Long Answers
Ch2:-Requirements Analysis and Specification

Q1.List out requirements for online movie ticket booking system for multiplex.

Non-functional requirement:

1. Security –
The system uses SSL (secured socket layer) in all transactions that include any confidential customer information.
The system must automatically log out all customers after a period of inactivity.
2. Reliability –
The system provides storage of all databases on redundant computers with automatic switchover.
The reliability of the overall program depends on the reliability of the separate components.
3. Availability –
The system should be available at all times, meaning the user can access it using a web browser, only restricted by the downtime of the server on which the system runs. In case of an of a hardware failure or database corruption, a replacement page will be shown.
4. Maintainability –
A commercial database is used for maintaining the database and the application server takes care of the site.

Functional requirement:

1. Registration –
If a customer wants to book the ticket then he/she must be registered, an unregistered user can't book the ticket.
2. Login –
Customer logins to the system by entering valid user id and password for booking the ticket.
3. Search Movie –
The system shall have a search function. Customer or visitor can search movies based on movie name, date, time and venue
4. Seat Viewing –
The customer shall be shown a 2D image of the seats from which the desired seats are selected.
5.Ticket canceling –
The customer shall be given an option to cancel ticket one hour before the movie with some fine.
6. Payment –
For the customer there are many types of secure billing will be prepaid by debit or credit card. The security will provide by the third party like Pay-Pal etc.
7. Logout –
After the payment or browse the movie, the customer will log out.
8. Generate ticket –
After booking, the system can generate the portable document file (.pdf) and then sent one copy to the customer's Email-address and another one as an SMS to customer's phone.
9. Add movies –
The system shall have a feature for admin to add movies and their details.
10. Remove movies –
The system shall have a feature for admin to remove movies.

Q2.Explain requirement gathering activities.

Requirement gathering:

• It is usually the first part of any software product.

• Thisisthe base for the whole development effort.

• The goal of the requirement gathering activities to collect all related information from the customer regarding the product to be developed.

• This Is done to clearly understand the customer requirements so that incompleteness and inconsistencies are removed.

• In this phase, meeting with customers, analyzing market demand and features of the product are mainly focused.

• So, activity of market research (for competitive analysis) is done.

• It involves interviewing the end-users and studying the existing documents to collect all possible information.

Q3.What is the role of System Analyst?

• Change in the environment or technical aspects may affect the requirement analysis process.

• System analyst identifies and resolves various requirements problems.

• Forthat, the analyst has to identify and eliminate the problems of anomalies, inconsistencies and incompleteness. Anomaly is the ambiguity in the requirement,Inconsistency contradicts the requirements, and Incompleteness may overlook some requirements.

Q4.Explain contents of the SRS document.

An SRS should clearly document the following three things:

(i) Functional requirements of the system

• Functional requirements are those which are related to the technical functionality of the system.

• These are the services which the end users expect from the final product.And these are the services which a system provides to the end users.
• It clearly describes each of the functions that the system needs to perform along with the input and output data set.

(ii) Non-functional requirements of the system
• The non-functional requirements describe the characteristics of the system that can't be expressed functionally. For example, portability, maintainability, reliability, usability, security, performance etc.
• Non-functional requirements are requirements that specify criteria that can be used to judge the operation of a system in particular conditions, rather than specific behaviors.
• Sometimes These requirements are also called quality attributes.

(iii) Constraints(restrictions) on the system
• That describes what the system should do or should not do. These are some general suggestions regarding development.
• A constraint can be classified as:
o Performance constraint
o Operating system constraint
o Economic constraint
o Life cycle constraint
o Interface constraint

Q5.Compare functional and non-functional requirement of software.

Functional requirement	Non-functional requirement
Functional requirements are those which are related to the technical functionality of the system.	The non-functional requirements describe the characteristics of the system that can't be expressed functionally.
These are the services which the end users expect from the final product.	Non-functional requirements are requirements that specify criteria that can be used to judge the operation of a system in particular conditions, rather than specific behaviors.
It clearly describes each of the function that the system needs to perform along with the input and output data set.	Sometimes These requirements are also called quality attributes.

