PAGE No.
DATE 1 1

Q1] What is the significance of Cyclomatic complexity? How Cyclomatic complexity is used for performance Testing.

→ • Cyclomatic Complexity is a source code complexity measurement that is being correlated to a number of coding errors.
• It is calculated by developing a control flow graph of the code that measures the number of linearly independent paths through a program module.
• Lower the program's cyclomatic complexity, lower the risk to modify and easier to understand.
It can be represented using the formula,
Cyclomatic complexity = $V(G) = E - N + 2*P$
where,
E = no. of edges in the flow graph
N = no. of nodes in the flow graph
P = no. of predicate nodes in ——

• It helps developers and testers to identify areas of the code that are more prone to errors and require additional testing.

• Cyclomatic Complexity can be used for performance testing in the following ways:

- Determine the number of test cases required to achieve sufficient coverage.
- Identify areas of the code that are more prone to errors and require additional testing
- Improve code maintainability by reducing the cyclomatic complexity.

Teacher's Signature:.....

Q2] Describe various testing techniques and testing strategies.

COCOMO – The Constructive Cost Model

Modes :	Project size	Nature
Organic	2-50 KLOC	; Small project with experienced developers
Semi-detached	50-300 KLOC	; Medium project; medium team size, Avg experience
Embedded	+300 KLOC	; large projects, real time systems, little previous exp.

Software project	a	b	c	d
Organic	2.4	1.05	2.5	0.38
Semidetached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

- Effort = E = $a * (KLOC)^b$ PM (measured in person months or Man months)
- Development/ Scheduled Time = D = $c * (E)^d$ Months
- Average Resource/Staff size = $\frac{E}{D}$ persons (or Mans)
- Productivity of Software = $\frac{KLOC}{E}$ (KLOC/PM)

Advantages :	Disadvantages
• Easy to estimate the total cost of the project	• Ignores requirements, customer skills, and hardware issues
• Easy to implement with various factors	• Limits the accuracy of the software costs
• Provide ideas about historical projects	• depends on time factors.

Q1] What
Comple
→ • Cyclom
That is
• It is
That
throu
• lowe
modi
It c
Cyclo
whe
• It h
That
• Cycl
in
–
–
–

Teacher's Signature:.....

PAGE No.
DATE / /

Function Point Analysis:

Parameters	Weight Factors		
	Low	Average	High
External Inputs (EI)	3	4	6
External outputs (Eo)	4	5	7
External Inquiries (EI)	3	4	6
Internal logic files (ILF)	7	10	15
External logic files (ELF)	5	7	10

- Total Count (TC) factor = addition of all individual multiplications of respective weight factors and amount of parameters

- $FP = TC * [0.65 + 0.01 * \Sigma(x_i)]$ (in int)

  where x_i = Addition of all given Processing factors

- Productivity = $\frac{FP}{Efforts}$ PM

- Quality = $\frac{Defects}{FP}$

- cost per function = $\frac{Rupees}{Productivity}$

- Documentation = (Pages of documentation) / FP

Teacher's Signature:.....

Complexity Adjustment factor (CAF) :

0 - No Influence
1 - Incidental } low
2 - Moderate
3 - Average — Avg
4 - Significant } High
5 - Essential

If Individual processing factors (X_i) not given then,

$\Sigma X_i = 14 \times$ (CAF given value)

eg 3 for Average / 4 for significant

Advantages :

- estimates the software application functional size
- estimates the development time of the software application
- It is technology independent approach to measure the support and maintenance req^d in the software application
- Non IT people can easily understand the functional size of application

Disadv :

- Time consuming process to implement in software application
- costly model of estimation
- Requires lots of internal or external parameters or future's data.
- Accuracy of FPA is very difficult as multiple factors are involved on it.

Teacher's Signature:.....

PAGE No.
DATE / /

#	Feature	LOC based estimation	Function-point based estimation
1.	Basis of measurement	number of lines of code	number of functionalities such as inputs, outputs, inquiries & files
2.	Granularity	fine-grained	coarse-grained
3.	Focus	emphasizes coding	Emphasizes business requirements
4.	Complexity	simple & easy	more complex & time consuming
5.	Accuracy	Can be inaccurate if the code is not well structured	Provides more accuracy if the functional requirements are well defined
6.	Types of projects	simple, well structured	projects with complex requirements or functionality

Risk Categorization & Projection

- Risk is an expectation of loss, a potential problem that may or may not occur in the future. It is generally caused due to lack of information, control or time.
- A possibility of suffering from loss in software development process is called a software risk.

