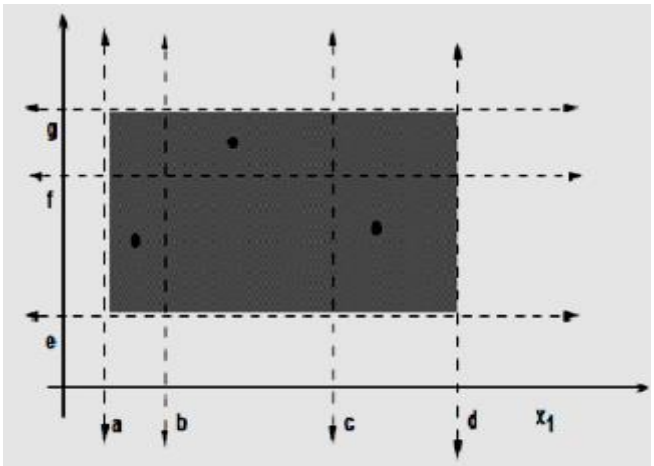
Internal Assessment Test I SCHEME– April 2023

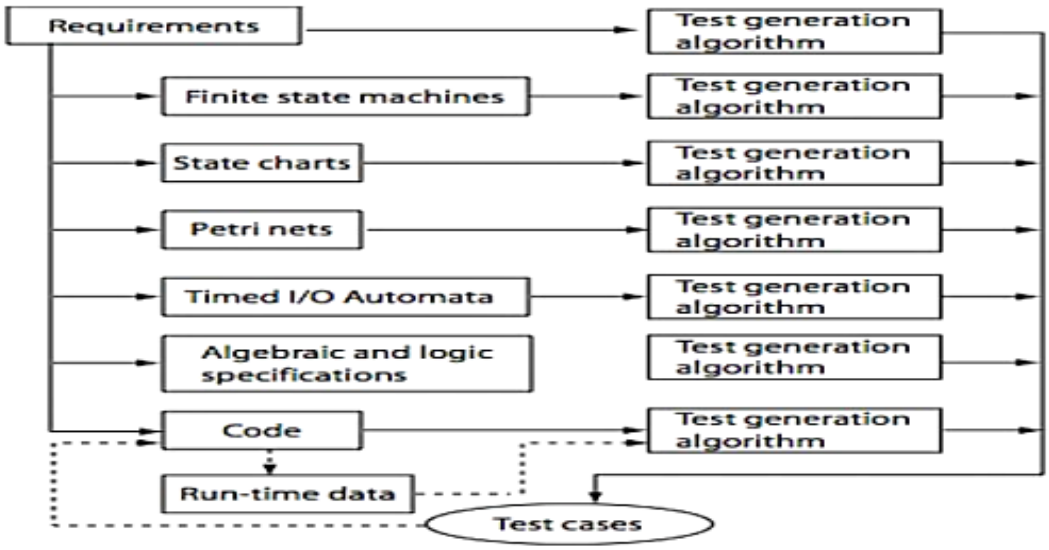
Sub:	SOFTWARE TESTING				Sub Code:	18IS62	Branch:	ISE
Date:	24/4/23	Duration:	90 mins	Max Marks:	50	Sem/Sec:	VI A,B&C	OBE

