Group Discussion Report 3
Software Quality Process Implementation Activities

Group Member's Names:
1. Akshay Kalsotra
2. Malika Hafiza Pasha
3. Mustafa Quraishi
4. Sajeel Mohammed Abdul
5. Shoaibuddin Mohammed

Part I Software Testing

The Clean Fuel Ltd. Is the owner of a chain of more than a hundred gas stations. The gas stations are operated by contractors. The monthly billing system includes the following table of basic monthly rent rates ($).

		The number of pumps		
		Up to 4	5–9	10 and more
The station's area (thousands of square feet)	**Up to 10.00**	1,500	2.200	2,500
	10.01-20.00	2,000	2,500	3,300
	20.01-40.00	2,400	2.700	4,000
	40.01 and more	2,800	3,000	4,500

a. Added rent for restaurant in the gas station:

A restaurant – extra 100%.

b. Deductions according to the road level:

National freeway – no reduction.

Local, urban road – 20% deduction

Discuss how to set up the test cases and complete the following requirement.

1. List the variables and equivalence classes according to the equivalence class partitioning method in the following table.

No	Variable	Equivalence Class			
		1(Road Type)	2(Restro)	3(Area sq ft)	4(No of pumps)
1	Sajeel	National Freeway	No	Upto 10000	Upto 4
2	Malika	National Freeway	Yes	10001-20000	10 or more
3	Mustafa	Local Road	No	20001-40000	5-9
4	Akshay	Local Road	Yes	Upto 400001	Upto 4

2. List the IECs for the Clean Fuel's basic monthly rent rates in the following table,

The Variable	Valid Equivalence Classes	Representing Values		Invalid Equivalence Classes	Representing values for invalid ECs
		Value for the valid ECs	Boundary values		
Sajeel	National freeway, Restro:No Area: Upto 10,000 No of pumps:4	National Highway, Restro: No Area: 9000, 6000 No of pumps: 1) 2 2) 3	9999 4 1000 1	Area less than 0, Pumps greater than 4 or less than 0, Alphanumeric	10,001 10,005 5,9 Local/Urban road
Malika	Local/Urban road, Restro: Yes	Local/Urban road, Restro: Yes	10001, 19999 10	Area less than 10,000 and greater than 20,000	9500 25000 1-9

	Area: 10,001-20,000 No of pumps: 10 or more	Area: 15000 16000 No of pumps: 1) 12 2) 13 3) 14	1	Pumps less than 10 Alphanumeric	No Local/Urban road
Mustafa	Local/Urban road, Restro: No Area:20,001-40,000 No of pumps: 5-9	Local/urban road, Restro: No Area: 25000 26000 No of pumps: 1)7 2)8	1) 20001 2) 20002 3) 39500 8 9	Area less than 20000 area more than 40000 Pumps greater than 5 or more than 9 Alphanumeric	19,500 45,000 3 15 National Highway, Yes
Akshay	Local/urban road, Restro: Yes Area: upto 10000 No of pumps: Uptp 4	Local/urban road, Restro: Yes Area: 9600 6000 No of pumps: 1)2 2)3	9500 1000 4 1 Local road	Area less than 0 or more than 10,000 Pumps less than 1 or greater than 4 Alphanumeric	10,001 40 National Highway No restro

3. List the OECs for the Clean Fuel's basic monthly rent rates in the following table.

	The number of pumps		
	Upto 4	5-9	10 or more

Restaurant		None		Exists		None		Exists		None		Exists	
Road Level		NF	Loc	NF	Loc	NF	Loc	NF	Loc	NF	Loc	NF	Loc
Station's area	Uptp 10,000	1500	1200	3000	2400	2200	1760	4400	3520	2500	2000	5000	4000
	10.0-20	2000	1600	4000	3200	2500	2000	5000	4000	3000	2640	6600	5280
	20.1-40	2400	1920	4800	3840	2700	2160	5400	4320	4000	3200	8000	6400
	40.1 to more	2800	2240	5600	4480	3000	2400	6000	4800	4500	3600	9000	72000

4. List the required test cases according to the equivalence class partitioning methods in the following table.

Test Case type	Test case no	Variable Values in the test case				Expected test case results
		Var 1 (Road type)	Var 2 (Restro)	Var 3 (No of pumps)	Var 4 (Area)	
Valid IEC test cases	1	National Freeway	Yes	2	6000	3000
	2	Local road	Yes	6	13000	4000
	3	National Freeway	No	10	23000	4000