Categories:

1. Project risk:
- Project risks threaten the project plan.
- If the project risk is real then it is probable that the project schedule will slip & the cost of the project will increase.

Teacher's Signature:.....

2. Technical risk:
- Technical risk threaten the quality and timeliness of the software to be produced.
- If technical risk is real then the implementation becomes impossible.

3. Business risk:
- It threatens the viability of the software to be built and often jeoparadize the project or the product.
- Subcategories
  - market risk: creating an excellent system that no one really wants
  - Strategic risk: creating a product which no longer fit into the overall business strategy for companies.
  - Sales risk: The sales force does not understand how to sell a creating product
  - Management risk: loose a support of senior management because of a change in focus
  - Budget risk: loosing a personal commitment.

4. known risks: risks that can be uncovered after careful assessment of the project program, business & technical envs.

5. Predictable risks: risks that are hypothesized from previous project exp.

6. Unpredictable risks: risks that can and do occur, but are extremely tough to identify in advance.

Risk projection:
- Risk projection is also known as risk estimation
- It attempts to rate each risk in two ways:
  - The probability that the risk is real and will occur
  - The consequences of the problems associated with the risk, should it occur.

Teacher's Signature:.....

Rank	Range	LOSS
1	Catastrophic	• Irreversible env damage • Closure to business
2.	Critical	• Reversible env damage • Violation of regulation / law
3.	Marginal	• Avoidable env damage where restoration activities can be done
4.	Negligible	• Does not violate law • Little or minimal env. damage

Three factors affect the consequences that are likely if a risk does occur:

1. Nature: It indicates the problems that are likely if it occurs.
2. Scope: combines how serious the risk is & how much of the project will be affected or how many customers are harmed.
3. Timing: It considers when and for how long the impact will be felt.

Teacher's Signature:.....

PAGE No.
DATE / /

feature	Reactive Risk strategy	Proactive Risk Strategy
Definition	Responding to risks as they arise	Identifying and addressing risks before they become problems
Timing	occurs after a risk event has occured	occurs before a risk event has occured
Approach	Managing risks after they occur	Anticipating risks and managing them before they occur
Focus	mitigating the impact of risk that have already occured	Preventing risks from occuring or minimizing their impact.
Examples	Insurance, Disaster recovery	Risk assessments, Contingency Planning

RMMM (Risk Mitigation Monitoring and Management)

- RMMM is a process that helps organizations manage & monitor risks to reduce their potential impact.
- RMMM process is an ongoing process that requires continuous monitoring and management to ensure that the organization's risk exposure is reduced to an acceptable level.
- There are three issues in strategy for handling the risk:
  - Risk mitigation
  - Risk monitoring
  - Risk management

Teacher's Signature:.....

1. Risk mitigation:

- means preventing the risk to occur (Risk avoidance)
- Steps for mitigating risk as follows:
  - Finding out the risk
  - Removing causes that are reason for risk creation
  - Controlling corresponding documents time to time
  - Conducting timely reviews to speed up the work.

2. Risk monitoring

- is an activity used for project tracking.
- Following primary objectives:
  - check if predicted risks occurs or not.
  - ensure proper application of risk aversion steps defined for risk.
  - collect data for future risk analysis.

3. Risk management

- It assumes that the mitigation activity has failed and the risk is a reality
- This task is done by the project manager
- The main objective of the risk management plan is the risk register which describes and focuses on the predicted threats to a software project.

- The RMMM plan documents all work performed as part of risk analysis and is used by the project manager as part of the overall project plan.
- Some software teams do not develop a formal RMMM document. Rather, each risk is documented individually using a Risk Information Sheet (RIS).

Teacher's Signature:.....

5. Status Reporting: Answers the following:
- what happened?
- who did it?
- when did it happen?
- what else will be affected?
• It helps eliminate problems by improving comm. bet' all people involved.

