CST458	SOFTWARE TESTING	CATEGORY	L	T	P	CREDIT	YEAR OF INTRODUCTION
		PEC	2	1	0	3	2019

Preamble: This is a course in theoretical computer science that introduces the concepts and methods in software testing. It covers various techniques for test case design used to test software artifacts, including requirements, design, and code, the different techniques for test case design based on graphs, programming language syntaxes and symbolic execution using PEX tool. It enables the learners to follow a systematic software testing approaches while developing applications.

Prerequisite: Nil

Course Outcomes: After the completion of the course the student will be able to:-

CO1	List a range of different software testing techniques and be able to apply specific unit testing method to the projects using Junit**.(Cognitive Knowledge Level: Understand)**
CO2	Illustrate using appropriate tools the mutation testing method for a given piece of code to identify hidden defects that can't be detected using other testing methods.**(Cognitive Knowledge Level: Apply)**
CO3	Explain graph coverage criteria in terms of control flow graph and data flow graph for a given program.**(Cognitive Knowledge Level: Understand**)
CO4	Demonstrate the importance of black-box approaches in terms of domain and functional testing.**(Cognitive Knowledge Level: Apply)**
CO5	Illustrate the use of PEX tool with symbolic execution.**(Cognitive Knowledge Level: Apply)**

Mapping of course outcomes with program outcomes

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO 9	PO10	PO11	PO12
CO1	✓	✓	✓									✓
CO2	✓	✓	✓	✓	✓					✓		✓
CO3	✓	✓	✓							✓		✓
CO4	✓	✓	✓	✓								✓

CO5	✓	✓	✓	✓	✓					✓		✓

Abstract POs defined by National Board of Accreditation			
PO#	**Broad PO**	**PO#**	**Broad PO**
PO1	Engineering Knowledge	PO7	Environment and Sustainability
PO2	Problem Analysis	PO8	Ethics
PO3	Design/Development of solutions	PO9	Individual and team work
PO4	Conduct investigations of complex problems	PO10	Communication
PO5	Modern tool usage	PO11	Project Management and Finance
PO6	The Engineer and Society	PO12	Life long learning

Assessment Pattern

Bloom's Category	**Continuous Assessment Tests**		**End Semester Examination Marks**
	Test 1 (Marks)	**Test 2 (Marks)**	
Remember	30	30	30
Understand	40	40	40
Apply	30	30	30
Analyze			
Evaluate			
Create			

Mark Distribution

Total Marks	**CIE Marks**	**ESE Marks**	**ESE Duration**
150	**50**	**100**	**3 hours**

Continuous Internal Evaluation Pattern:

Attendance	**10 marks**
Continuous Assessment Tests(Average of SeriesTests1& 2)	**25 marks**
Continuous Assessment Assignment	**15 marks**

Internal Examination Pattern:

Each of the two internal examinations has to be conducted out of 50 marks. First series test shall be preferably conducted after completing the first half of the syllabus and the second series test shall be preferably conducted after completing remaining part of the syllabus. There will be two parts: Part A and Part B. Part A contains 5 questions (preferably, 2 questions each from the completed modules and 1 question from the partly completed module), having 3 marks for each question adding up to 15 marks for part A. Students should answer all questions from Part A. Part B contains 7 questions (preferably, 3 questions each from the completed modules and 1 question from the partly completed module), each with 7 marks. Out of the 7 questions, a student should answer any 5.

End Semester Examination Pattern:

There will be two parts; Part A and Part B. Part A contains 10 questions with 2 questions from each module, having 3 marks for each question. Students should answer all questions. Part B contains 2 questions from each module of which a student should answer any one. Each question can have maximum 2 sub-divisions and carries 14 marks.

Syllabus

Module - 1 (Introduction to Software Testing)

Some Popular Errors – Ariane 5, Therac 25, Intel Pentium Bug. What is Software testing? Why should it be tested? Software Quality, Role of Testing. Testing Process - Level 0 thinking, Level 1 thinking, Level 2 thinking, Level 3 thinking, Level 4 thinking. Software Testing Terminologies - Verification, Validation and Testing, Faults, Error and Bug, Test cases, Coverage Criteria. Types of Testing- Unit testing, integration testing, System testing, Acceptance testing, Beta testing, Functional testing, Stress testing, Performance testing, Usability testing and Regression testing. Testing Methods - Black Box testing, White Box testing, Grey Box testing.

Module - 2 (Unit Testing)

Concept of Unit testing. Static Unit testing. Dynamic Unit testing - Control Flow testing, Data Flow testing, Domain testing, Functional Program testing. Mutation testing - Mutation and Mutants, Mutation operators, Mutation score. Junit - Framework for Unit testing. Case Study - Mutation testing using Junit and Muclipse.