	4	Local road	No	5	45000	2500
Boundary conditions	5	Local road	Yes	4	20005	3840
	6	National Freeway	No	1	500	1500
	7	Local road	No	9	9999	1760
	8	Local road	Yes	10	40002	7200
	9	National Freeway	Yes	5	10005	5000
	10	National Freeway	No	20	60000	4500
Invalid	11	National Freeway	Maybe	2	1000	Invalid restro info
	12	Suburbs	No	10	40000	Invalid road info
	13	Local	Yes	10	20000	Invalid pump info
Test cases for OEC	14	National Freeway	Yes	3	3343	3000
	15	National Freeway	No	15	11500	3300
	16	National Freeway	Yes	7	27000	5400
	17	Local road	No	8	38000	2160
	18	Local road	Yes	11	1100	4000
	19	Local road	No	44	44444	3600
	20	Local road	Yes	33	33333	3200

Part II Reading

We introduced the software product quality assurance activities for conformance to the requirement specification and user needs, including evaluating software products for conformance, reviews (inspections and walkthroughs), various software testing techniques, assuring software quality conformance for operation services, developing software product quality metrics, and defining procedures and work instructions for product quality assurance activities.

In this part of the group discussion, each group is required to complete the following.

1. Read through the textbook chapters 12 through 17.

Summary of Concepts Learned from Chapters 12-17:

Chapters 12-17 cover various aspects of software quality assurance (SQA), focusing on evaluation, review methodologies, testing strategies, operational services, quality metrics, and procedures:

1. Software Product Evaluation (Chapter 12):

- Types of evaluation activities include assessing project plans, evaluating project products, determining customer acceptability, and using measurement.

- Processes like conformance evaluation and measurement-based evaluation are outlined, emphasizing the identification of nonconformities and the iterative nature of improvement.

2. Review Methodologies and Testing Strategies (Chapters 13-14):

- Review methodologies aim to identify errors, detect risks, and support process improvement.

- Various review methods, such as formal design reviews, inspections, and walkthroughs, are compared based on their objectives, participants, and contributions.

- Testing strategies encompass incremental and big bang approaches, along with types like black box and white box testing.

- Requirement-driven testing focuses on factors such as operation, revision, and transition, with planning involving test case sources, performers, and location considerations.

3. Software Operation Services and Managerial SQA Components (Chapters 15-16):

- Software operation services include user support and maintenance activities like corrective, adaptive, and perfective maintenance.

- High-quality foundations involve ensuring quality software packages, version release policies, and implementing infrastructure QA processes.

- Managerial SQA components cover contract review, operational planning, progress control, metrics, and quality cost management.

4. Software Quality Metrics and Procedures (Chapters 16-17):

- Objectives of software quality metrics include supporting management control, observing conformance, and providing data for process improvement.

- Successful metrics possess characteristics such as relevance, validity, reliability, and ease of implementation.

- Procedures in SQA aim to ensure conformity, uniformity, and improved communication, distinguishing between procedures (universal) and work instructions (team-specific).

These chapters collectively provide understanding of the principles, methodologies, and practices essential for effective software quality assurance in the development lifecycle.

2. Select one software product quality assurance activity.

   In the papers mentioned, one software product quality assurance activity is “test case prioritization and selection”.

   The selection of test case prioritization and selection as a software product quality assurance activity is derived from the topics covered in **Chapter 14**, specifically under the discussion of testing strategies and requirement-driven testing. In Chapter 14, various aspects of testing are discussed, including the planning, execution, and management of test cases. Test case prioritization and selection are crucial activities within the broader context of software testing, as they directly impact the effectiveness and efficiency of the testing process in ensuring software quality.

   one software product quality assurance activity is test case prioritization and selection.

3. Search for at least two recent publications on the software product quality assurance activity you selected.

   Paper 1:

   Generating Test Input with Deep Reinforcement Learning

   Link: https://ieeexplore.ieee.org/document/8452812

Details: "Generating Test Input with Deep Reinforcement Learning," the focus is on using deep reinforcement learning to generate test inputs. This process involves prioritizing and selecting specific test cases or inputs that are most likely to uncover bugs or vulnerabilities in the software.