Q6.What are the benefits of SRS?

• SRS provides the foundation for design work. Because it works as an input to the design phase.
• It enhances communication between customer and developer because user requirements are expressed in natural language.
• Developers can get the idea what exactly the customer wants.
• It enables project planning and helps in verification and validation process.
• Format of forms and rough screen prints can also be represented in SRS.
• High quality SRS reduces the development cost and time efforts.

Ch3:-Software Design with UML

Q1.Write a short note on pipe and filter architecture.

Not found in Notes.

Q2.Explain Client Server Architecture with Diagram.

Not found in Notes.

Q3.What are the advantages and disadvantages of DFD?

Advantages of DFD model:

• DFD is a simple graphical technique which is very simple to understand and easy to use.
• It can be used as a part of system documentation.
• DFD can provide detailed description of the system components. • It provides clear understanding to the developers about the system boundaries and analysis of the system.
• It explains the logic behind the data flow within system.
• It provides structure analysis of the system.
• Symbols used in DFD model are very less.
• It Does not provide any time dependent behavior like we cannot consider at which time we have to do a particular process.

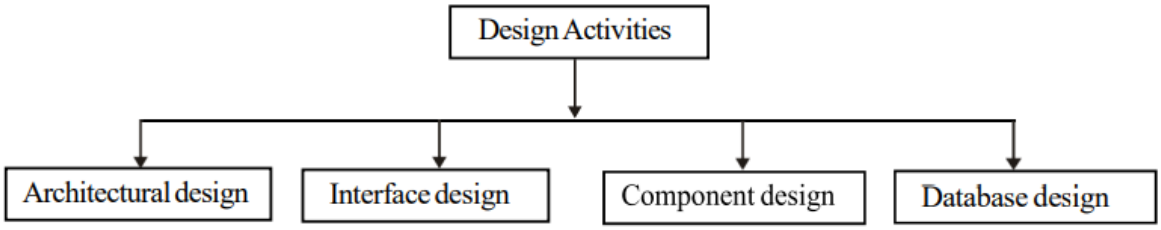
Disadvantages of DFD model:

• Control information is not defined by a DFD.
• DFD does not provide any specific guidance as how exactly decompose a given function into its sub functions; we have to use subjective judgment to carry out decomposition.
• Sometimes It puts programmers in a little confusing state.
• Different DFD models have different symbols, e.g. in Gane and Sarson notations the process is represented as a rectangle while in

Demarco and Yordan notations it is ellipse, making confusion at the time of referring. (In some of the notations you can see the process symbol is rectangle while in some other notations, the process symbol is ellipse and that is confusing).
• Physical considerations are left out in DFD.

Q4.Explain classification of design activities.

Classification of design activities:
• Design activities are depending on the type of the software being developed.
• Design activities can be classified like:



1. Architectural Design:
Design Activities
• Where you can identify the overall of the system,subsystems, modules and their relationships.
• It defines the framework of the computer based system.
• It can be derived from DFD (data flow diagram).

2. Interface design:
• Where you can define the interface between system components.
• It describes how the system communicates with itself and with the user also.
• It can be derived from DFD and State transition diagrams.

3. Component design:

• That defines each system component and shows how they operate.
• It can be derived from the State transition diagram.

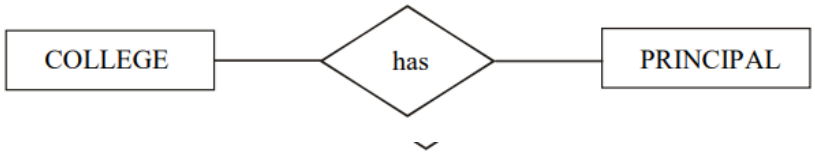
4. Database design:
• Where you can define the system data structure and show how they are represented in a database.
• Existing databases can be reused or a new database to be created.
• The data objects and their relationships are defined in ERD (entity relationship diagram) and detailed content described in data dictionary (DD).
• It can be derived from ERD and DD.