Module - 3 (Unit Testing - White Box Approaches)

Overview of Graph Coverage Criteria. Structural Graph Coverage Criteria - Node/vertex coverage, Edge coverage, Edge pair coverage, Path coverage, Complete path coverage, Prime path coverage, Complete round trip coverage, Simple round trip coverage. Data Flow Criteria - du paths, du pairs. Subsumption Relationships among Graph Coverage Criteria. Graph Coverage for Source Code - Control flow graphs for code, CFG: If statement, CFG: If statement with return, CFG: Switch-case, CFG: Loops, CFG: Exceptions (try-catch). Example program – Statistics. Graph Coverage for Design Elements - Call graphs and classes, Class inheritance testing: Coverage criteria, Coverage criteria on inheritance graph, Data flow at the design level, Inter-procedural DU pairs, Coupling du-pairs example. Example - Quadratic Root. Case Study - Graph Based testing using JUnit Framework.

Module - 4 (Unit Testing - Black Box Approaches)

Domain Testing / Input Space Partitioning - Partitions of a set. Input domain modelling - Interface-based approach, Functionality-based approach. Identifying values. Multiple partitions of the input domain - All Combinations Coverage (ACoC), Each Choice Coverage (ECC), Pair-wise Coverage, T-wise Coverage, Base Choice Coverage, Multiple Base Choices Coverage. TriTyp example. Functional Testing - Functional Testing Concepts of Howden. Functional testing - Important Steps. Types of Functional testing - Equivalence Class Partitioning, Boundary Value Analysis, Decision Tables, Random Testing. Case Study - Black Box testing approaches using JUnit.

Module - 5 (Grey Box Testing Approaches)

Introduction to Grey Box testing - Why Grey Box testing, Gray Box Methodology, Advantages and Disadvantages. Techniques of Grey Box Testing - Matrix Testing, Regression Testing, Orthogonal Array Testing or OAT, Pattern Testing. An Introduction to PEX - Parameterized Unit Testing, The Testing Problem. Symbolic Execution – Example, Symbolic execution tree. PEX application Case Study – PEX.

Text Books

1. Paul Ammann and JeffOffutt , Introduction to Software Testing, Cambridge University Press
2. Kshirasagar Naik and Priyadarshi Tripathy, Software Testing And Quality Assurance: Theory And Practice, Wiley.

Reference Materials

1. King, James C, “Symbolic Execution and Program Testing”, Association for Computing Machinery, July 1976.

Sample Course Level Assessment Questions

Course Outcome 1 (CO1):

Explain the following types of testing methods with examples.

(i) Balck-box testing.

(ii) White-box testing.

(iii) Grey-box testing.

Course Outcome 2 (CO2):

Define 12 mutants for the following method *power*() using effective mutation operators. Try to use each mutation operator at least once. Approximately, how many mutants do you think there would be, if all mutants for *power*() were created?

```
public static int power (int left, int right)
{
//*************************************
// Raises Left to the power of Right
// precondition : Right >= 0
// postcondition: Returns Left**Right
//*************************************
```

```
intrslt;
rslt = Left;
if (Right == 0)
{
rslt = 1;
}
else
{
for (int i = 2; i <= Right; i++)
rslt = rslt * Left;
}
return (rslt);
}
```

Course Outcome 3 (CO3):

Draw the control flow graph and data flow graph of given piece of code.

```
public static double ReturnAverage(int value[],int AS, int MIN, int MAX){
/*
Function: ReturnAverageComputes the averageof all those numbers in the input array in the positive range [MIN, MAX]. The maximum size of the array is AS. But, the array size could be smaller than AS in which case the end of input is represented by -999.
*/
int i, ti, tv, sum;
doubleav;
i = 0; ti = 0; tv = 0; sum = 0;
while (ti< AS && value[i] != -999) {
ti++;
if (value[i] >= MIN && value[i] <= MAX) {
tv++;
sum = sum + value[i];
}
i++;
}
```

```
if (tv> 0)
av = (double)sum/tv;
else
av = (double) -999;
return (av);
}
```

Course Outcome 4 (CO4):

Explain the following with examples.

1. Input domain modelling.
2. All Combinations Coverage (ACoC)
3. Each Choice Coverage (ECC)
4. Pair-wise Coverage
5. T-wise Coverage
6. Base Choice Coverage
7. Multiple Base Choices Coverage.

