SOFTWARE TESTING METHODOLOGIES (CS615PE)
COURSE PLANNER

I. COURSE OVERVIEW:

This course will examine fundamental software testing and program analysis techniques. In particular, the important phases of testing will be reviewed, emphasizing the significance of each phase when testing different types of software. Students will learn the state of the art in testing technology for object-oriented, component-based, concurrent, distributed, graphical-user interface, and web software. In addition, closely related concepts such as mutation testing and program analysis (e.g., program-flow and data-flow analysis) will also be studied. Emerging concepts such as test-case prioritization and their impact on testing will be examined. Students will gain hands-on testing/analysis experience via a multi-phase course project. By the end of this course, students should be familiar with the state-of-the-art in software testing. Students should also be aware of the major open research problems in testing.

II. PRE-REQUISITES:

- Software engineering

III. COURSE OBJECTIVIES:

1. To provide knowledge of the concepts in software testing such as testing process, criteria, strategies, and methodologies.
2. To develop skills in software test automation and management using latest tools.

IV.COURSE LEARNING COUCOMES (CLOs):

CLO's	At the end of the course, the student will have the ability to:	Bloom's Taxonomy Levels
CLO1	List a range of different software testing techniques and strategies and be able to apply specific(automated) unit testing method to the projects.	L3: APPLY
CLO2	Distinguish characteristics of structural testing methods.	L4: ANALYZE
CLO3	Demonstrate the integration testing which aims to uncover interaction and compatibility problems as early as possible.	L3: APPLY
CLO4	Discuss about the functional and system testing methods	L2: UNDERSTAND
CLO5	Demonstrate various issues for object oriented testing	L3: APPLY

V. HOW PROGRAM OUTCOMES ARE ASSESSED:

Program Outcomes (POs)	Level	Proficiency assessed by

Program Outcomes (POs)		Level	Proficiency assessed by
PO1	**gineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.	3	Presentation on al-world problems
PO2	**Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.	3	Assignments
PO3	**Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.	2	Assignments
PO4	**Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.	2	Mini/Major Projects
PO5	**Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.	2	Mini/Major Projects
PO6	**The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.	-	--
PO7	**Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.	-	--
PO8	**Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.	-	--
PO9	**Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.	-	--
PO10	**Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.	-	-
PO11	**Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in	-	--

Program Outcomes (POs)		Level	Proficiency assessed by
	multidisciplinary environments.		
PO12	**Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.	-	-

VI.HOW PROGRAM SPECIFIC OUTCOMES ARE ASSESSED:

Program Specific Outcomes (PSOs)		Level	Proficiency assessed by
PSO1	**Foundation of mathematical concepts:** To use mathematical methodologies to crack problem using suitable mathematical analysis, data structure and suitable algorithm.	2	Assignments
PSO2	**Foundation of Computer System:** The ability to interpret the fundamental concepts and methodology of computer systems. Students can understand the functionality of hardware and software aspects of computer systems.	2	Mini/Major Projects
PSO3	**Foundations of Software development:** The ability to grasp the software development lifecycle and methodologies of software systems. Possess competent skills and knowledge of software design process. Familiarity and practical proficiency with a broad area of programming concepts and provide new ideas and innovations towards research.	-	-

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High) - : None

VII. MAPPING COURSE OUTCOMES LEADING TO THE ACHIEVEMENT OF PROGRAM OUTCOMES AND PROGRAM SPECIFIC OUTCOMES:

Course Learning Outcomes	Program Outcomes (PO)												Program Specific Outcomes (PSO)		
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**	**PSO3**
CLO1	3	2	2	-	-	-	-	-	-	-	-	-	-	-	2
CLO2	3	2	2	2	-	-	-	-	-	-	-	-	-	3	2
CLO3	2	2	3	2	-	-	-	-	-	-	-	-	1	2	-
CLO4	3	3	2	-	-	-	-	-	-	-	-	-	-	2	3
CLO5	2	2	3	2	2	2	-	-	-	-	-	-	2	3	-
AVG	**2.6**	**2.2**	**2.4**	**2**	**2**	**2**	**-**	**-**	**-**	**-**	**-**	**-**	**1.5**	**2.5**	**2.3**

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High) - : None

VIII. SYLLABUS:

UNIT - I

Introduction: Purpose of testing, Dichotomies, model for testing, consequences of bugs, taxonomy of bugs Flow graphs and Path testing: Basics concepts of path testing, predicates, path predicates and achievable paths, path sensitizing, path instrumentation, application of path testing.