Q5.Explain cardinality and modality.

Cardinality :
• The elements of data modeling - data objects, attributes, and relationships, provide the basis for understanding the information domain of a problem.However, additional information related to these basic elements must also be understood.
• Object X has a relationship to object Y does not provide enough information for software engineering purposes. We must understand how many occurrences of object X are related to how many occurrences of objectY.This leads to a data modeling concept called cardinality.
• The concept of cardinality defines the maximum number of objects that can participate in a relationship. That means number of occurrences of one [object] that can be related to the number of occurrences of another [object].
• Cardinality is usually expressed as simply 'one' or 'many.' For example a father can have one or more children but a child can have only one father.

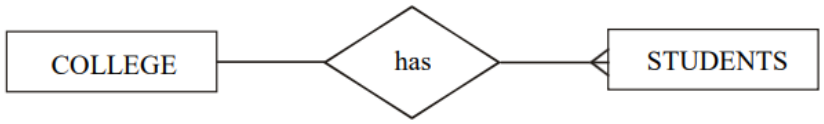
➢ **One to One (1 : 1)**

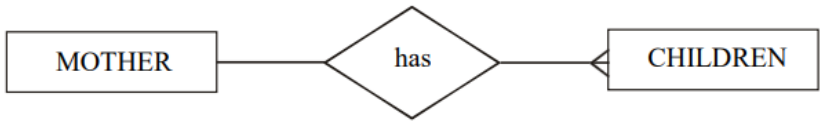
- An instance (occurrence) of entity (object) A can relate to one only instance (occurrence) of B and instance of B can relate to only one instance of A.
- For example one college has only one principal and one principal can take charge of one college only.



➢ **One to Many (1 : M)**

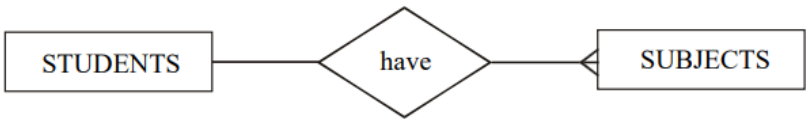
- One instance of entity A can relate to one or many instances of B, but an instance of B can relate to only one instance of entity A.
- A mother can have many children but a child can have only one mother. In another example, one college can have many students but a student can be enrolled to one college.

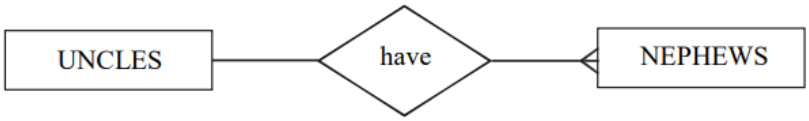




➢ **Many to Many (M : M)**

- One or more instances of entity A can relate to one more instances of entity B. and vice versa.
- For example an uncle can have many nephews while a nephew can have many uncles. In another example, many students can registered with one or more subjects as well one subject can be registered by one or more students.



