Unit 1: Perspective on Testing, Examples

Chapter 1 & 2 from Dr. Paul C. Jorgensen

Sl no	Question Description	Marks
1.	Make a Venn Diagram that reflects a part of the following statement: " … we have left undone that which we ought to have done, and we have done that which we ought not to have done …"	5 Marks
2.	Describe each of the eight regions in Figure 1.4. Can you recall examples of these in software you have written?	6 Marks
3.	Define the following 1) Error 2) Fault 3)Failure 4) Incident 5) Test Case	5 Marks
4.	Define Software Testing. Why Software Testing is necessary?	5 Marks
5.	Explain the significance of Testing Life Cycle	5 Marks
6.	When are defects introduced in SDLC? How can we minimize the impact of those defects during each Phase	7 Marks
7.	Why Test Cases are important? Explain the typical test case information.	5 Marks
8.	What is False Positive & False Negative with respect to test cases? Explain three examples for each.	6 Marks
9.	Explain the two fundamental approaches for Identifying test cases Or Explain Functional Testing & Structural Testing with diagram Or Explain Black Box & White Box Testing	5 Marks
10.	What is Software Quality Assurance (SQA)?	4 Marks
11.	Explain Test Driven Development (TDD) Approach	5 Marks
12.	List & explain the advantages and drawbacks of functional testing approach	4 Marks
13.	List & explain the various error and fault taxonomies based upon a) Input / Output Faults b) Logic Faults c) Computation Faults d) Interface Faults e) Data Faults	6 Marks Each
14.	What is Sins of omission and Sins of commission	4 Marks
15.	With Diagram, Explain Levels of Testing	7 Marks
16.	Define Unit Testing , Integration Testing & System Testing	5 Marks

17.	Write a generalized Pseudo code program for Triangle Problem	6 Marks
18.	Make a Flow chart for the traditional triangle program Implementation	5 Marks
19.	Show the dataflow diagram for structural triangle program Implementation	4 Marks
20.	Give the Problem definition & structural pseudo code for NextDate Function	8 Marks
21.	Along with the problem definition, Discuss the Commission problem with suitable example.	7 Marks
22.	Write structural pseudo code for Commission Problem	5 Marks
23.	Along with the problem definition, Discuss the Simple ATM (SATM) problem with suitable SATM Screens.	7 Marks
24.	Along with the problem definition, Discuss the Currency Converter with suitable example.	5 Marks
25.	Revisit the traditional Triangle Program flowchart in Figure 2.1. Can the variable match ever have the value of 4? Of 5? Is it ever possible to "execute" the following sequence of numbered boxes: 1, 2, 5, 6?	4 Marks
26.	One common addition to the Triangle Problem is to check for right triangles. Three sides constitute a right triangle if the Pythagorean Relationship is satisfied: c2 = a2 + b2	5 Marks
27.	One of the folk tales of software lore describes a disgruntled employee who writes a payroll program. The program contains logic that checks for the employee's identification number before producing paychecks. If the employee is ever terminated, the program creates havoc. Discuss this situation in terms of the error, fault, and failure pattern, and decide which form of testing would be appropriate	4 Marks `
28.	Recall the discussion from Chapter 1 about the relationship between the specification and the implementation of a program. If you study the implementation of NextDate carefully, you will see a problem. Look at the CASE clause for 30 day months (4, 6, 9, 11). There is no special action for day = 31. Discuss whether or not this implementation is "correct". Repeat this discussion for the treatment of values of day ≥ 29 in the CASE clause for February.	3 Marks
29.	In Chapter 1, we mentioned that part of a test case is the expected output. What would you use as the expected output for a NextDate test case of June 31, 1812? Why?	2 Marks
30.	What will the triangle2 program do for the sides -3, -3, 5? Discuss this in terms of the considerations we made in Chapter 1.	5 Marks

Unit 2: Boundary Value Testing, Equivalence Class Testing & Decision Table Based Testing