UNIT - II

Transaction Flow Testing: transaction flows, transaction flow testing techniques. Dataflow testing: Basics of dataflow testing, strategies in dataflow testing, application of dataflow testing. Domain Testing: domains and paths, Nice & ugly domains, domain testing, domains and interfaces testing, domain and interface testing, domains and testability.

UNIT - III

Paths, Path products and Regular expressions: path products & path expression, reduction procedure, applications, regular expressions & flow anomaly detection. Logic Based Testing: overview, decision tables, path expressions, KV Charts, specifications.

UNIT - IV

State, State Graphs and Transition testing: state graphs, good & bad state graphs, state testing, Testability tips.

UNIT - V

Graph Matrices and Application: Motivational overview, matrix of graph, relations, power of a matrix, node reduction algorithm, building tools. (Student should be given an exposure to a tool like JMeter or Win-runner).

Text Books:

1. Software Testing techniques - Baris Beizer, Dreamtech, second edition.
2. Software Testing Tools – Dr. K. V. K. K. Prasad, Dreamtech.

References:

1. The craft of software testing - Brian Marick, Pearson Education.
2. Software Testing Techniques – SPD(Oreille)
3. Software Testing in the Real World – Edward Kit, Pearson.
4. Effective methods of Software Testing, Perry, John Wiley.
5. Art of Software Testing – Meyers, John Wiley.

NPTEL RESOURCES:

1. NOC: Software Testing, ST: (Video)
https://nptel.ac.in/courses/106/101/106101163/

GATE SYLLABUS: NOT APPLICABLE

IES SYLLABUS: NOT APPLICABLE

IX. LESSON PLAN:

Lecture No.	Unit No.	Topics to be covered	Link for PPT	Link for PDF	Link for Small Projects/ Numericals(if any)	Course learning outcomes	Teaching Methodolog	Reference
1	I	OUTCOME BASED EDUCATION AWARENESS	https://drive.google.com/file/d/1yssWHzFTl6hKlAzellH75WyJJs0tnfPW/view?usp=sharing	NA		UNDERSTAND:OUTCOME BASED EDUCATION	CHALK BOARD ,PPT PRESENTAION	
2		Introduction to Software Testing Methodologies, Purpose of testing	https://drive.google.com/file/d/10bZx8c_5oS3CTGDVSrl7IaFta5UcjhHJ/view?usp=sharing	https://drive.google.com/file/d/1vZq2zD2_IlsxwLSKFSIAJe9opn4OFmHe/view?usp=sharing	**Small Projects/ Numericals(if any) Link**	Define: Software Testing		**T1**
3		Dichotomies			**Small Projects/ Numericals(if any) Link**	Explain: Dichotomies		**T1**
4		model for testing			**Small Projects/ Numericals(if any) Link**	Describe: Model for Testing		**T1**
5		consequences of bugs			**Small Projects/ Numericals(if any) Link**	Understand: Consequences of Bugs		**T1**
6		taxonomy of bugs			**Small Projects/ Numericals(if any) Link**	Describe: Taxonomy of Bugs		**T1**
7		Basics concepts of path testing	https://drive.google.com/file/d/1jlSK46LconlSqXyrRJiVLLryPFOFMq1d/view?usp=sharing	https://drive.google.com/file/d/1BYFeMHrLYDDFtVEr1U6hwxgK2EwYNS4p/view?usp=sharing	**Small Projects/ Numericals(if any) Link**	Define: Path Testing		**T1**
8		predicates			**Small Projects/ Numericals(if any) Link**	Define: Predicates		**T1, T2**
9		path predicates and achievable paths			**Small Projects/ Numericals(if any) Link**	Describe: Path Predicates and		**T1, T2**