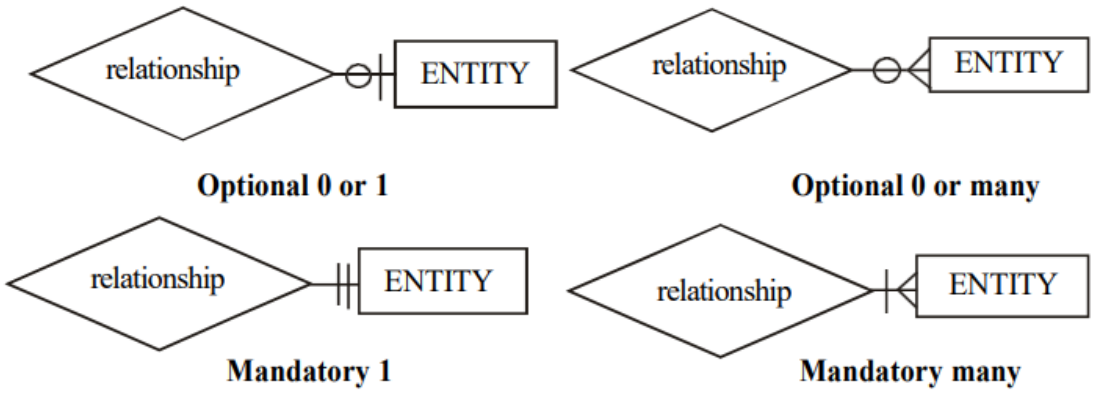


Modality:

• Cardinalitydoes not, however, provide an indication of whether or not a particular data object must participate in the relationship.To specify this information, the data model adds modality concept to the object/relationship pair.

• Modality The form of cardinality.

• Modality Means a classification of relationships on the basis of whether they claim necessity, possibility or impossibility.
• The modality of relationship is 0 if there is no explicit need for the relationship or the relationship is 0 or it is optional.
• The Modality Is 1 if an occurrence of relationship is mandatory.
• The notations for modality are explained below.



Q6.Explain use case diagram.

Writing Use-Cases (Use-Case diagram):
• Use case model in UMLprovidessystem behavior.
• The use case model for any system consists of a set of "use cases".
• Use cases represent the different ways in which a system can be used by the users.
• The purpose of a use case is to define the logical behavior of the system without knowing the internal structure of it.
• Use case identify the functional requirements of the system.
• Use case diagram describes "who can do what in a system".
• A use case typically represents a sequence of interactions between the user and the system.
• Use cases corresponding to the high level functional requirements.
• For example inATM systems, the system should have the following use cases. Check balance Withdraw Money Deposit money Transfer Money Print Mini Statement Link aadhaar card Pin change etc.

Ch4:-Software Project Management

Q1.Explain different types of risks involved in software development.

• There are three main categories of risks which can affect a software project :

1. Project risks Project risks threaten the project. They concerned various forms of budgetary, schedule, personnel, resource, and customer-related problems. An important project risk is schedule slippage. Because it is very difficult to monitor and control this risk.
2. Technical risks Risks that threaten the quality of the product. These risks concern potential design, implementation, interfacing, testing, and maintenance problems. It also covers the specification risks. Most technical risks occur due to the development team's insufficient knowledge about the project.
3. Business risks Risks that threaten the development (or client) organization. This type of risks include risks of building an excellent product that no one wants, losing budgetary or personnel commitments, etc.

Q2.Explain WBS.

• Work Breakdown Structure (WBS) is used to decompose a given task set recursively into small activities.

• The WBS is a uniform, consistent, and logical method for dividing the project into small, manageable components for purposes of planning, estimating, and monitoring.

• An effective WBS encourages a systematic planning process, reduces the omission of key project elements, and simplifies the project by dividing it into manageable units.
• A WBS will provide a roadmap for planning, monitoring, and managing all factors of the project like resource allocation, scheduling, budgeting, productivity, performance etc.
• WBS can be shown graphically in a hierarchical tree structure and developed top to bottom manner.
• The root of the tree is labeled by the problem name.
• Each node of the tree is broken down into smaller activities that are made the children of the node.
• Each activity is recursively decomposed into smaller sub-activities until at the leaf level.
• WBS can be done by the decision of the project manager.