Chapter 5, 6 &7 from Dr. Paul C. Jorgensen

Sl No	Question Description	Marks
1.	Why Boundary value testing belong to functional testing technique	2 Marks
2.	Explain with appropriate diagram Boundary Value Analysis technique for a function of **two** variables by generating all possible test cases.	6 Marks
3.	Explain with appropriate diagram Boundary Value Analysis technique for a function of **three** variables by generating all possible test cases	8 Marks
4.	Explain in details the two ways to generalize Boundary Value Analysis Technique.	4 Marks
5.	List out & explain the various limitations of Boundary Value Analysis.	5 Marks
6.	Explain with appropriate diagram the mechanism to generate test cases for a function of two variables. a. Robustness Testing b. Worst-Case Testing c. Robust Worst-Case Testing	7 Marks Each
7.	What do you mean by Special Value testing	3 Marks
8.	Develop a formula for the number of robustness test cases for a function of n variables.	2 Marks
9.	Develop a formula for the number of robust worst-case test cases for a function of n variables.	3 Marks
10.	Make a Venn diagram showing the relationships among test cases from boundary value analysis, robustness testing, worst-case testing, and robust worst-case testing.	4 Marks
11.	What happens if we try to do output range robustness testing? Use the commission problem as an example.	4 Marks
12.	What is Random Testing	4 Marks

13.	Explain the problem definition along with the all possible BVA test cases for 1. Triangle Problem 2. NextDate Function 3. Commission Problem	15 Marks
14.	List and explain the drawbacks of Boundary Value Testing	4 Marks
15.	List out some of the guidelines and observations about Boundary Value Analysis Testing	4 Marks
16.	What is the motivation behind Equivalence Classes & how does it addresses the problem of BVA method.	6 Marks
17.	Illustrate the concept of equivalence classes for function of two variables.	6 Marks
18.	Explain with appropriate diagram Equivalence Class testing technique for a function of **two** variables for generating all possible test cases. a. Weak Normal Equivalence Class Testing b. Strong Normal Equivalence Class Testing c. Weak Robust Equivalence Class Testing d. Strong robust Equivalence Class Testing	6 Marks Each
19.	Starting with the 36 strong normal equivalence class test cases for the NextDate function, revise the day classes as discussed, and then find the other nine test cases.	5 Marks
20.	If you use a compiler for a strongly typed language, discuss how it would react to robust equivalence class test cases.	3 Marks
21.	Revise the set of weak normal equivalence classes for the extended triangle problem that considers right triangles.	4 Marks
22.	Compare and contrast the single/multiple fault assumption with boundary value and equivalence class testing.	6 Marks
23.	Explain Output Range equivalence class test cases for commission problem.	5 Marks
24.	List out all possible Valid & Invalid Equivalence Classes for generating test cases for 1. Triangle Problem 2. NextDate Function 3. Commission Problem	10 Marks
25.	List out some of the guidelines and observations about Equivalence Class Testing	3 Marks

26.	List out 10 Sample test cases for **Triangle, Commission Problem and NextDate Function** by considering these techniques a. Weak Normal Equivalence Class Testing b. Strong Normal Equivalence Class Testing c. Weak Robust Equivalence Class Testing d. Strong robust Equivalence Class Testing	15 Marks
27.	Explain the functional testing method done by decision table based approach	4 Marks
28.	Illustrate the Portions of a Decision Table.	5 Marks
29.	Explain the technique for building a decision table for triangle problem.	7 Marks
30.	Show & explain the refined decision table for triangle problem.	6 Marks
31.	How can we illustrate the decision table with mutually exclusive conditions? How to handle don't care entries	6 Marks
32.	Illustrate Concept of Rule Count using Decision Table for Triangle Program	6 Marks
33.	When we conclude our decision table as Redundant & Inconsistent Explain with the help of decision table	6 Marks
34.	Explain with appropriate decision table to generate all possible test cases for NextDate Function by considering a. Decision Table with 256 Rules b. Decision Table with 36 Rules c. Decision Table with 22 Rules	20 Marks
35.	Along with the problem definition Explain all possible Decision Table based Test cases for 1. Triangle Problem 2. NextDate Function 3. Commission Problem	15 Marks
36.	Develop a decision table and additional test cases for the right triangle addition to the Triangle Problem	4 Marks