						Achievable Paths		
10		path sensitizing			**Small Projects/ Numericals(if any) Link**	Describe: Path Sensitizing		**T1, T2**
11		path instrumentation			**Small Projects/ Numericals(if any) Link**	Define: Path Instrumentation		**T1, T2**
12		application of path testing			**Small Projects/ Numericals(if any) Link**	Explain: Application of Path Testing		**T1, T2**
13		Bridge Class - I						**T1, T2**
14		**Mock Test-I**						
15	**II**	transaction flows	https://drive.google.com/file/d/1nfPHVmiGY6QMO83AsoLvXZ_bpmlHk37Y/view?usp=sharing	https://drive.google.com/file/d/1ujuMlzt4H7O7vCabLakWmuDhPDa0wLYC/view?usp=sharing	**Small Projects/ Numericals(if any) Link**	Define: Transaction Flows	CHALK BOARD ,PPT PRESENTAION	**T1, T2**
16		transaction flow testing techniques			**Small Projects/ Numericals(if any) Link**	Describe: Transaction Flow Testing Techniques		**T1**
17		**Basics of dataflow testing**	https://drive.google.com/file/d/1J4UcVcOT9ZeRk6V2OvFnABEgrXQyYynA/view?usp=sharing		**Small Projects/ Numericals(if any) Link**	Define: Dataflow Testing		**T1**
18		strategies in dataflow testing			**Small Projects/ Numericals(if any) Link**	Describe: Strategies in Dataflow Testing		**T1**
19		application of dataflow testing			**Small Projects/ Numericals(if any) Link**	Describe: Application of Dataflow Testing		**T1**
20		domains and paths	https://drive.google.com/file/d/1TEIVwcikDcqiNIHftVwa8hz2xsx4ogcK/view?usp=sharing	https://drive.google.com/file/d/1SAUPoW4J3bZPO2zy7BMWLcjWIuCpj0Ye/v	**Small Projects/ Numericals(if any) Link**	Define: Domain and Path		**T1**
21		Nice & ugly domains			**Small Projects/**	Define: Nice &		**T1**

				iew?usp=sharing	**Numericals(if any) Link**	Ugly Domains		
22		domain testing			**Small Projects/ Numericals(if any) Link**	Define: Domain Testing		**T1**
23		**domains and interfaces testing**			**Small Projects/ Numericals(if any) Link**			**T1**
24		domains and testability			**Small Projects/ Numericals(if any) Link**	Describe: Domains and Testability		**T1**
25	**III**	path products	https://drive.google.com/file/d/1CiBHuAaFCqmDp9SMTWVDaVGiAFq24sRY/view?usp=sharing	https://drive.google.com/file/d/19wdqL1zrUJV7gj71-6fY3msvOAOwTJfl/view?usp=sharing	**Small Projects/ Numericals(if any) Link**	Explain: Path Products	CHALK BOARD ,PPT PRESENTAION	**T1, T2**
26		path expression			**Small Projects/ Numericals(if any) Link**	Define: Path Expression		**T1, T2**
27		reduction procedure			**Small Projects/ Numericals(if any) Link**	Define: Reduction Procedure		**T1, T2**
28	**III**	applications	https://drive.google.com/file/d/1CiBHuAaFCqmDp9SMTWVDaVGiAFq24sRY/view?usp=sharing	https://drive.google.com/file/d/19wdqL1zrUJV7gj71-6fY3msvOAOwTJfl/view?usp=sharing	**Small Projects/ Numericals(if any) Link**	Describe: Applications of Domain Testing		**T1, T2**
29		regular expressions			**Small Projects/ Numericals(if any) Link**	Define: Regular Expressions		**T1, T2**
30		flow anomaly detection			**Small Projects/ Numericals(if any) Link**	Describe: Flow Anomaly Detection		**T1, T2**
31		Logic Based Testing: overview	PPT Link	https://drive.google.com/file/d/1ONPo4j_5gBZ0_ukFZ-paSNHvbgssKxj8/view?usp=sharing	**Small Projects/ Numericals(if any) Link**	Define: Logic Based Testing		**T1, T2**
32		decision tables, path expressions	PPT Link		**Small Projects/ Numericals(if any) Link**	Define: Decision Tables		**T1, T2**
33		kv charts, specifications.	PPT Link		**Small Projects/ Numericals(if any) Link**	Describe: KV Charts and		**T1, T2**