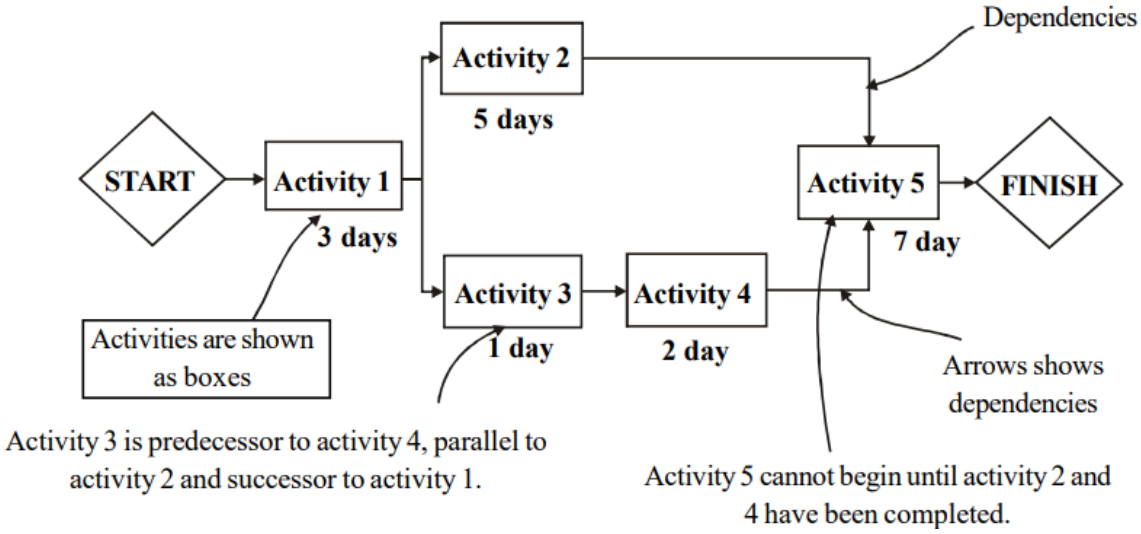
Types of WBS :
• There are three types of WBS as follows :
(i) Process WBS :
it decomposes large processes into smaller ones. Each process finally decomposed in the task.
(ii) Product WBS :
it decomposes large entities into smaller components. It is used by system engineers.
(iii) Hybrid WBS :
it includes both process and product elements into a single WBS.

Q3.Write a short note on activity network

• In project management, an activity is a task that needs to be accomplished within defined period of time or by deadline to achieve goal.
• Activities in a project are graphically represented using an activity network diagram.

• Activity network is a network graph using nodes with interconnecting edges to represent tasks and their planned sequence of completion, interdependence and interrelationship that must be accomplished to reach the project goals.
• An Activity Network Diagram helps to find out the most efficient sequence of events needed to complete any project. It enables you to create a realistic project schedule by graphically showing :

The total amount of time needed to complete the project
The sequence in which tasks must be carried out
Which tasks can be carried out at the same time
Which are the critical tasks that you need to keep an eye on.
• At the time of drawing an activity network diagram, each activity is represented by a rectangular node and the duration of the activity is shown alongside each task.



• The Activity Network diagram displays interdependencies between tasks through the use of boxes and arrows. Arrows pointing into a task box come from its predecessor tasks, which must be completed before the task can start. Arrows pointing out of a task box go to its successor tasks, which cannot start until at least this task is complete.

Q4.Explain FP.

• Function point metric was first proposed by Albrecht in 1979 and internationally approve modified model in1983. This metric overcomes many of the shortcomings of the LOC metric.
• Function point metrics, measure functionality from the users' point of view, that is, on the basis of what the user requests and receives in return. 4.2 METRICS FOR PROJECT SIZE ESTIMATION 4.2.1 Lines of code (LOC) :
• The simplest measure of problem size is Lines Of Code (LOC) or Source Lines Of Code (SLOC).
• It is very popular because of its simplicity.
• LOC is a software metric used to measure the size of a software program by counting the number of lines in
• LOC doesn't work well with non-procedural languages. 4.2.2 Function Point metric (FP) :
• One of the important advantages of using the function point metric is that it can be used to easily estimate the size of a software product directly from the problem specification.
• In the LOC metric, the project size can be accurately determined only after the product has fully been developed. While in case of FP, the size of a software product is directly dependent on the number of different functions or features it supports.
• Software supporting many features or functions should have larger FP in size.
• Each function when invoked reads some input data and transforms it to the corresponding output data.
• By using the number of input and output data values, function point metric computes the size of a software product.
• FP considers five different characteristics of the product to calculate the size. The function point (FP) of givensoftware is the weighted sum of these five items, and it will give an unadjusted function point (UFP).
UFP = (Number of inputs)*4 +

(Number of outputs)*5 +
(Number of inquiries)*4 +
(Number of files)*10 +
(Number of interfaces)*10

Q5.Explain PMC in brief.