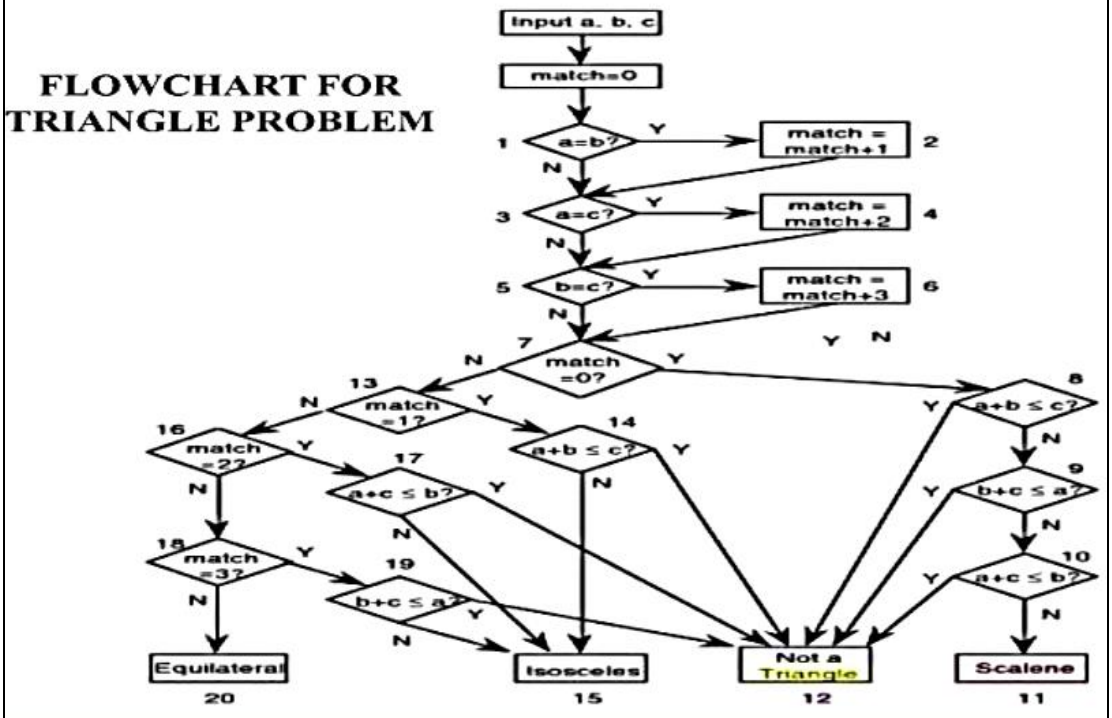
Answer any FIVE FULL Questions

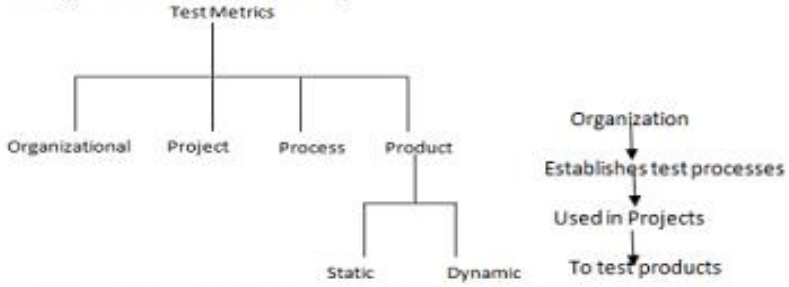
		MARKS	CO	RBT
1a)	Generate Boundary Value Analysis test cases for commission problem.	[06]	CO2	L3

Case	Locks	Stocks	Barrels	Sales	Comm	Comment
1	1	1	1	100	10	Output minimum
2	1	1	2	125	12.5	Output minimum +
3	1	2	1	130	13	Output minimum +
4	2	1	1	145	14.5	Output minimum +
5	5	5	5	500	50	Midpoint
6	10	10	9	975	97.5	Border point –
7	10	9	10	970	97	Border point –
8	9	10	10	955	95.5	Border point –
9	10	10	10	1000	100	Border point
10	10	10	11	1025	103.75	Border point +
11	10	11	10	1030	104.5	Border point +
12	11	10	10	1045	106.75	Border point +
13	14	14	14	1400	160	Midpoint
14	18	18	17	1775	216.25	Border point –
15	18	17	18	1770	215.5	Border point –
16	17	18	18	1755	213.25	Border point –
17	18	18	18	1800	220	Border point
18	18	18	19	1825	225	Border point +
19	18	19	18	1830	226	Border point +
20	19	18	18	1845	229	Border point +
21	48	48	48	4800	820	Midpoint
22	70	80	89	7775	1415	Output maximum –

b)	Write a note on Weak normal Equivalence class testing. Weak Normal Equivalence Class Testing.(2M) Accomplished by using one variable from each equivalence class in a test cases. These three test cases use one value from each equivalence class. (2M)  	[04]	CO2	L1