						Specifications		
34	**I V**	state graphs	https://drive.google.com/file/d/1JCoS_VRqWoxNDntBb21slRb5RALnhdwG/view?usp=sharing	https://drive.google.com/file/d/1ah57BCW_I512NRQ8ew_Q9qcg9tmH6GXU/view?usp=sharing	**Small Projects/ Numericals(if any) Link**	Define: State Graphs	CHALK BOARD ,PPT PRESENTAION	**T1**
35		good & bad state graphs			**Small Projects/ Numericals(if any) Link**	Describe: good & bad state graphs		**T1**
36		good & bad state graphs			**Small Projects/ Numericals(if any) Link**			**T1**
37		state testing			**Small Projects/ Numericals(if any) Link**	Define: State Testing		**T1**
38		state testing			**Small Projects/ Numericals(if any) Link**			**T1**
39		Testability tips			**Small Projects/ Numericals(if any) Link**	Explain: Testability tips		**T1**
40		Testability tips			**Small Projects/ Numericals(if any) Link**			**T1**
41		Bridge Class - 2						
42		**Mock Test-II**						
43	**V**	Graph Matrices and Application: Motivational overview	https://drive.google.com/file/d/1jsNXstm8z32WBap5DWPuBqVC9n7ceU9Z/view?usp=sharing	https://drive.google.com/file/d/1ZI2F9Mc3smqq17ud1Gn1Fqz14f__wdvR/view?usp=sharing	**Small Projects/ Numericals(if any) Link**	Define: Graph Matrices	CHALK BOARD ,PPT PRESENTAION	**T1**
44		matrix of graph			**Small Projects/ Numericals(if any) Link**	Describe: Matrix of Graph		**T1**
45		relations			**Small Projects/ Numericals(if any) Link**	Define: Relations		**T1**
46		power of a matrix			**Small Projects/ Numericals(i**	Describe: Power of a matrix		**T1**

					f any) Link			
47		node reduction algorithm			**Small Projects/ Numericals(if any) Link**	Explain: Node Reduction Algorithm		**T1**
48		node reduction algorithm			**Small Projects/ Numericals(if any) Link**			**T1**
49		building tools			**Small Projects/ Numericals(if any) Link**	Describe: Building Tools		**T1**
50		building tools			**Small Projects/ Numericals(if any) Link**			**T1**
51		Case Study	NA	NA	**Small Projects/ Numericals(if any) Link**	Explain: Case Studies	CHALK BOARD ,PPT PRESENTAION	
52		Case Study			**Small Projects/ Numericals(if any) Link**			
53		Case Study			**Small Projects/ Numericals(if any) Link**			
54		Case Study			**Small Projects/ Numericals(if any) Link**			
55		Revision	NA	NA	**Small Projects/ Numericals(if any) Link**	**NA**		
56		Revision			**Small Projects/ Numericals(if any) Link**	**NA**		
57		Revision			**Small Projects/ Numericals(if any) Link**	**NA**		
58		Revision			**Small Projects/ Numericals(if any) Link**	**NA**		
59		Revision			**Small Projects/**	**NA**		

					Numericals(if any) Link			

X. DESCRIPTIVE QUESTIONS

UNIT-1
Short Answer Questions

QUESTIONS	Blooms taxonomy level	Course Learning Outcomes
What is meant by testing? Why we need it.	Remember	CLO1
Define a model for software testing.	Remember	CLO2, CLO3
List the goals of software testing.	Remember	CLO3
Explain various loops with an example?	Understand	CLO3
Explain concatenated loops with an example?	Understand	CLO4
What are link counters?	Understand	CLO4
Distinguish between builder and buyer.	Understand	CLO3

