.

Test management tools are use to manage entire test process. It starts from test data preparation, test data exaction towards the preparation of final result of how many test & outstanding.
The features or characteristics of test management tools are
1. Management of test
2. Scheduling of test to be executed (manually or by a test execution tools).
3. Management of testing activites (time spent in test design, test execution whether we are on schedule or.

Q.7(c) Explain features and characteristic of incident configuration management tools. [5]

(A) Configuration management tools are used whenever any changes encounter during software life cycle software configuration management is an art of identifying and implementing necessary changes in a software.
Software configuration management tools are use when any request for change is encountered by end users or developer
SCM is 4 step process:
1. Identify the change
2. Control the change
3. Ensure that the change has been properly implemented
4. Status reporting

Features or characteristics of configuration management tools includes:
1. Staring information about version & builds of the software & test ware.
2. Traceability between software and test ware & different versions & variants.
3. Keeping track of which versions belong with which configurations (e.g. o.s. libraries).
4. Reporting of statistics/metrics about incidents.
5. Access control.

Incident Management Tools:
These are used when there is incident occur during software execution.
Features or characteristics of incident management tools:
1. Storing information about attributes of incidents.
2. Storing attachments (e.g. screenshot).
3. Prioritizing incidents.
4. Assigning actions to people (Fix, confirmation test).
5. Status (e.g. open, rejected, duplicate, dereferred, closed).
6. Reporting of statistics/metric about incidents (e.g. average time open).
7. Number of incidents with each status, total number raised, open or closed.

Q.7(d) Explain reporting and characteristics of unit test tools and security tools. [5]

(A) Unit test framework tool is use to perform unit testing on software component.
Features / Characteristic includes:
Supplying inputs to the unit/module of software
Receiving o/p generated by unit of software
Which are being tested
Executing set of test cases on s/w unit
Record the result of each test case performed on each unit using pass or fail basis.
Storing the result of test cases
Support for debugging
Covering measurement at code level.
Security tools:
There are number of tools that protect system from external attack for e.g. firewalls, which are important for any system security testing tools can be used to test security by trying to break into systems whether or not it is protected by a security tool.
The attack may focus on the network the software support, the application code or underlying database.
Features or characteristics of security testing tools includes support for:
1. Identifying software's.
2. Detecting instructions such as denial of service attacks
3. Simulating various types of service attack
4. Probing for open ports or other externally visible points of attacks
5. Identifying weakness in password files & passwords
6. Security checks during operation. E.g. for checking integrity of files & intrusion detection e.g. checking result of test attacks.

Q.7(e) Explain Agile methodology in details. [5]
(A) Agile Model
1. Extreme programming is currently one of the most well known agile development life cycle model.
2. The methodology claims to be more human friendly than the traditional development method.
3. Using Agile model, Developer can develop simple and Interesting GUI for S/W.
Some of the characteristics of XP are:
- It promotes the generation of business stories to define functionality.
- It demands an on side costumer for continues feedback and to define and carry out functional test.
- It promotes pair programming and shared code ownership among developers.
- It states that component test scripts shall be written before code is written and those test should be automated.
- It states that integration and testing of code shall happen several times a day.
- It always states that we should always implement simplest that should meet solutions to the today’s problem.
Q.7(e) Explain Agile methodology in details. 

(A) Test Design Tool:
Features:
Generating test input values from
Requirements
Design Models, (state, data and Object):
Code;
Graphical User Interface
Test condition
Generating Expected Result if an oracle is available to the tool

Test Data Preparation Tool:
1. Extracted Selected data records from files or database.
2. 'Massage Data Records' to make them anonymous and or not able to identified with real people (For data protection);
3. Enable records to be shored or arranged in different order.
4. Generate new records populated with pseudorandom data, or data set up according to some guidelines e.g. an operational profile.
5. Constrict a large number of similar records from template to give large set of records for volume test.

Test Execution Tools:
1. Capturing (record) test inputs while test are execute manually.
2. Storing an expected result in the form of screen or object to compare to the next time test is run Executing test from scored script and optionally data files accessed by script.
3. Dynamic comparison (while test is running) of screen elements, links, controls, objects and values;
4. Ability to initiate post execution comparison logging result for example : Exciting the screed displayed current data and time which is not of interest to a particular test.
Synchronising input with the application under test e.g. wait unfill the application.