• Planning is one of the most important project activities. Without proper planning, project monitoring and control is not possible.
• Monitoring - collecting, recording, and reporting information concerning project performance that project manager and others wish to know.
• Controlling - it uses data from monitoring activity to bring actual performance from planned performance.
• Project monitoring and planning (PMC) activities take place in parallel with project execution, so the implementation should be corrective at appropriate level.
• The main purpose is to - to ensure the quality of the final product.

Q6.Explain GANTT chart?

• It was proposed by Henry Gantt in 1914.
• It is also called a timeline chart.
• It's mainly used to allocate resources to activities (Resource Planning).
• The resources allocated to activities include staff, hardware, and software etc.
• A Gantt chart is a special type of bar chart where each bar represents an activity.
• It is one of the most popular and useful ways of showing activities displayed against time.
• The bars are drawn along a timeline. Activities against time are drawn in the bar chart. The length of each bar is proportional to the duration of time planned for the corresponding activity.
• The slack time for a particular task is also shown in the bars.

• The chart is prepared by the project manager.

Q7.Write a short note on Risk Management.

• Tomorrow's problem is today's risk.
• Software risk is a problem that could cause some loss or threaten the success of software project, but which hasn't happened yet.
• This risk may negatively affect the cost, schedule, technical success or quality of the project.
• Risk management is the process of identifying, addressing and eliminating the problems of risks before they can damage the project.

Ch:-5 Software Coding & Testing

Q1.Explain software documentation.

• Software documentation is an important aspect of both software projects and software engineering in general.
• It could be a paper document (manuals, brochure etc) or an electronic document (text written inside code). Paper documents are called external documents and electronic documents are called internal documents.
• Software documentation can be defined as, an artefact ({kLkð MkŠsík) whose purpose is to communicate information about the software system to which it belongs.
• Software documents work as an information repository for software engineers.
• It could be part of the software (internal) and it can be available offline (external).
• When various kinds of software products are developed then not only the executable files and the source code are developed but also

various kinds of documents such as users' manual, software requirements specification(SRS) documents, design documents, test documents, installation manual, etc are also developed as part of any software engineering process.
• Two main requirements for good documentation are that it is complete and up-to-date.

Q2.Explain code walkthrough.

(i) Code walkthrough
• Code walk through is an informal code analysis technique proposed by Fagan.
• This technique can be used throughout the software lifecycle to assess and improve the quality of the software products.
• A Code Walkthrough is an informal meeting where the programmer leads the review team through his/her codeand the reviewers try to identify faults.
• In the process of code walkthrough, after a module has been coded, successfully compiled and all syntax errorseliminated, a few members of the development team are given the code few days before the walk through meeting to read and understand code. Each member selects some test cases and simulates (çkLkkðxe) executionof the code by hand.
• Members note down their findings and discuss them in the meeting.
• Several guidelines are produced in the meetings and accepted as examples.
• Some of the prerequisite guidelines are the following.
The team performing code walk through should not be either too big or too small. Ideally, it should consist of between three to seven members.
Discussion should focus on discovery of errors and not on how to fix the discovered errors.

Q3.Write a short note on test documentation.