Long Answer Questions

QUESTIONS	Bloom's taxonomy level	Course Learning Outcomes
Why is it impossible for a tester to find all the bugs in a system? Why might it not be necessary for a program to be completely free of defects before it is delivered to its customers?	Understand	CLO3
To what extent can testing be used to validate that the program is fit for its purpose. Discuss?	Understand	CLO3
What is meant by integration testing? Goals of Integration Testing?	Remember	CLO3
Explain white-box testing and behavioral testing?	Understand	CLO3
State and explain various dichotomies in software testing?	Understand	CLO4
Discuss about requirements, features and functionality bugs.	Analyze	CLO3
What are control and sequence bugs? How they can be caught?	Understand	CLO3
State and explain various kinds of predicate blindness with examples?	Understand	CLO3
Discuss Traversal marker with an example.	Analyze	CLO3
Explain about Co - incidental Correctness with an example.	Analyze	CLO2
What is meant by statement testing and branch testing with an example.	Remember	CLO3
State and explain various path selection rules.	Understand	CLO4
What is meant by program's control flow? How is it useful for path testing?	Remember	CLO2
Discuss various flow graph elements with their notations.	Analyze	CLO3

UNIT-2
Short Answer Questions

QUESTIONS	Blooms taxonomy level	Course outcomes
Distinguish Control Flow and Transaction flow.	Understand	CLO3
Write about any two application of data flow testing	Understand	CLO4
Define nice and ugly domains.	Remember	CLO5
Write the applications of data flow testing.	Remember	CLO5
What is meant by Domain Dimensionality.	Remember	CLO3
Define domain testing with example.	Understand	CLO3
In what a nice domain differs from and ugly domains.	Understand	CLO2

Long Answer Questions

QUESTIONS	Blooms taxonomy level	Course Learning Outcomes
What is meant by transaction flow testing. Discuss its significance.	Understand	CLO2
Discuss in detail data - flow testing strategies.	Understand	CLO2
What are data - flow anomalies? How data flow testing can explore them?	Understand	CLO3
What is meant by a program slice? Discuss about static and dynamic program slicing.	Understand	CLO3
Compare data flow and path flow testing strategies?	Analyze	CLO3
Discuss with example the equal - span range/Doman compatibility bugs.	Analyze	CLO2
What is meant by nice - domain? Give an example for nice two - dimensional domain.	Understand	CLO5
State and Explain various restrictions at domain testing processes.	Understand	CLO4
Explain how one-dimensional domains are tested?	Understand	CLO3
Discuss in detail the domains and interface testing.	Analyze	CLO2

UNIT-3
Short Answer Questions

QUESTIONS	Blooms taxonomy level	Course outcomes
Explain Regular Expressions.	Understand	CLO2
Explain sum of product form and product of sum form.	Understand	CLO3
What is logic based testing?	Remember	CLO3
Write eight steps in a reduction procedure.	Remember	CLO2
Write the limitations of path testing.	Remember	CLO4

Long Answer Questions

QUESTIONS	Blooms	Course

	taxonomy level	outcomes
Explain Regular Expressions and Flow Anomaly detection.	Understand	CLO1
Explain Huang's theorem with examples	Understand	CLO2
Discuss Path Sums and Path Product.	Understand	CLO2
Discuss in brief applications of paths	Analyze	CLO3
Whether the predicates are restricted to binary truth-values or not. Explain.	Understand	CLO4
How can we determine paths in domains in Logic based testing?	Analyze	CLO2
How the Boolean expression can be used in test case design?	Understand	CLO4
Explain prime implicant, sum of product form and product of sum form.	Understand	CLO5
Explain about the ambiguities and contradictions in specifications.	Understand	CLO5
Demonstrate by means of truth tables the validity of the following theorems of Boolean algebra: i. Associative Laws ii. Demorgan's theorems for three variables iii. Distributive Law iv. Absorption Rule	Analyze	CLO3

UNIT-4

Short Answer Questions

QUESTIONS	Blooms taxonomy level	Course outcomes
Define good state and bad state graphs.	Remember	CLO2
What is state transition?	Remember	CLO2
What is impossible state?	Remember	CLO2
What are testability tips?	Remember	CLO2
Define good state and bad state graphs.	Remember	CLO2