37.	Develop a decision table for the “second try” at the NextDate function. At the end of a 31-day month, the day is always reset to 1. For all non-December months, the month is incremented; and for December, the month is reset to January, and the year is incremented.	6 Marks
38.	Expand the Commission Problem to consider "violations" of the sales limits. Develop the corresponding decision tables and test cases for a "company friendly" version and a "salesperson friendly" version.	6 Marks
39.	Discuss how well decision table testing deals with the multiple fault assumption.	1 Marks
40.	List out some of the guidelines and observations about Decision Table Based Testing	4 Marks
41.	Mention the characteristics features or properties of a. Weak Normal Equivalence Class Testing b. Strong Normal Equivalence Class Testing c. Weak Robust Equivalence Class Testing d. Strong robust Equivalence Class Testing	2 Marks Each

Unit 3: Path Testing & Data Flow Testing

Chapter 9 from Dr. Paul C. Jorgensen

Sl No	Question Description	Marks
1.	Mention the characteristics of structural testing	2 Marks
2.	Define Program Graph considering Node 'i' & Node 'j' as Statements.	3 Marks
3.	Explain the procedure to construct a program graph (Control Flow Graph) for any program with the help of an example	8 Marks
4.	Mention the pseudo code of triangle program and make a program graph of the same.	10 Marks
5.	Mention the types of structural testing	5 Marks
6.	Define Path Testing	3 Marks
7.	Define Decision to Decision Path defined by E. F .Miller	3 Marks
8.	Given a sample triangle program graph, Trace the graph showing the path traversed by considering the value of variables a, b, & c which forms a Equilateral, Isosceles & Scalene, Not a Triangle condition.	12 Marks
9.	Considering the triangle program graph, Can you find values of a, b, and c such that the path traverses nodes 7 and 18? If not, justify the reason for your answer	6 Marks
10.	Considering Schach's "Program", Explain with the help of program graph the Common Objection: Trillions of Paths. List out all possible paths possible to write test cases for the same	10 Marks
11.	Define DD paths with example which initial node, internal nodes & final nodes	4 Marks
12.	Mention the DD-path of nodes in a program graph with the help of 'cases' each node belong to.	5 Marks
13.	Considering the program graph of triangle program, show the DD path formation by identifying the types of DD-path. Mention DD-Path Name & Case of Definition.	10 Marks
14.	Define DD path Graph	2 Marks

15.	By mentioning the program code of commission problem & NextDate Function , Show DD - Path Graph for the same	20 Marks
16.	Mention the significance of Test Coverage Metrics	4 Marks
17.	List out all possible Structural Test Coverage Metrics	6 Marks
18.	Discuss these metrics with an example C_0 , C_1 , C_{1p} , C_2, C_d, C_{MCC}, C_{ik} , C_{stat} , C_∞	15 Marks
19.	Mention Statement & Predicate (C_0) Coverage with an example, Mention the path traversed for each test case input.	7 Marks
20.	Mention Decision Testing (DD Path C_1) Coverage with an example, Mention the path traversed for each test case input.	7 Marks
21.	Mention Condition Testing (C_{1p}) Coverage with an example, Mention the path traversed for each test case input.	7 Marks
22.	Mention Multiple Compound Condition (C_{MCC}) Coverage with an example, Mention the path traversed for each test case input.	7 Marks
23.	Mention Path Testing (C_∞) Coverage with an example, Mention the path traversed for each test case input.	7 Marks
24.	Mention Dependent DD-Path Pairs Coverage (Cd) Coverage with an example, Mention the path traversed for each test case input.	7 Marks
25.	Mention Loop Coverage (C_2) Coverage with an example, Mention the path traversed for each test case input.	7 Marks
26.	Mention Statistically Significant Path Coverage (C stat) Coverage with an example, Mention the path traversed for each test case input.	7 Marks
27.	With a Diagram, mention the kind of loops possible in program graph	10 Marks
28.	Why loop testing important? Mention the strategy for Loop Testing	6 Marks
29.	Mention these loops with a neat diagram. 1. Simple Loops 2. Nested Loops 3. Concatenated Loops 4. Unstructured Loops	12 Marks