Course Outcome 5 (CO5):

Draw the symbolic execution tree for the following program code and explain the symbolic execution of testme (α1, α2).

```
int twice (int v) {
 return 2 ∗ v;
 }
 void testme (int x, int y ) {
 z = twice ( y);
 if ( z == x ){
 if ( x > y + 10)
ERROR;
 }
 }
int main() {
 x = sym input();
 y = sym input();
testme ( x , y);
```

return(0);

Model Question Paper

QP CODE: **PAGES: 3**

Reg No:_____ **Name :_____**

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

EIGHTH SEMESTER B.TECH DEGREE EXAMINATION, MONTH & YEAR

Course Code: CST458

Course Name: Software Testing

Max.Marks:100 **Duration: 3 Hours**

PART A

Answer all Questions. Each question carries 3 Marks

1. Explain the differences between Validation and Verification?
2. Explain the differences between Fault, Error, and Bug?
3. Define Ground string, Mutation score, and Mutants?
4. What are the functions of Test driver and Test stubs in dynamic unit testing?
5. Define Node coverage, Edge coverage and Prime path coverage in a control flow graph?
6. What are du paths and du pairs in a data flow graph?
7. Explain the two approaches in input domain modelling?
8. Explain the difference between Equivalence Class Partitioning and Boundary Value Analysis?
9. Briefly explain three techniques of Grey box testing?
10. Explain the concept of symbolic execution with the help of a toy example?

(10x3=30)

Part B

(Answer any one question from each module. Each question carries 14 Marks)

11. (a) Explain the following types of testing

(i) Black Box testing (ii) White Box testing (iii) GreyBox testing (iv) Unit testing (v) Integration testing (vi) System testing (vii) Acceptance testing **(14)**

OR

12. (a) Explain the following coverage criterias based on the code fragment given below? (i) Functional coverage (ii) Statement coverage (iii)Conditional coverage (iv)Branch coverage **(8)**

```
int foo (int x, int y){
    int z = 0;
    if ((x > 0) && (y > 0)){
    z = x;}
    return z;
    }
```

(b) Write positive and negative test cases for an ATM Machine? **(6)**

13. (a) Explain Dynamic unit test environment with a neat figure. **(8)**

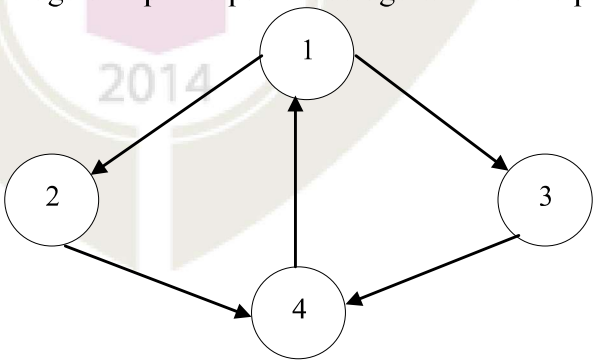
(b) Explain the major difference between control flow testing and data flow testing. **(6)**

OR

14. (a) Explain seven types of mutation operators with neat examples? **(14)**

15. (a) Explain touring, side trips and detours with a neat example **(7)**

(b) Explain simple path coverage and prime path coverage with the help of CFG given below? **(7)**



OR

16. (a) Draw CFG fragment for
(i) Simple *if* (ii) Simple *while* loop (iii) Simple *for* loop **(7)**

(b) Explain the following concepts with examples? **(7)**
(i)Call graph (ii) Inheritance graph (iii) Coupling du-pairs

17. (a) What are the four important steps in functional testing? **(7)**

(b) Briefly explain input domain modelling approaches? **(7)**

OR

18. (a) Consider the triangle classification program with a specification: **(6)**
The program reads floating values from the standard input. The three values *A*, *B*, and *C* are interpreted as representing the lengths of the sides of triangle. The program then prints a message to the standard output that states whether the triangle, if it can be formed, is scalene, isosceles, equilateral, orright angled. Determine the following for the above program:
(i) For the boundary condition $A + B > C$ case (scalene triangle), identify test cases to verify the boundary.
(ii) For the boundary condition $A = C$ case (isosceles triangle), identify testcases to verify the boundary.
(iii) For the boundary condition $A = B = C$ case (equilateral triangle), identify testcases to verify the boundary.

(b) Develop a decision table to generate test cases for this specification. **(8)**

19. (a) Explain the importance of grey box testing, its advantages and disadvantages? **(9)**

(b) Explain the concept of symbolic execution tree? **(5)**

OR

20. (a) Consider the code fragment given below: - **(7)**

1. POWER: PROCEDURE(X, Y);
2. Z ← 1;

3. J ← 1;
4. LAB: IF Y ≥ J THEN
5. DO; Z← Z * X;
6. J ← J + 1;
7. GO TO LAB; END;
8. RETURN (Z) ;
9. END;

a) Explain Symbolic execution of POWER (αl, α2).

