OOSE 2 Learning Outcomes	Exam Question
Understand different requirements and software modelling techniques.	1,2
Apply design principles and patterns while designing and implementing simple systems, based on reusable technology.	2, 3, 4, 5, 6, 7
Use simple software architectures in engineering interaction between software components.	4, 5, 6, 7
Identify different coding defects and how they can be identified and managed.	8, 9, 10
Understand different approaches to software testing.	8, 9, 10
Apply a range of measurement and analysis techniques to improve quality of software.	9?
Discuss the breadth of software engineering.	

Question 1:

Select two of the following relationships for which composition would be a more appropriate way to model the relationship than aggregation?

A composition relationship is a *part-whole* relationship in which the *part* cannot exist without the *whole*. In other words, if the *whole* disappears then the *part* disappears also.

There were 2 versions of this question. Correct examples in the question were:

Person (whole) - Brain (part)

Calendar (whole) - Day (part)

Country (whole) - Region (part)

Football Stadium (whole) - Football Pitch (part)

Many students got at least 50% of this question correct and a good number of students achieved full marks for the question. However, some people selected more than two answers for the question so inevitably lost marks due to negative marking. The mean attainment for this question was 38% correct.

Question 2:

Which of the relationships listed below are examples of inheritance relationships that are appropriate for modelling as Super Class - Sub Class? **Select all that apply**.

This question is about the principles of hierarchy. The super class – sub class design principle models an *is-a* relationship. In other words, the sub class *is an* instance of the super class.

There were 2 versions of this question. Each version had two correct answers:

Plant – Tree: a tree is a plant.

Computer – Laptop: A laptop is a computer.

Festival – Christmas: Christmas is a festival.

Fruit – Apple: an apple is a fruit.

In general, students performed very well on this question. More than a third of the class achieved full marks and wore than 80% of the class got at least half marks for this question. The mean attainment for this question was 81% correct. Again, selecting too many answers was a common reason for losing marks on this question.

Question 3:

Identify all of the violations of Liskov's Substitution Principle (LSP).

The violations of LSP were in how the calculateInterest method was implemented in the CurrentAccount, SavingsAccount and CorporateAccount classes. The violations were examples of strengthening the pre-conditions (CurrentAccount.calculateInterest), weakening the post-conditions (SavingsAccount.calculateInterest), and not adhering to the contract of the base class (CorporateAccount.calculateInterest). There were many possible ways to refactor the classes so that they adhered to LSP and maintained the functionality of the code. Any reasonable suggested refactorings were accepted.

In general, students did quite well on this question. Over 65% of the class scored 50% or more and ~25% of the class scored 75% or more on this question. The mean attainment for this question was ~55% correct.

Question 4:

This question is about design patterns. The correct answer was ‘Observer design pattern because objects that interact are loosely’.

In general, students performed very well on this question. More than three quarters answered this question correctly.

Question 5:

This question is about design patterns, and it had two parts. The first part requires to pick the appropriate design pattern, which is the strategy design pattern, and second part requires to pick the key principles behind the strategy design pattern:
a) Add new functionality without altering the main class, b) Add new functionality without altering the subclasses and c) It is able to inherit properties from the superclass and thus avoid code duplication.

In general, students performed well on this question. Around 40% of the students scored more than 70%.

Question 6:

This question is about design patterns and the correct answer is the decorator design pattern because it attaches additional responsibilities to an object dynamically. Most students answered this question correctly.

Question 7:

This question is about design patterns and the correct answer is the observer design pattern because it enables asynchronous communication. This question has been answered correctly from most students.

Question 8:

This question is about software testing. The correct answers are:
a)

The Statement coverage is 10 out of total 14 statements.
The Branch coverage is 3 out of total 6 conditional branches.

The Condition coverage is 0 out of total 3 conditions.

b)

The Statement coverage is 11 out of total 14 statements.
The Branch coverage is 4 out of total 6 conditional branches.

The Condition coverage is 1 out of total 3 conditions.

Most of the students scored above 50%. However, several students have not understood well the difference between branch and condition coverage.

Question 8:

This question is about software testing. Refactoring was required for the lines below:
public class TestObserverPatternExample{

- BeforeAll void init() throws Exception{
- @Test (timeout=DEFAULT_TIMEOUT)
- public void testFacebookClientAddMessage(){
- public void testwhatsAppClientAddMessage(){
- Assert.assertEquals(”Message - sent from WhatsApp”,subject.getState());

Around 67% of the students scored more than 65% of total marks.

Question 9:

This question is about software testing. The correct answer is that ‘assertSame would fail because ‘assertSame’ by default it compares the references of the two objects. ‘assertEqual’ would fail because it behaves similarly to assertSame for class objects.

Around 60% of the students scored above 70% in this question.