30.	What are Test Coverage Tools	4 Marks
31.	Mention the significance & motivation of Basis Path Testing	5 Marks
32.	Explain McCabe Algorithm to determine Basis Paths with an McCabe original graph	8 Marks
33.	Mention the incidence Matrix which shows Path / Edge Traversal for McCabe Graph	8 Marks
34.	Considering Triangle Program, Apply McCabe's Basis Path Testing, Explain the observation done & identify all possible (Feasible & Infeasible) topological paths.	12 Marks
35.	Considering Triangle Program, Apply McCabe's Basis Path Testing, Explain the observation done & identify all possible infeasible topological paths. Illustrate the significance of such paths.	8 Marks
36.	Mention McCabe's notion of Essential Complexity of a program graph	4 Marks
37.	When is the essential complexity of a Structured Program = 1	5 Marks
38.	Illustrate with a neat diagram a program graphs of Structured Programming Constructs like 1. Sequence 2. Pre-test loop 3. Post-test loop 4. If-Then 5. If-Then-Else 6. Case 7. While Loop 8. Do While Loop	3 Marks Each
39.	Considering the DD-path Graph of the Pseudo-Code program for triangle program. Explain Condensation with respect to the Structured Programming Constructs until the final try. Or Show the essential complexity of the DD- Path Graph	10 Marks

40.	Considering the DD-path Graph of the Schach's program Explain Condensation with respect to the Structured Programming Constructs until the final try. Or Show the essential complexity of the Schach's DD- Path Graph	8 Marks
41.	With a neat Diagram, Illustrate Violations of Structured Programming Constructs. How can we reduce such practice	10 Marks
42.	List out some of the guidelines and observations about Path Testing with the help of Venn diagram	5 Marks
43.	Find the cyclomatic complexity of the graph in Figure 9.2	3 Marks
44.	Identify a set of basis paths for the graph in Figure 9.2.	6 Marks
45.	Discuss McCabe's concept of "flipping" for nodes with outdegree ≥ 3	3 Marks
46.	Suppose we take Figure 9.2 as the DD-Path graph of some program. Develop sets of paths (which would be test cases) for the C0, C1, and C2 metrics.	7 Marks
47.	Develop multiple-condition coverage test cases for the pseudo code triangle program. (Pay attention to the dependency between statement fragments 14 and 16 with the expression (a = b) AND (b = c).)	10 Marks
48.	Rewrite the program segment 14–20 such that the compound conditions are replaced by nested if-then else statements. Compare the cyclomatic complexity of your program with that of the existing version.	10 Marks

49.		Look carefully at the original statement fragments 14–20. What happens with a test case (e.g., a = 3, b = 4, c = 3) in which a = c? The condition in Line 14 uses the transitivity of equality to eliminate the a = c condition. Is this a problem?	10 Marks
50.		(For mathematicians only.) For a set V to be a vector space, two operations (addition and scalar multiplication) must be defined for elements in the set. In addition, the following criteria must hold for all vectors x, y, and z ∈ V, and for all scalars k, m, 0, and 1: a. if x, y ∈V, the vector x + y ∈ V. b. x + y = y + x. c. (x + y) + z = x + (y + z). d. there is a vector 0 ∈V such that x + 0 = x. e. for any x ∈V, there is a vector -x ∈V such that x + (-x) = 0. f. for any x ∈V, the vector kx ∈V. g. k(x + y) = kx + ky. h. (k + m)x = kx + mx. i. k(mx) = (km)x. j. 1x = x. How many of these 10 criteria hold for the “vector space” of paths in a program?	10 Marks
51.	Design, develop, code and run the program in any suitable language to implement the binary search algorithm. Determine the basis paths and using them derive different test cases, execute these test cases and discuss the test results.		20 Marks

52.	Design, develop, code and run the program in any suitable language to implement the quicksort algorithm. Determine the basis paths and using them derive different test cases, execute these test cases and discuss the test results.	20 Marks
53.	Design, develop, code and run the program in any suitable language to implement an absolute letter grading procedure, making suitable assumptions. Determine the basis paths and using them derive different test cases, execute these test cases and discuss the test results.	20 Marks
54.	Propose a definition for the length of a path in a graph	2 Marks
55.	What loop(s) is/are created if an edge is added between nodes n5 and n6 in the graph in Figure 4.1? Suppose the added edge between n5, and n6 is edge e6. Then nodes n1, n2, n5, n6, and n4, are a loop.	2 Marks
56.	Convince yourself that 3-connectedness is an equivalence relation on the nodes of a digraph.	3 Marks
57.	Compute the cyclomatic complexity for each of the structured programming constructs in Figure 4.5	3 Marks Each
58.	The digraphs in Figure 4.15 were obtained by adding nodes and edges to the digraph in Figure 4.3. Compute the cyclomatic complexity of each new digraph, and explain how the changes affected the complexity.	10 Marks