Long Answer Questions

QUESTIONS	Blooms taxonomy level	Course outcomes
The behavior of a finite state machine is invariant under all encodings. Justify?	Analyze	CLO2
Write testers comments about state graphs	Understand	CLO2
What are the principles of state testing. Discuss advantages and disadvantages.	Understand	CLO4
Write the design guidelines for building finite state machine into code.	Understand	CLO4
Explain with an example how to convert specification into state-graph. Also discuss how contradictions can come out.	Apply	CLO5

Write short notes on: i. Transition Bugs ii. Dead States iii. State Bugs iv. Encoding Bugs	Understand	CLO4

UNIT-5

Short Answer Questions

QUESTIONS	Blooms taxonomy level	Course outcomes
How can the graph be represented in Matrix form?	Understand	CLO1
List the different types of tools required for test planning.	Remember	CLO1
What are graph matrices?	Remember	CLO2
Categorize various testing tools necessary for testing.	Understand	CLO2
Distinguish between manual testing and automated testing.	Understand	CLO3

Long Answer Questions

QUESTIONS	Blooms taxonomy level	Course outcomes
Discuss node reduction algorithm.	Understand	CLO2
What are the matrix operations in tool building.	Remember	CLO2
Discuss the algorithm for finding set of all paths	Apply	CLO4
How can a relation matrix be represented and what are the properties of relations?	Understand	CLO3
Explain cross-term reduction and node term reduction optimization.	Understand	CLO3
Write about matrix powers and products.	Understand	CLO4
Write about equivalence relation and partial ordering relation	Understand	CLO5
What are graph matrices and their applications?	Understand	CLO5

XI. OBJECTIVE QUESTIONS

UNIT-1

1. Deviation in expected result and actual result is called_____.
2. Number of phases in testing_____.
3. Testing whether software is working as per clients requirements or not is called_____.
4. Differences in opinions is called_____.
5. _____proves programmers failure.
6. _____is programmers vindication.
7. Testing performed by the developer is called_____.
8. Bug if noticed by the client is called as_____.
9. Bug discovery is more important than_____.
10. improper layout of switch case is considered as_____bug.

11. _____starts with possibly known conditions.
12. _____starts with possible unknown conditions.
13. Functionality testing is also called as_____.
14. _____look at implementation details.
15. Automated _____is not possible yet.
16. Bug model depends on _____.
17. Testing done after combining various components is called_____.
18. _____defines how often a kind of bug can occur
19. Importance of bug is computed as_____.
20. Bug is important because_____.

UNIT-2

1. The effectiveness of path testing decreases when _____.
2. _____is sequence of program statements uninterrupted by decisions or junctions.
3. _____is a graphical representation of program‟s control structure.
4. more than one arrow leaving a circle is called _____.
5. more than one incoming arrow to a circle is called_____.
6. length of a path is measured by_____.
7. executing all statements of a program under some test is called_____.
8. If every routine has single entry and single exit then it is_____.
9. A predicate associated with a path is called_____.
10. _____blindness occurs when buggy predicate appears to work correctly.
11. _____blindness occurs when path selected by previous predicate works both for buggy and correct predicate.
12. _____blindness occurs when buggy predicate is a multiple of correct predicate.
13. The act of finding path predicate is called as_____.
14. _____is used to confirm the outcome achieved by intended path.
15. Line marker is one form of_____.
16. _____is a simulator of low level components.
17. A _____test cannot be reproduced
18. _____of test cases and results is very important.
19. Probe designed to reveal bugs may in fact hide them such bugs are called _____.
20. simplified flow graph contains only_____.