PAGE No.
DATE / /

Software Configuration Management (SCM)

- SCM is also called as 'change management'
- SCM is an umbrella activity that is applied throughout the software process.
- Because change can occur at any time, SCM activities are developed to:
  - identify change
  - Control change
  - ensure that change is being properly implemented
  - report changes to others who may have an interest.

Steps:

1. Identification of objects in software configuration:
- control & manage software configuration items
- 2 types of objects can be identified:
  - Basic object: unit of info created during design, analysis, code, or test.
  - Aggregate object: collection of basic objects & other aggregate objects.

2. Version control: combines procedures and tools to manage different versions of configuration objects that are created during the software process.

3. Change control: combines human procedures and automated tools to provide a mechanism for the control of change.

4. Configuration control: ensures that the change has been properly implemented.
- Formal technical reviews: focuses on technical correctness of the configuration object that has been modified.
- Software configuration audit: complements FTR by assessing a configuration object for characteristics that are generally not considered during review.

Teacher's Signature:.....

PAGE NO.
DATE / /

Change control

- Change control combines human procedures and automated tools to provide a mechanism for the control of change.
- A change request is submitted and evaluated to assess technical merit, potential side effects, overall impact on other configuration objects and system functions.
- The results of the evaluation are presented as a change report, which is used by a change control authority (CCA).
- An engineering change order (ECO) is generated for each approved change.
- The "check-in" and "check-out" process implements two important elements of change control - access control and synchronization control.

Need for change is recognized
↓
Change request for user
↓
Developer evaluates
↓
Change report is generated
↓
Change control authority decides

Request is queued for action, ECO generated
↓
Assign individuals to configuration objects
↓
"check-out" configuration objects
↓
Make the change
↓

Chang request is denied
↓
User is informed

Teacher's Signature:.....

PAGE No.
DATE / /

↓

Review (audit) the change.

↓

"check in" the configuration items that have been changed

↓

Establish a baseline for testing

↓

Perform quality assurance and testing activity

↓

"Promote" changes for inclusion in next release (revision)

↓

Rebuild appropriate version of software

↓

Review the change of all configuration items

↓

Include changes in new version

↓

Distribute the new version

Teacher's Signature:.....

PAGE NO.
DATE 11

Software metrics

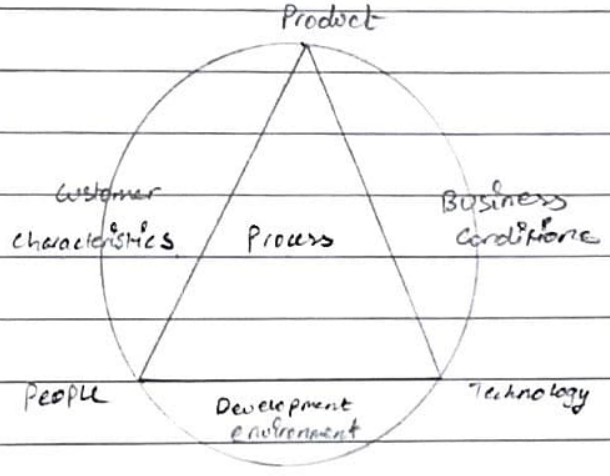
- checks the quality of the product
- There are 4 functions related to software metrics:
  - Planning
  - Organizing
  - Controlling
  - Improving
- Software metrics are classified as: Product metrics, Process metrics, and Project metrics.

Process metrics:

- It is used to measure the development and maintenance activity of the software

For example:

- Efforts require in the process
- Time to produce the product
- Effect of development tools & techniques
- No. of defects found in testing
- Productivity
- failure rate
- efficiency



- Process metrics are collected over all project & long period of time.

Teacher's Signature:.....

PAGE No.
DATE 11

Project Metrics:

- It describes the project characteristics and execution
- For example:
  - no. of software developers and other staff required over the life cycle of the project.
  - cost
  - schedule
  - Productivity

Teacher's Signature:.....

PAGE No.
DATE 11

Software testing techniques

Any engineered product can be tested in one of two ways:
(1) Knowing the specified function that a product has been designed to perform, [redacted] - Black Box testing
(2) Knowing the internal workings of the product - White Box testing

White Box Testing

- is software testing technique in which internal structure, design and coding of software are tested to verify flow of input-output and to improve design, usability, and security.
- In this, the code is visible to testers so it is also called clear box testing, Glass box testing, Open box testing.