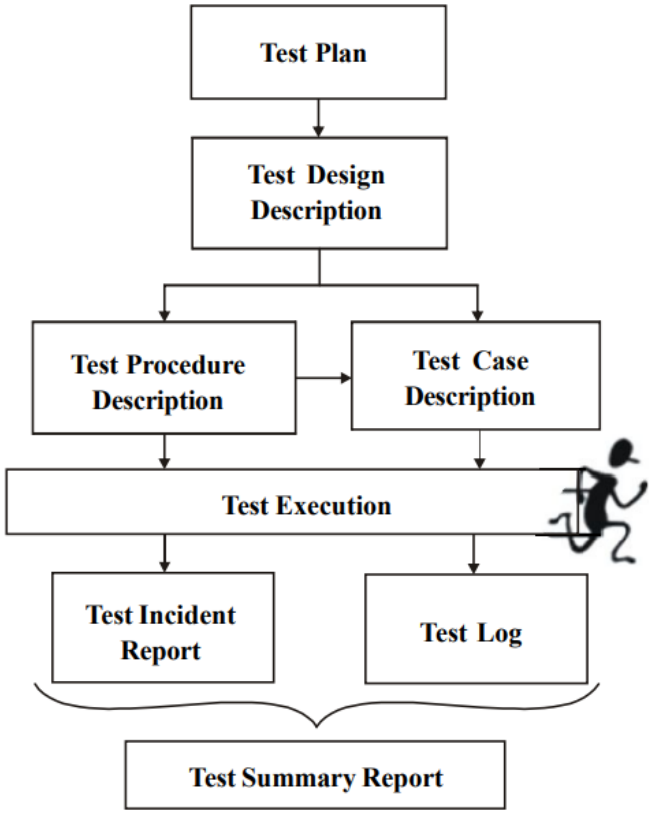
• The documentation which is generated towards the end of testing or during the testing, is the test summary reports for test documentation.

• It provides a summary of test suits which has been applied to the system.

• It specifies how many test suits are successful, how many are unsuccessful and what is the degree of successful and unsuccessful.

• Test documentation should follow the IEEE standards 829.

• Test documentation should include : scope, approach, resources and schedule of the testing activity to identify the items being tested, the features to be tested, the testing tasks to be performed, responsible personnel and the associated risks.



• A test design specification/description to identify the features to be tested and associated tests.

• A test case specification/description to define test cases identified by test design specifications.
• A test procedure specification to specify the steps for executing a set of test cases.
• A test item transmittal report to identify the test items being transmitted for testing.
• A test log' to document the generated results.
• A test incident report to document occurred events during the testing process.
• A test summary report to summarize the results of the testing activities and to provide evaluation of these results.

Q4.Compare black box & white box testing.

Black box testing	White box testing
• Synonyms of black box testing are functional testing, close box testing, data driven testing and opaque testing.	• Synonyms of white box testing are structural testing, glass box testing, clear box testing, open box testing, logic driven testing.
• No need to know internal structure of the system.	• Internal structure of the system must be known.
• It is concerned with results.	• It is concerned with details and internal workings of the system.
• Performed by end users and also by testers and developers.	• Normally done by testers and developers.
• Granularity is low.	• Granularity is low.
• It is least exhaustive and time consuming.	• Potentially most exhaustive and time consuming.
• Not suited for algorithm testing.	• It is suited for algorithm testing.
• Example : search engine.	• Example : electrical circuit testing.

Q5.Differentiate verification & validation.

Verification	Validation
→ Verification is the process of confirming that software meets its specification.	→ Validation is the process of confirming that software meets customers' requirements.
→ Verification is the process of determining whether the output of one phase of software development confirms to that of its previous phase.	→ Validation is the process of determining whether a fully developed system confirms to its requirements specification.
→ Verification is concerned with phase containment of errors	→ Validation is concerned with final product be error free.
→ Verification → "are we doing right ?"	→ And Validation → "have we done right ?"
→ Verification comes before validation.	→ Validation comes before verification.
→ It is static testing.	→ It is dynamic testing.
→ Cost of verification is less.	→ Cost of validation is more.
→ Verification does not include the execution of code.	→ Validation includes the execution of code.