59.	Define & Explain with an help of an graph 1. Degree of a Node 2. Incidence Matrix 3. Adjacency Matrix 4. Paths in a graph 5. Connectedness 6. Condensation Graph 7. Directed Graph 8. Indegree & Outdegree 9. Type of Nodes 10. Adjacency Matrix of a Digraph 11. Semi Paths 12. Reachability Matrix 13. Strong Components 14. Condensed Diagrams 15. Cyclomatic Number of a graph.	2 Marks Each
60	Define & Explain with an help of an graph N- Connectedness – (0-,1-,2-,3- Connected)	5 Marks

Chapter 10 from Dr. Paul C. Jorgensen

Sl No	Question Description	Marks
1.	Define data flow testing & mention the motivation of data flow testing	4 Marks
2.	Differentiate between static and dynamic data flow testing	5 Marks
3.	Explain Define / Use testing	4 Marks
4.	Mention all possible Data Flow Analysis Anomalies	6 Marks
5.	With respect to a Program Graph considering a Variable V , Define Defining Node , Usage Node, Predicate Use, Computation Use, du-paths, definition clear	10 Marks
6.	Illustrate the significance of DU- Paths and DC- Paths	6 Marks
7.	Mention the program graph for commission problem	5 Marks

8.	Considering the program graph of Commission problem, show the DD path formation by identifying the types of DD-path. Mention DD-Path Name & Case of Definition.	10 Marks
9.	Considering the program graph of Commission problem, Mention few Define / Use nodes for Variables in the commission Problem	10 Marks
10.	Illustrate with the help of code fragments for Locks, Show Predicate Use (P-Use) & Computation Use (C- Use)	6 Marks
11.	Illustrate with the help of code fragments for few variables show selected DU paths and mention whether it is definition clear	8 Marks
12.	Illustrate with the help of code fragments for **Commission variable** show selected DU paths and mention whether it is definition clear	8 Marks
13.	Explain the Coverage Metrics Based on du- Path Or Explain Rapps-Weyuker Coverage hierarchy of dataflow coverage metrics	10 Marks
14.	Design, develop, code and run the program in any suitable language to solve the commission problem. Analyze it from the perspective of dataflow testing, derive different test cases, execute these test cases and discuss the test results.	20 Marks

Unit 4: Levels of Testing, Integration Testing

Chapter 12 & 13 from Dr. Paul

Sl No	Question Description	Marks
1.	With a neat diagram, explain in brief the waterfall spin-offs.	10 Marks
2.	Give differences among the three spin-offs of the waterfall model	03 Marks
3.	Explain the specification based life cycle models with neat diagrams a)Rapid prototyping b)Executable specification model.	10 Marks
4.	Write a short note on SATM system and explain the SATM terminal with a neat diagram.	10 Marks
5.	Illustrate SATM with help of a neat diagram for the: a. Data flow model b. ER model	10 Marks
6.	Draw the finite state machines for upper level SATM and pin entry SATM.	8 Marks
7.	Explain the significance of various output screens of SATM.	10 Marks
8.	Draw and explain the context diagram of SATM system	7 Marks
9.	With pseudo codes, explain the working of the following functions: a) GetPINforPAN b) GetPIN of an SATM machine.	6 Marks
10.	SATM Functional Decomposition Tree	10 Marks
11.	Define call graph. Give an example.	5 Marks
12.	Explain the basis on which Decomposition-Based Integration testing is performed.	2 Marks
13.	List the advantages and disadvantages of Decomposition-Based Integration testing.	6 Marks
14.	Explain pair wise and neighborhood integration with appropriate examples. Give the pros and cons of call graph-based testing.	7 Marks