UNIT-3

1. a unit of work done completely is called_____.
2. transactions are recorded in a _____file
3. terminal controlled by PC is called _____terminal
4. creation of new transactions is called_____.
5. parent retains its transaction which is called as_____.
6. predator transaction consumes a prey this is called as_____.
7. _____are conducted at preliminary design level
8. transaction information is stored in_____.
9. _____uses finite state machine to determine next process
10. transaction control tables are used to perform_____.
11. data anomaly is represented using _____sequence
12. _____analysis is done on source code without execution

13. _____analysis is done on the fly during execution
14. a variable which cannot be used or reached is called_____.
15. _____is a path segment for which every node is visited at most once
16. _____is a path segment for which every node is visited twice
17. ku represents _____.
18. _____is the best data flow testing strategy
19. _____is part of program
20. _____is part of slice
21. set of possible values is called_____.
22. union of specified domains is incomplete, it means domain is_____.
23. domain testing is example of_____.
24. a _____can change domains boundary
25. all points with in arbitrary distance is called_____.
26. _____is a point that does not lie between any two other points
27. COOOOI stands for_____.
28. shifted boundaries can cause bug_____.
29. _____boundary occurs when coefficients in boundary are wrong
30. domain testing is easy for _____dimension
31. converting non linear boundary to linear boundary is called _____transformations.
32. linear boundary are found in_____.
33. _____analysis is done on the fly during execution
34. a variable which cannot be used or reached is called_____.
35. _____is a path segment for which every node is visited at most once
36. _____is a path segment for which every node is visited twice
37. epsilon represents _____.
38. _____is the best data flow testing strategy
39. _____is the range of all possible boundaries
40. _____transformations lead to wrong boundaries.
41. _____is allowed in kv chart
42. generally hardware logic is designed in _____
43. reducing complexity reduces_____
44. _____ does processing of inputs
45. _____are convenient way to organize statements

UNIT-4

1. without output _____cannot do anything
2. total no. of states equal to _____.
3. a state once entered cannot be left is called _____.
4. PDL stands for _____.
5. parent retains its transaction which is called_____.
6. predator transaction consumes a prey this is called_____.
7. _____are conducted at preliminary design level
8. transaction information is stored in _____.
9. _____uses finite state machine to determine next process
10. transaction control tables are used to perform_____.
11. _____analysis is done on the fly during execution
12. a variable which cannot be used or reached is called_____.
13. _____is a path segment for which every node is visited at most once

14. _____is a path segment for which every node is visited twice
15. an alternate representation of state graph is _____.
16. presence of bug can lead to _____.
17. wrong no. of states happens because of presence of_____.
18. state graph sometimes looks like a_____.

UNIT-5

1. _____is square matrix with one row and one column for each node
2. _____satisfies reflexive,transitive,antisymmetric
3. sum of out degree and in degree is called_____.
4. _____is an example of testing tool
5. parent retains its transaction which is called_____.
6. a matrix for which $A^2 = A$ is called_____.
7. _____is done by reversing pointer directions
8. transactions done should be_____.
9. transitive closure is used to define _____relation
10. transaction control tables are used to perform_____.
11. data anomaly is represented using _____sequence
12. _____analysis is done on source code without execution
13. _____analysis is done on the fly during execution
14. a variable which cannot be used or reached is called_____.
15. _____is a path segment for which every node is visited at most once
16. _____is a path segment for which every node is visited twice
17. link weighted graph is represented using _____.
18. graph matrices usually represented as_____.
19. combining every inner link with outer link happens in _____.
20. unnecessary links are removed in _____.

XII. WEBSITES:

1. Software Testing Tutorial
   https://www.tutorialspoint.com/software_testing/index.htm
2. Software Testing Methodologies

https://smartbear.com/learn/automated-testing/software-testing-methodologies/

XIII. EXPERT DETAILS:

1. Dr. Debasis Samanta, Department of Computer Science and Engineering, Takshashila Building, Indian Institute of Technology Kharagpur, Kharagpur, India — 721302.
2. Prof.Rajib Mall, Indian Institute of Technology Kharagpur, Kharagpur, India — 721302.

XIV. JOURNALS:

- Journal of Software: Testing, Verification and Reliability
- IEEE Transactions on Software Engineering
- ACM Transactions on Software Engineering Methodology

XV. LIST OF TOPICS FOR STUDENT SEMINARS:

- Flow Graphs
- Path Testing
- Data Flow Testing
- Domain Testing

- Flow Anomaly Detection
- KV Charts
- State Graphs
- State Testing
- Node Reduction Algorithm