2	Explain: a. Software quality attributes b. Test generation Strategies (5+5M) a. Software quality attributes Static quality attributes:structured, maintainable, testable code as well as the availability of correct and complete documentation. Dynamic quality attributes:software reliability, correctness, completeness, consistency, usability, and performance. Completeness refers to the availability of all features listed in the requirements, or in the user manual. An incomplete software is one that does not fully implement all features required. Consistency refers to adherence to a common set of conventions and assumptions. For example, all buttons in the user interface might follow a common color coding convention. Usability refers to the ease with which an application can be used. This is an area in itself and there exist techniques for usability testing. Psychology plays an important role in the design of techniques for usability testing. Performance refers to the time the application takes to perform a requested task. It is considered as a non-functional requirement. It is specified in terms such as ``This task must be performed at the rate of X units of activity in one second on a machine running at speed Y, having Z gigabytes of memory." **b. Test generation Strategies (5M)** Model based: require that a subset of the requirements be modeled using a formal notation (usually graphical). Models: Finite State Machines, Timed automata, Petri net, etc. Specification based: require that a subset of the requirements be modeled using a formal mathematical notation. Examples: B, Z, and Larch. Code based: generate tests directly from the code.  	10 M	CO1	L2

3	Explain the Traditional Implementation and its flowchart for triangle problem statement using Fortran style. (5 + 5) Marks  Program triangle1 'Fortran-like version Dim a, b, c, match As INTEGER Output("Enter 3 integers which are sides of a triangle") Input(a,b,c) Output("Side A is ",a) Output("Side B is ",b) Output("Side C is ",c) match = 0 If a = b '(1) Then match = match + 1 '(2) EndIf If a = c '(3) Then match = match + 2 '(4) EndIf If b = c '(5) Then match = match + 3 '(6) EndIf If match = 0 '(7) Then If (a+b)<=c '(8) Then Output("NotATriangle") '(12.1) Else If (b+c)<=a '(9) Then Output("NotATriangle") '(12.2) Else If (a+c)<=b '(10) Then Output("NotATriangle") '(12.3) Else Output ("Scalene") '(11) EndIf EndIf EndIf Else If match=1 '(13) Then If (a+c)<=b '(14) Then Output("NotATriangle") '(12.4) Else Output ("Isosceles") '(15.1) EndIf Else If match=2 '(16) Then If (a+c)<=b Then Output("NotATriangle") '(12.5) Else Output ("Isosceles") '(15.2) EndIf Else If match=3 '(18) Then If (b+c)<=a '(19) Then Output("NotATriangle") '(12.6) Else Output ("Isosceles") ' '(15.3) EndIf Else Output ("Equilateral") '(20) EndIf EndIf EndIf	10	CO1	L2

