Q. no. 1

a) ans:

Given

Estimated total LOC on any project = 75,000

Organizational average productivity = 244 LOC/pm

Labor rate per month = Rs.7500

Then,

Cost per LOC = (Rs.7500/m)/(244LOC/pm)

= Rs.31

Total project cost = Total LOC x Cost per LOC

= 75,000 x Rs.31

= Rs. 23,25,000

Estimated effort = Total LOC Average productivity

= (75,000 LOC) /(244 LOC/person month)

= 307 persons

1 b)ans:

Proactivity refers to anticipatory, change-oriented and self-initiated behavior in situations." In risk management and elsewhere, "proactive risk management" entails addressing problems before they start: in case fire fighting becomes necessary, an extinguisher needs to go here. But what if the extinguisher doesn't work? You have a backup: a contingency plan. Proactive risk management is what happens before a risk becomes a threat.

Reactivity, on the other hand, involves action in response to something. A fire breaks out; you grab one of the fire extinguishers called for in the proactive risk management plan.
According to the risk table developed for a project, one of the risks is 'staff turnover will be high'. List the possible steps to mitigate this risk.Reactive risk management is what happens after a risk becomes a threat.

Q. no. 2
a)ans:
Define software quality assurance(SQA).
Software quality assurance (SQA) is a process which assures that all software engineering processes, methods, activities and work items are monitored and comply against the defined standards. These defined standards could be one or a combination of any like ISO 9000, CMMI model, ISO15504, etc.

Formal Technical Review is a software quality assurance activity performed by software engineers which includes walkthroughs, inspections, roundrobin reviews. Each FTR is conducted as a meeting(Involves 3 to 5 people) and will be successful only if it is properly planned, controlled, and attended.
Its importance are:
i)to uncover errors in function, logic, or implementation for any representation of the software;
ii)to verify that the software under review meets its requirements;

iii)to ensure that the software has been represented according to predefined standards;
iv)to achieve software that is developed in a uniform manner; and
v) to make projects more manageable.

Steps to conduct FTR:
- 3-5 people involved in a review
- advanced preparation (no more than 2 hours for each person)
- the duration of the review meeting should be less than 2 hours
- focus on a specific part of a software product

2 b)ans:
Software Configuration Management(SCM)In Software Engineering, is a process to systematically manage, organize, and control the changes in the documents, codes, and other entities during the Software Development Life Cycle. The primary goal is to increase productivity with minimal mistakes. SCM is part of cross-disciplinary field of configuration management and it can accurately determine who made which revision.
The importance of configuration audit in SCM are:
-to verify compliance with configuration control standards. auditing and reporting the changes made.
- to ensure that traceability is maintained during the process.
- to ensure that changes made to a baseline comply with the configuration status reports

- to ensure the validation of completeness and consistency.
The importance of Status reporting in SCM are:
-to keep a record of all the changes made to the previous baseline to reach a new baseline
-to identify all items to define the software configuration
- to monitor status of change requests.
-to complete listing of all changes since the last baseline

Q. no. 3]
a) ans
Ans: Analysis model operates as a link between the 'system description' and the 'design model'. In the analysis model, information, functions and the behaviour of the system is defined and these are translated into the architecture, interface and component level design in the 'design modeling'.
Elements of analysis model are
1. Scenario based element This type of element represents the system user point of view. Scenario based elements are use case diagram, user stories.
2. Class based elements The object of this type of element manipulated by the system. It defines the object,attributes and relationship. The collaboration is occurring between the classes. Class based elements are the class diagram, collaboration diagram.
3. Behavioral elements Behavioral elements represent state of the system and how it is changed by the external events. The behavioral elements are sequenced diagram, state diagram.

4. Flow oriented elements An information flows through a computer-based system it gets transformed. It shows how the data objects are transformed while they flow between the various system functions. The flow elements are data flow diagram, control flow diagram.

Q.no.3}
b) ans:
A design model in Software Engineering is an object-based picture or pictures that represent the use cases for a system. Or to put it another way, it is the means to describe a system's implementation and source code in a diagrammatic fashion. Six design principles are i. Separation of Concerns
ii. Modularity
iii. Information Hiding
iv. Functional Independence
v. Refinement
vi. Refactoring
Data-flow architectures: This architecture is applied when input data are to be transformed through a series of computational or manipulative components into output data. A pipe-and-filter pattern has a set of components, called filters, connected by pipes that transmit data from one component to the next. Each filter works independently of those components upstream and downstream, is designed to expect data input of a certain form, and produces data output of a specified form.