Unit 7: Fault-Based Testing, Test Execution

Chapter 16 & 17 from Mauro Pezze & Michal Young

Sl No	Question Description	Marks
1.	Explain the significance of fault-based testing	5 Marks
2.	Illustrate with an example how would we estimate test suite Quality	6 Marks
3.	List the basic assumptions of fault-based testing	5 Marks
4.	Mention the terminology used in Fault-based testing	6 Marks
5.	Explain Mutation Analysis & Testing with an example	8 Marks
6.	Mention the terminology used in Mutation Analysis	5 Marks
7.	Mention the competent programmer hypothesis & Coupling effect hypothesis	6 Marks
8.	When will we find Live Mutant or How Mutant Survive	5 Marks
9.	Explain valid Mutant and useful mutant with an example	6 Marks
10.	Illustrate with the help of code snippet some of the mutation operator	7 Marks
11.	Write a Short on i) Weak Mutation Analysis ii) Statistical Mutation Analysis	5 Marks Each
12.	We have described weak mutation as continuing execution up to the point that a mutant is killed, then restarting execution of the original and mutated program from the beginning. Why doesn't execution just continue after killing a mutant? What would be necessary to make continued execution possible?	3 Marks
13.	Motivation the need for the component programmer and the coupling effect hypotheses. Would mutation analysis still make sense if these hypotheses did not hold? Why?	5Marks

14.	Generate some invalid, valid-but-not-useful, useful, equivalent and nonequivalent mutants for the program in Figure 16.1 using mutant operators from Figure 16.2	5 Marks
15.	Why Test Automation is needed? List some of the key features of Test Automation	8 Marks
16.	What is Scaffolding? Explain the purpose of Scaffolding	6 Marks
17.	Define Test Driver , Test Stubs & Test Harness	5 Marks
18.	Distinguish between Generic Verses Specific Scaffolding	5 Marks
19.	What is Test Oracle? What are its advantages and disadvantages over human oracle? Define Common Test Oracles	7 Mark
20.	Given a program and test suite T, Explain 3 steps of mutation analysis along with an example	6 Marks
21.	Define Capture & Replay Or Record & Replay	4 Marks
22.	Explain in detail Scaffolding and test oracles w.r.t test execution	10 Marks
23.	Explain with neat diagram, Comparism based oracle & Self checking code as oracle	10 Marks
24.	Define Partial Oracle	4 Marks
25.	Voluminous output can be a barrier to naïve implementations of comparison-based oracles. For example, sometimes we wish to show that some abstraction of program behavior is preserved by a software change. The naïve approach is to store a detailed execution log of the original version as predicted output, and compare that to detailed execution log of the modified version. Unfortunately, a detailed log of a single execution is quite lengthy, and maintaining detailed logs of many test cases executions may be impractical. Suggest more efficient approaches to implementing comparison-based test oracles when it is not possible to store the whole output.	6 Marks

26.	We have described as an ideal but usually unachievable goal that test oracles could be derived automatically from the same specification statement used to record and communicate the intended behavior of a program or module. To what extent does the “test first” approach of extreme programming (XP) achieve this goal? Discuss advantages and limitations of using test cases as a specification statement.	4 Marks
27.	Often we can choose between on-line self-checks (recognizing failures as they occur) and producing a log of events or states for off-line checking. What considerations might motivate one choice or the other?	10 Marks

28. Consider the C function in figure 16.4 used to determine whether a misspelled word differs from a dictionary word by at most one character, which may be a deletion, an insertion, or a substitution(e.g., "text" is edit distance 1 from "test" by a substitution, and edit distance 1 from "tests" by deletion of "s"). — 10 Marks