4.	Explain different test metrics in testing the software. Explanation of each metric carries 2+2+2+2+2 = 10 Marks **Test Metrics:** • Quantitative measurement determining the extent to which a software process, product or project possesses a certain attribute (used for tracking purposes) • Goal for the metric is to quantify the progress of the product toward a specified quality objective • standard measurement. • Variety of metrics in Software Testing  • Each set of metrics has its value in Monitoring, planning and control • There are 4 metrics which are core areas • Schedule -measures actual completion times of various activities and compare these with estimated time to completion	10	CO1	L2
5	Define the terms: i) Error ii) fault iii) failure iv) incident v) test vi) test case Error: People make errors. A good synonym for error is —mistake. When people make mistakes while coding, we call these mistakes —bugs. Errors tend to propagate; a requirements error may be magnified during design, and amplified still more during coding e.g developer's coding error like Syntactic Error, User interface error, Flow control error, Error handling error, Calculation error, Hardware error, Testing Error Fault: A fault is the result of an error. It is more precise to say that a fault is the representation of an error, where representation is the mode of expression, such as narrative text, dataflow diagrams, hierarchy charts, source code, and so on. Failure: A failure occurs when a faulty code executes. Two subtleties arise here: one is that failures only occur in an executable representation, which is usually taken to be source code, or more precisely, loaded object code. The second subtlety is that this definition relates failures only to faults of commission Test: Testing is obviously concerned with errors, faults, failures, and incidents. A test is the act of exercising software with test cases. There are two distinct goals of a test: either to find failures, or to demonstrate correct execution. Test Case: The essence of software testing is to determine a set of test cases for the item to be tested. A test case is (or should be) a recognized work product. A test case has an identity, and is associated with a program behavior. A test case also has a set of inputs, a list of expected outputs	6	CO1	L1
(b)	Differentiate between black box testing and white box testing. black box testing Program-a function that maps values from its input domain to values in its output range . Content/implementation is not known. Function is understood completely in terms of its inputs & outputs . For test case identification only specification of the software is used.	[04]	CO1	L1

	White box testing. Implementation is known and used to identify test cases. Concept of linear graph theory is required to understand. Test coverage metrics –provides way to state the extent to which the software item can be tested.			
6	Generate the test cases using BVA testing and Equivalence classes for triangle problem. BVA: Table 5.1 Normal Boundary Value Test Cases (table below) **Equivalence test cases for the triangle problem** • In the problem statement, we **note that four possible outputs can** occur: NotATriangle, Scalene, Isosceles, and Equilateral. We can use these to identify **output (range) equivalence classes** as follows. R1 = {<a, b, c>: the triangle with sides a, b, and c is equilateral} R2 = {<a, b, c>: the triangle with sides a, b, and c is isosceles} R3 = {<a, b, c>: the triangle with sides a, b, and c is scalene} R4 = {<a, b, c>: sides a, b, and c do not form a triangle} Four weak normal equivalence class test cases, chosen **arbitrarily from** each class are as follows: (table below)	(5+5M)	CO2	L3

Case	a	b	c	*Expected Output*
1	100	100	1	Isosceles
2	100	100	2	Isosceles
3	100	100	100	Equilateral
4	100	100	199	Isosceles
5	100	100	200	Not a triangle
6	100	1	100	Isosceles
7	100	2	100	Isosceles
8	100	100	100	Equilateral
9	100	199	100	Isosceles
10	100	200	100	Not a triangle
11	1	100	100	Isosceles
12	2	100	100	Isosceles
13	100	100	100	Equilateral
14	199	100	100	Isosceles
15	200	100	100	Not a triangle

Test Case	a	b	c	Expected Output
WN1	5	5	5	Equilateral
WN2	2	2	3	Isosceles
WN3	3	4	5	Scalene
WN4	4	1	2	Not a Triangle

Test Case	a	b	c	Expected Output
WR1	-1	5	5	a not in range
WR2	5	-1	5	b not in range
WR3	5	5	-1	c not in range
WR4	201	5	5	a not in range
WR5	5	201	5	b not in range
WR6	5	5	201	c not in range

Test Case	a	b	c	*Expected Output*
SR1	-1	5	5	Value of a is not in the range of permitted values
SR2	5	-1	5	Value of b is not in the range of permitted values
SR3	5	5	-1	Value of c is not in the range of permitted values
SR4	-1	-1	5	Values of a, b are not in the range of permitted values
SR5	5	-1	-1	Values of b, c are not in the range of permitted values
SR6	-1	5	-1	Values of a, c are not in the range of permitted values
SR7	-1	-1	-1	Values of a, b, c are not in the range of permitted values

Faculty Signature CCI Signature HoD Signature