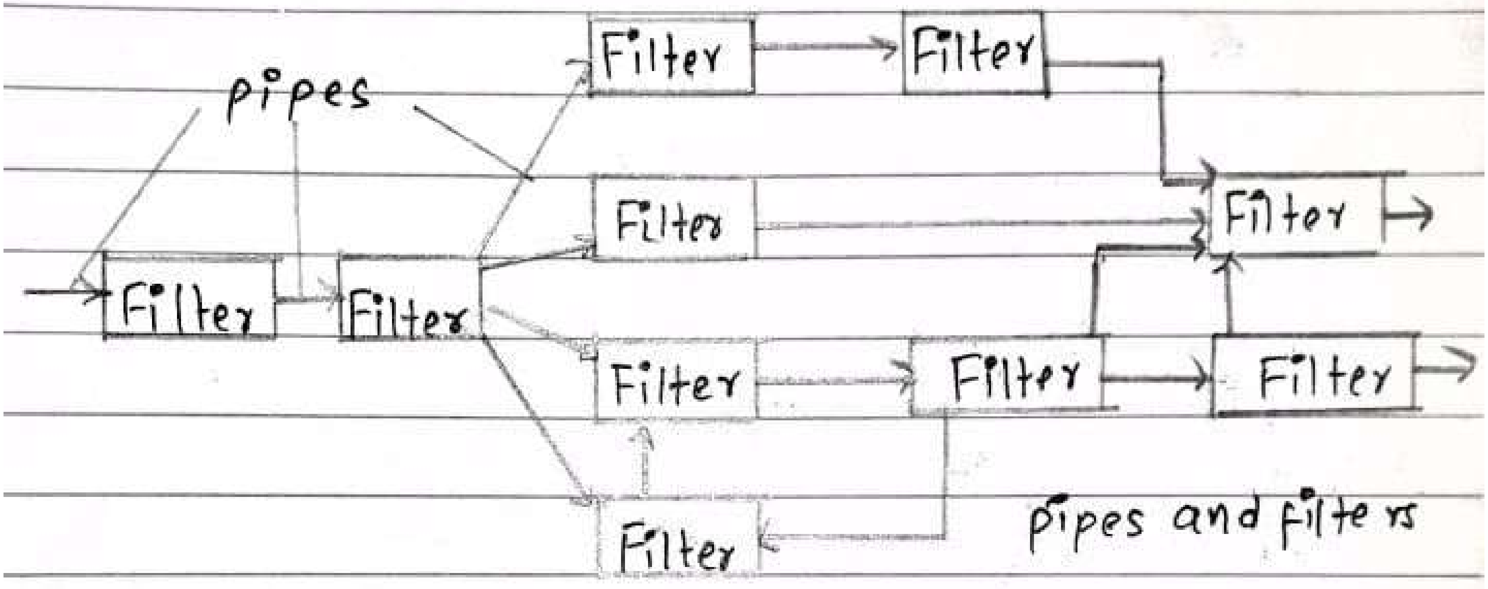

fig: Data flow diagram

4a

Ans:

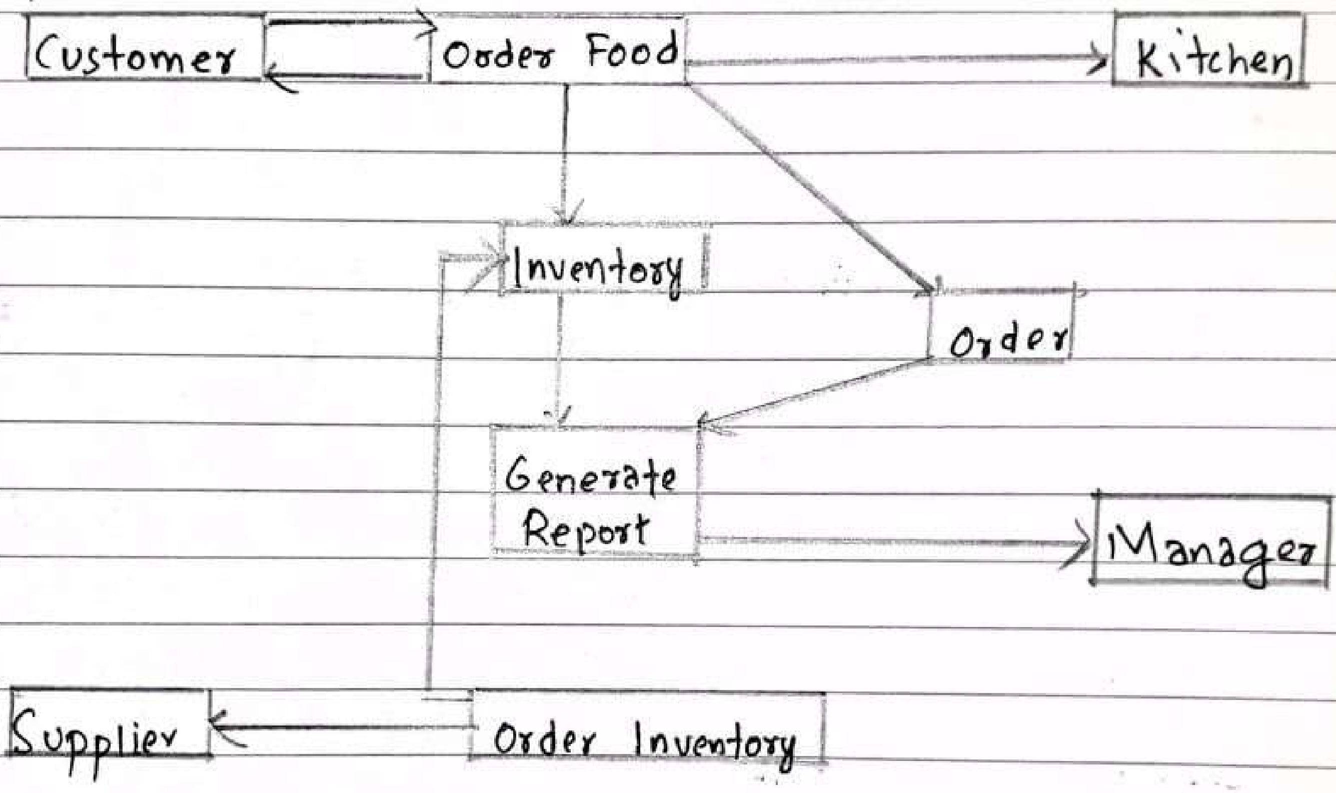

fig: Data flow diagram

Q. no. 4 b) What do you mean by software testing? List out the objective of testing. Explain software testing strategies with examples.

Ans: Software Testing is evaluation of the software against requirements gathered from users and system specifications.he major objectives of Software testing are as follows:

-Finding defects which may get created by the programmer while developing the software.

-Gaining confidence in and providing information about the level of -quality.

-To prevent defects.

-To make sure that the end result meets the business and user requirements.

-To ensure that it satisfies the BRS that is Business Requirement Specification and SRS that is System Requirement Specifications.

To gain the confidence of the customers by providing them a quality product.

Software testing strategies:

1. Test case
2. White box testing
3. Basis path testing
4. Control Strcture testing
5. Cyclomatic complexity
6. Black box testing
7. Unit testing
8. Integration testing
9. Validation testing
10. System testing

Q.no. 5) a)
Ans) Validation Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed in an appropriate environment. It answers the question, Are we building the right product?
Cyclomatic complexity: Cyclomatic complexity of a code section is the quantitative measure of the number of linearly independent paths in it. It is a software metric used to indicate the complexity of a program. It is computed using the Control Flow Graph of the program.