Suppose we seed a fault I line 27, replacing s1+1 by s1+0. Is there a test case that will kill this mutant using weak mutation, but not using strong mutation? Display such a test case if there is one, or explain why there is none.

```
1
2       /*edit1 (s1, s2) returns TRUE iff s1 can be transformed to s2
3       *by inserting, deleting, or substituting a single character, or
4       *by a no-op (i.e., if they are already equal).
5       */
6       int edit1 (char *s1, char *s2) {
7       if (*s1 == 0) {
8         if (*s2 == 0) return TRUE;
9         /* Try inserting a character in s1 or deleting in s2 */
10      if (*(s2+1) == 0) return TRUE;
11      return FALSE;
12      }
13      if (*s2 == 0) {/* Only match is by deleting last char from s1 */
14        if (*(s1+1) == 0) return TRUE;
15      return FASLE;
16      }
17      if (*s2 == 0) {/* Only match is by deleting last char from s1 8/
18        if (*(s1+1) == 0) return TRUE;
19        return edit1 (s1 +1, s2+2);
20      }
21
22      /*Mismatch; only dist 1 possibilities are identical strings after
23        *inserting, deleting, or substituting character
24      */
25
26      /* Substitution: We "look past" the mismatched character */
27      if (strcmp (s1+1, s2+1) == 0) return TRUE;
28      /* Deletion: look past character in s2 */
29      if (strcmp (s1+1, s2) ==0) return TRUE;
30      /* insertion: look past character in s2 */
31      if (strcmp (s1, s2+1) ==0) return TRUE;
32      return FALSE;
33      }
```

Figure 16.4: c function to determine whether one string is within edit distance 1 of another.

Unit 8: Planning and Monitoring the Process, Documenting Analysis and Test

Chapter 20 & 24 from Mauro Pezze & Michal Young

Sl No	Question Description	Marks
1.	Mention the significance and purpose of Planning and monitoring the process	5 Marks
2.	In Quality and Process plan, specify how we perform verification steps on intermediate typical artifacts by below mentioned mechanisms 1. Internal Consistency Check 2. External Consistency Check 3. Generation of Correctness Conjectures	6 Marks
3.	Explain with the help of diagram the significance and activities involved in Cleanroom Quality Process	7 Marks
4.	Explain with diagram the steps involved in Software reliability engineered testing (SRET) approach	7 Marks
5.	Explain with the help of diagram the significance and activities involved in Extreme Programming (XP) Methodology	8 Marks
6.	Mention Test & Analysis Strategies, Describe various factors that particularize these strategies	8 Marks
7.	Explain Test & Analysis Plans, Mention the questions those should be addressed in quality strategies	6 Marks
8.	Mention the key elements of a Plan	5 Marks
9.	Mention the significance of quality goals	6 Marks
10.	Mention how the tasks are scheduled with an help of sample schedule	10 Marks
11.	Mention critical path and critical dependencies	5 Marks Each
12.	Illustrate the possible schedule with different risks and resources allocation for 1. Critical Schedule 2. Unlimited Resources 3. Limited Resources	7 Marks Each

13.	Write a short Notes on: 1. Quality & Process 2. Quality Team	6 Marks
14.	Describe the standard organization of a plan	5 Marks
15.	Provide briefly an overview of some risks generic to process management	10 Marks
16.	Mention the significance of Risk Planning, With an Example risks mention the associated control Tactics (strategies) for 1. Personnel Risks 2. Technology Risks 3. Schedule Risks 4. Development Risks 5. Test Execution Risks 6. Requirement Risks	10 Marks
17.	Mention Contingency Plans	5 Marks
18.	What are faults? Define standard severity levels for RCA	7 Marks
19.	Explain the evolution of the plan	5 Mark
20.	Illustrate with a neat graph, A typical distribution of faults for system builds through time	6 Marks
21.	Define Process improvement wrt to Monitoring and improvement within a project or across multiple projects	6 Marks
22.	Mention the orthogonal Defect Classification and Analysis	7 Marks
23.	Explain ODC Activities Performed for preliminary analysis 1. Distribution of fault types versus activities 2. Distribution of triggers over time during field test 3. Age distribution over target code 4. Distribution of fault classes over time	10 Marks
24.	Write a short notes on improving the process (current & Next Process)	5 Marks

25.	Write a short note on Root Cause Analysis and mention these four steps 1. *What* are the faults? 2. *When* did faults occur? When and when were they found? 3. *Why* did faults occur? 4. *How* could faults be prevented?	10 Marks
26.	Mention the standard fault severity levels for RCA	4 Marks
27.	Describe 80/20 or Pareto Rule for fault classification in RCA	5 Marks
28.	Mention the roles and responsibility at the Tactical Level	6 Marks
29.	Mention the roles and responsibility at the Strategic Level	6 Marks
30.	Mention ODC Classification of Defect types for Customer Impact	6 Marks
31	Mention ODC Classification of Defect Types for Targets Design and Code	6 Marks