   Paper 2:

   Reinforcement Learning for Automatic Test Case Prioritization and Selection in Continuous Integration

Link:
https://www.computer.org/csdl/journal/ts/2022/08/09394799/1stroHW2Ums

Details: "Reinforcement Learning for Automatic Test Case Prioritization and Selection in Continuous Integration," specifically addresses the use of reinforcement learning for prioritizing and selecting test cases within the continuous integration process. This activity ensures that the most critical test cases are executed promptly, thus improving the efficiency and effectiveness of the testing process, ultimately enhancing software quality assurance.

4. Read and discuss the articles.

Paper 1: Generating Test Input with Deep Reinforcement Learning:

This paper likely explores the application of deep reinforcement learning techniques to generate test inputs for software testing purposes.

Deep reinforcement learning involves training agents to take actions in an environment to maximize some notion of cumulative reward, typically through trial and error.

In the context of software testing, this could mean training an agent to generate inputs (such as test cases) for a program in a way that maximizes the detection of bugs or vulnerabilities.

The paper may discuss how deep reinforcement learning algorithms are applied to this problem, the challenges involved, and the potential benefits compared to traditional methods of test input generation.

It could also delve into experimental results demonstrating the effectiveness of the proposed approach in generating diverse and effective test inputs.

Paper 2: Reinforcement Learning for Automatic Test Case Prioritization and Selection in Continuous Integration:

This paper likely focuses on leveraging reinforcement learning techniques for the automatic prioritization and selection of test cases within the context of continuous integration (CI).

Continuous integration involves frequently integrating code changes into a shared repository and running automated tests to detect integration errors early in the development process.

Test case prioritization is crucial in CI environments to ensure that limited testing resources are allocated efficiently, prioritizing critical test cases that are most likely to uncover defects.

The paper may discuss how reinforcement learning algorithms are used to learn optimal test case prioritization policies based on historical data, feedback from previous test runs, or other relevant metrics.

It may also explore the challenges associated with test case prioritization in CI, such as dealing with dynamically changing codebases and evolving testing requirements, and how reinforcement learning can address these challenges.

Experimental results demonstrating the effectiveness of the proposed approach in improving testing efficiency and quality may be presented and discussed.

These papers likely contribute to advancing the field of software testing by introducing novel techniques and methodologies that leverage reinforcement learning for test input generation and test case prioritization in CI environments. They could offer valuable insights into how machine learning approaches can be applied to address challenges in software testing and enhance testing practices.

5. Summarize your readings as a reading report.

Answer:

Key takeaway from Paper 1:

- The goal of the paper is to replace typical manually built metaheuristic algorithms with an inventive application of reinforcement learning (RL) in Search-based Software Testing (SBST) to automate test data creation.

- It employs RL in SBST by drawing a comparison between the decision-making process of RL and metaheuristic trial-and-error procedures.

- Applying RL algorithms is made possible by the advent of the GunPowder framework, which extends software under test into an RL-interactable environment.

- In order to train RL agents, the study uses Double Deep Q-Networks (DDQN), which show promise in learning efficient test data generation techniques and attaining 100% branch coverage in empirical assessments.

- These results demonstrate the future promise of combining RL and deep neural networks with SBST to improve test data generating efficacy and efficiency.

Key takeaway from Paper 2:

- Motivated by Reinforcement Learning's (RL) effectiveness in adaptive contexts such as gaming and real-time bidding, this work investigates RL's application to the dynamic problems and timing restrictions in Continuous Integration (CI) environments.

- By presenting test case prioritizing as a ranking problem, it presents a novel technique that makes it easier to apply RL solutions by modeling the sequential interactions between a test case prioritization agent and the CI environment.

- To make sure the answers are perfectly suited to the CI situation, this research uses three different ranking models to conduct a thorough examination using state-of-the-art, customized reinforcement learning approaches.

- The flexibility of the suggested approach to change the prioritization strategy automatically and continuously is a crucial component that has demonstrated considerable accuracy gains over earlier techniques.
- The encouraging outcomes show that RL can successfully approach an ideal test case prioritization strategy, which represents a major breakthrough in the area and opens the door for RL approaches to be more widely used in CI scenarios.