(b) Explain Execution tree for POWER (αl, α2). **(7)**

TEACHING PLAN

No	Contents	No of Lecture Hrs (35 hrs)
Module 1 (Introduction to Software Testing) -(7 Hours)		
1.1	Some Popular Errors– Ariane 5, Therac 25, Intel Pentium Bug.	1 Hour
1.2	What is Software testing? Why should it be tested? Software Quality, Role of Testing.	1 Hour
1.3	Testing Process - Level 0 thinking, Level 1 thinking, Level 2 thinking, Level 3 thinking, Level 4 thinking.	1 Hour
1.4	Software Testing Terminologies- Verification, Validation and Testing, Faults, Error and Bug, Test cases, Coverage Criteria.	1 Hour
1.5	Types of Testing- Unit testing, integration testing, System testing, Acceptance testing, Beta testing	1 Hour
1.6	Functional testing, Stress testing, Performance testing, Usability testing and Regression testing.	1 Hour
1.7	Testing Methods - Black Box testing, White Box testing, Grey Box testing.	1 Hour
Module 2 (Unit testing)- (6 Hours)		
2.1	Concept of Unit testing, Static Unit Testing	1 Hour

2.2	Dynamic Unit testing - Control Flow testing, Data Flow testing, Domain testing, Functional Program testing.	1 Hour
2.3	Mutation testing - Mutation and Mutants, Mutation operators, Mutation score.	1 Hour
2.4	Junit - Framework for Unit testing.	1 Hour
2.5	Case Study - Mutation testing using Junit	1 Hour
2.6	Case Study - Mutation testing using Muclipse	1 Hour
Module 3 (Unit Testing:- White Box Approaches)- (8 Hours)		
3.1	Structural Graph Coverage Criteria - Node/vertex coverage, Edge coverage, Edge pair coverage, Path coverage	1 Hour
3.2	Complete path coverage, Prime path coverage, Complete round trip coverage, Simple round trip coverage.	1 Hour
3.3	Data Flow Criteria - du paths, du pairs	1 Hour
3.4	Subsumption Relationships among Graph Coverage Criteria	1 Hour
3.5	Graph Coverage for Source Code – Control Flow Graphs (CFG) for code, CFG: If statement, CFG: If statement with return, CFG: Switch-case, CFG: Loops, CFG: Exceptions (try-catch). Example program - Statistics	1 Hour
3.6	Graph Coverage for Design Elements – Structural graph coverage and data flow graph coverage for design elements	1 Hour
3.7	Case Study - Graph Based testing using JUnit Framework. (Lecture 1)	1 Hour
3.8	Case Study - Graph Based testing using JUnit Framework. (Lecture 2)	1 Hour
Module 4 (Unit Testing:- Black Box Approaches) -(7 Hours)		
4.1	Domain Testing / Input Space Partitioning - Partitions of a set.	1 Hour
4.2	Input domain modelling - Interface-based approach, Functionality-based approach.	1 Hour

4.3	Multiple partitions of the input domain - All Combinations Coverage (ACoC), Each Choice Coverage (ECC), Pair-wise Coverage, T-wise Coverage, Base Choice Coverage, Multiple Base Choices Coverage.	1 Hour
4.4	Functional Testing - Functional Testing Concepts of Howden. Important Steps.	1 Hour
4.5	Types of Functional testing - Equivalence Class Partitioning, Boundary Value Analysis	1 Hour
4.6	Decision Tables, Random Testing.	1 Hour
4.7	Case Study - Black Box testing approaches using JUnit.	1 Hour
Module 5 (Grey Box Testing Approaches)- (7 Hours)		
5.1	Introduction to Grey Box testing - Why Grey Box testing, Gray Box Methodology, Advantages and Disadvantages.	1 Hour
5.2	Techniques of Grey Box Testing - Matrix Testing, Regression Testing, Orthogonal Array Testing or OAT, Pattern Testing.	1 Hour
5.3	An Introduction to Pex - Parameterized Unit Testing, The Testing Problem.	1 Hour
5.4	Symbolic Execution – Example, Symbolic execution tree.	1 Hour
5.5	Case Study – PEX (Lecture 1)	1 Hour
5.6	Case Study – PEX (Lecture 2)	1 Hour
5.7	Case Study – PEX (Lecture 3)	1 Hour