Using white box testing methods, you can derive test cases that
1. guarantee that all independent paths within a module have been exercised at least once.
2. exercise all logical decisions on their true and false sides
3. execute all loops at their boundaries and within their operational bounds, and
4. exercise internal data structures to ensure their ~~[redacted]~~ validity.

Adv:
- very thorough
- results in optimization of codes
- easy to automate.

Disadv:
- very expensive
- very complex process
- testers are required to have in depth knowledge of code.

Teacher's Signature:.....

PAGE NO.
DATE 11

Types:

1] Basic Path testing:

- Test cases derived to exercise the basic set are guaranteed to execute every statement in the program at least once during testing.
- 4 steps followed are:
  1. construct the control flow graph: a directed graph which represents the control structure of a program.
  2. Compute the cyclomatic complexity of the Graph: It is a software metric that measures complexity of a program
  3. Identify the independent paths: a path through the program that introduces at least one new set of processing statement or a new condition.
  4. Design test cases from Independent paths.

Adv:

- Provides a systematic way of designing test cases.
- uncovers all the possible errors of the program.

2] Control Structure testing

(a) Condition testing:

- ensures all logical conditions in the program are exercised.
- complete test coverage is established.

(b) Data Flow testing:

- selects test paths of a program according to the locations of definitions and uses of variables in the program

(c) Loop testing: testing the validity of loop constructs.

4 classes of loops are:

i. Simple loops ii Nested loops iii Concatenated loops iv. unstructured loops

Teacher's Signature:.....

PAGE No.
DATE 1 1

Black Box testing

- It is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths.
- Also known as Behavioral testing.

Types:

1. Functional testing: test specific functions or features of the software under test.
2. Non-functional testing: test non-functional requirements such as performance, scalability, and usability
3. Regression testing: is done after code fixes, upgrades or any other system maintenance to check new code has not affected existing code.

Techniques / methods:

1. Boundary value analysis: Boundary value testing is focused on the various values at boundaries. This technique determines whether a certain range of values are acceptable by the system or not

2. Equivalence class partitioning: Input data is divided into partitions of valid and invalid values and it is mandatory that all partitions must exhibit the same behavior.

3. Graph based testing: Each application is built by using some objs. All the objs which are used are noted and a graph is prepared. From this graph, the relationship of every obj is identified, and test cases are written accordingly.

Teacher's Signature:.....

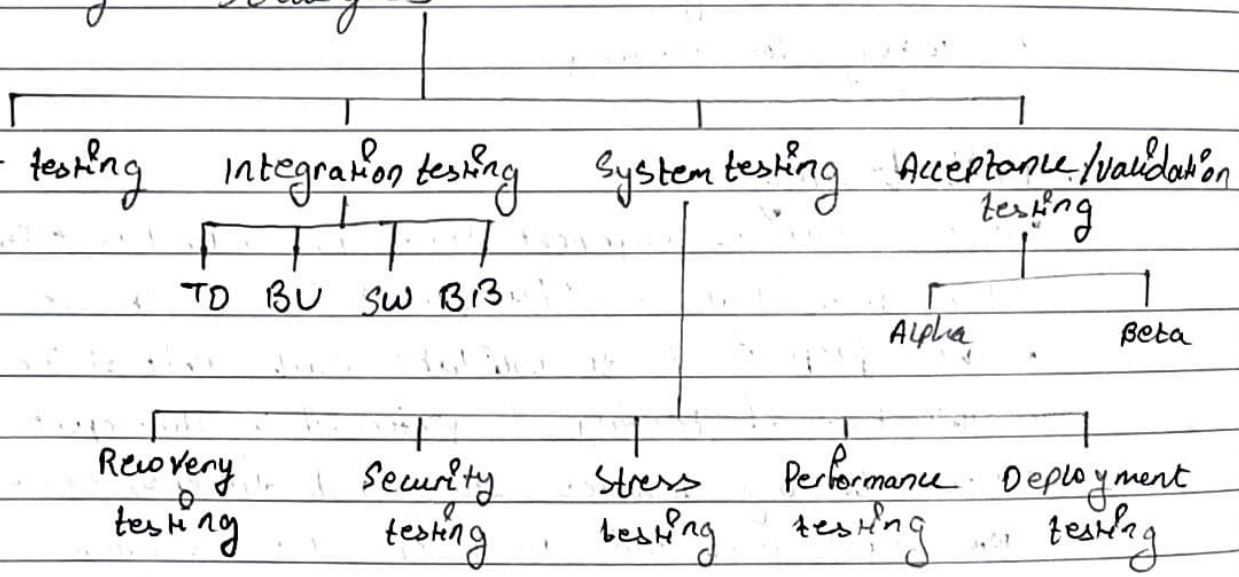
4. Decision table testing: whenever there are logical conditions or decision making steps then this technique is to be used. These can be like if a particular condition is not satisfied then perform Action A, else perform action B.