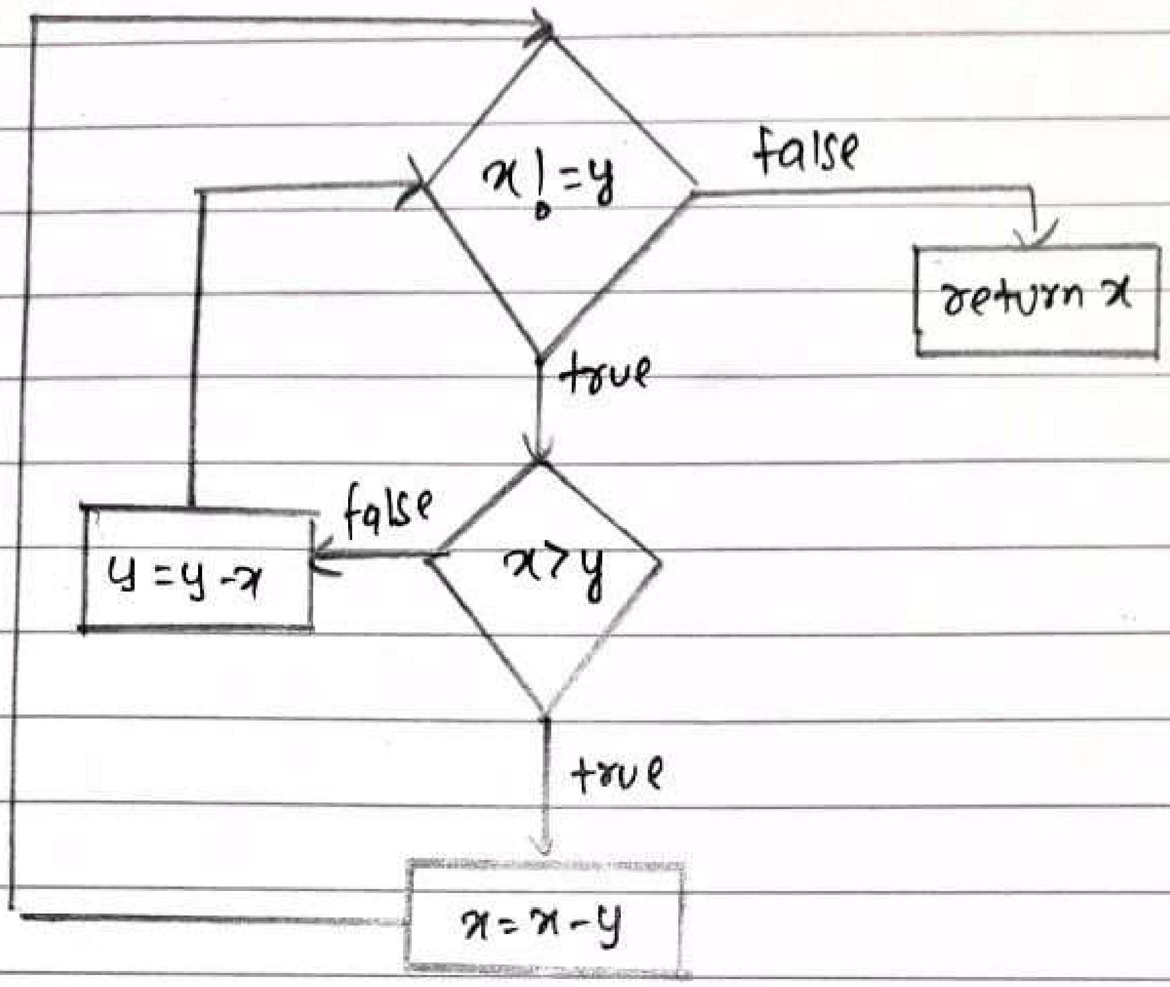

fig: flow graph

Cyclomatic complexity $V(G) = E - N + 2 \cdot (P)$

$= 6 - 5 + 2 \cdot 1$

$= 3$

5 b) What do you mean by Encapsulation? What are the steps involved in identifying the elements of an object model?

ans:

Encapsulation: Encapsulation is a process of combining data members and functions in a single unit called class. This is to prevent the access to the data directly, the access to them is provided through the functions of the class.

Steps involved in identifying the elements of an object model are:

i) Specifying attributes: Attributes describe an object that has been selected for inclusion in the analysis model.