5. Error guessing testing: is based on the previous experience of a tester.

Adv:
- No technical or programming knowledge is required.
- more effective for large and complex applications.
- Tests are carried out as soon as the programming and specifications are completed.

Disadv:
- Tester may ignore possible conditions of scenarios to be tested due to lack of programming and technical knowledge.
- complete coverage in case of large & complex project is not possible

Testing Strategies



PAGE No.
DATE 11

- Verification is the process of checking that a software achieves it's goals without any bugs
- It is the process to ensure whether the product that is developed is right or not.
- Verification is static Testing.

Activities involved

1. Inspections
2. Reviews
3. Walkthroughs
4. Desk-checking.

- Validation is the process of checking whether the software product is up to mark or a product has high level requirements.
- It checks what we are developing is the right product.
- It is validation of actual and expected product
- Validation is the Dynamic Testing

Activities involved:

1. Black Box Testing
2. White Box Testing
3. Unit testing
4. Integration testing.

1. Unit testing

- Also known as component testing, is a level of software testing where individual units/components of a software are tested.
- The purpose is to validate each unit of the software.
- This type of testing is performed by developers.
- Unit testing requires stubs and drivers, stubs simulates the called unit and driver simulates the calling unit.
- Stubs and drivers are two different types of dummy modules which are mostly used in the case of component testing.

Teacher's Signature:.....

PAGE NO.
DATE / /

2. Integration testing
- Level of software testing in which units/components are integrated in a single module and then test to check communication between these components.
- It focuses on the interface & flow of the data between the modules.
- This testing comes under both black box and white box testing. This testing is done by software testers or developers.

(a) Top-down: higher level modules are tested first and then lower level modules are tested and integrated in order to check the software functionality. Stubs are used if a module is not ready.

(b) Bottom-up: lower levels of modules are tested first. once the lower level modules are tested and integrated, then the next level of modules are formed.

(c) Hybrid Integration testing: it is a strategy in which top level modules are tested with lower level modules & at the same time lower level modules are integrated with top modules and tested as a system. It makes use of both stubs as well as drivers.

(d) Big Bang Testing: all components or modules are integrated together at once and then tested as a unit. If all components in the unit are not completed, integration process will not continue.

3. Validation testing:
- The alpha test is conducted at the developer's site by a customer.
- The software is used in the natural setting with the developer "looking over the shoulder" of the user and recording errors & usage problems.
- Alpha tests are conducted in a controlled env.
- Beta test is conducted at one or more customer sites by the end-user of the software.
- unlike alpha test, the developer is generally not present.
- Beta test is a "live" application of the software in an env that cannot

Teacher's Signature:.....

controlled by the developer.
- The customer records all problems that are encountered during beta testing and reports these to the developer at regular intervals.

4. System testing
- It is a level of testing that validates the complete and fully integrated software product
- The purpose of system is to evaluate end-to-end system specifications.

Steps:
1. verification of input functions of the applications to test whether it is producing the expected output or not.
2. Testing of integrated software by including external peripherals to check the interaction of various components with each other.
3. Testing of the whole system for end to end testing.
4. Behavior testing of the application via user's experience.

- Popular types used by software testing companies:
Recovery testing; Security testing; Stress testing; Performance testing; Deployment testing; etc.

Teacher's Signature:.....