To develop a meaningful set of attributes for an object, the analyst can again study the processing narrative (or statement of scope) for the problem and select those things that reasonably "belong" to the object.

ii) Defining operations: Defines behavior of an object and change the object's attributes in some way.Different types of operations:

(1) operations that manipulate data in some way (e.g., adding, deleting, reformatting, selecting),

(2) operations that perform a computation, and

(3) operations that monitor an object for the occurrence of a controlling event.

iii) Finalizing the Object Definition

object must be created, modified, manipulated or read in other ways, and possibly deleted.

iv) Class Construction Options: Build new class from scratch without using inheritance

Use inheritance to create new class from existing class contains most of the desired attributes and operations
Restructure the class hierarchy so that the required attributes and operations can be inherited by the newly created class
Override some attributes or operations in an existing class and use inheritance to create a new class.

Q. no. 6. a)

Ans:

Object Oriented Analysis

Object Oriented Analysis (OOA) is the first technical activity performed as part of object oriented software engineering. OOA introduces new concepts to investigate a problem. It is based in a set of basic principles, which are as follows-

-The information domain is modeled.

-Behavior is represented.

-Function is described.

-Data, functional, and behavioral models are divided to uncover greater detail.

-Early models represent the essence of the problem, while later ones provide implementation details.

Object Oriented Design:

An analysis model created using object oriented analysis is transformed by object oriented design into a design model that works as a plan for software creation. OOD results in a design having several different levels of modularity i.e., The major system components are partitioned into subsystems (a system level "modular"), and data their manipulation operations are encapsulated into objects (a modular form that is the building block of an OO system.).

In addition, OOD must specify some data organization of attributes and a procedural description of each operation.

Software domain analysis is the identification, analysis, and specification of common requirements from a specific application domain, typically for reuse on multiple projects within that application domain.
The domain analysis process can be characterized by a series of activities that begin with the identification of the domain to be investigated and end with a specification of the objects and classes that characterize the domain.

Q. no. 6b

ans: Design patterns allow the designer to create the system architecture by integrating reusable components. The best designers in any field have an uncanny ability to see patterns that characterize a problem and corresponding patterns that can be combined to create a solution.

Importance of Object oriented analysis and design:

-Focuses on data rather than the procedures as in Structured Analysis.

- The principles of encapsulation and data hiding help the developer to develop systems that cannot be tampered by other parts of the system.

- The principles of encapsulation and data hiding help the developer to develop systems that cannot be tampered by other parts of the system.

- It allows effective management of software complexity by the virtue of modularity.

- It can be upgraded from small to large systems at a greater ease than in systems following structured analysis.

Q. no. 7

b Statstical Quality Assurance

Ans: Statistical quality assurance reflects a growing trend throughout industry to become more quantitative about quality. Statistical quality assurance implies the following steps:

-Information about software defects is collected and categorized

- An attempt is made to trace each defect to its underlying cause

- Using the Pareto principle (80 percent of the defects can be traced to 20 percent, and isolate the 20 percent)

-Once the vital few causes have been identified, correct the defects.

Causes of errors:

- incomplete or erroneous specification (IES)
- misinterpretation of customer communication (MCC)
- intentional deviation from specification (IDS)
- violation of programming standards (VPS)
- error in data representation (EDR)
- inconsistent module interface (IMI)
- error in design logic (EDL)
- incomplete or erroneous testing (IET)
- inaccurate or incomplete documentation (IID)
- error in programming language translation of design (PLT)
- ambiguous or inconsistent human-computer interface (HCI)
- miscellaneous (MIS)

7c) Functional and Behavioral Modeliing

Ans:

Functional Modeling

Information is transformed as it flows through a computer-based system. The system accepts input in a variety of forms; applies hardware, software, and human elements to transform it; and produces output in a variety of forms. Regardless of size and complexity, the flow model for any computer-based system can be created. Data flow diagram (DFD) and Control flow diagram (CFD) are two common flow models. DFD or CFD have various graphical notations. A rectangle is used to represent an external entity such as hardware, a person, another program or another system that produces or receives information. A circle (or bubble) represents a process or transform that is applied to data (or control) and changes it in some way. An arrow represents one or more data items (data objects). The double line represents a data store.

Behavioral modeling

The behavioral model indicates how software will respond to external events or stimuli. To create the model, you should perform the following steps:

1. Evaluate all use cases to fully understand the sequence of interaction within the system.
2. Identify events that drive the interaction sequence and understand how these events relate to specific objects.
3. Create a sequence for each use case.
4. Build a state diagram for the system.
5. Review the behavioral model to verify accuracy and consistency.