TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI

VICTORIA UNIVERSITY OF WELLINGTON

EXAMINATIONS – 2017

TRIMESTER 2

SWEN222

SOFTWARE DESIGN

Time Allowed: TWO HOURS

CLOSED BOOK

Permitted materials: No calculators permitted.
Non-electronic Foreign language to English dictionaries are allowed.

Instructions: Answer all questions

Answer all questions in the boxes provided.
Every box requires an answer.
If additional space is required you may use a separate answer booklet.

Question	Topic	Marks	
1.	Design Patterns	30	
2.	Composite and Visitor Pattern	30	
3.	Testing	30	
4.	Functional Design	30	
	Total	120	

Question 1. Design patterns

[30 marks]

(a) [5 marks] What is a Design Pattern?

(b) [5 marks] How did Design Patterns get discovered and categorised? What kind of terminology is used?

(c) [5 marks] Compare a group of programmers that have good knowledge of Design Patterns and related terminology with a group of programmers that lack this knowledge.

What advantages has the first group over the second one, and why?

(d) [5 marks] What is the opposite of a good Design Pattern? Why is it important to categorize those too?

(e) [5 marks] Write the names of 8 different Design Patterns

(f) [5 marks] Explain with concrete examples how Design Patterns help create reusable library code instead of code with just a fixed set of behaviours.

Question 2. Composite and Visitor Pattern [30 marks]

(a) [5 marks] Explain the Composite Pattern. Feel free to use a class diagram if you think it will help your explanation.

(b) [2 marks] Name a famous library (that we used a lot in SWEN222) that uses the Composite Pattern.

(c) [5 marks] Using exactly one interface and two classes implementing such interface, provide an example of Composite Pattern. Here I want you to show the code structure, not a realistic example inspired on a real world scenario. Thus use abstract names and minimal code. Call your interface I and the two classes A and B. If methods or fields are needed, name them m1, m2, f1, f2 and so on.

(d) [5 marks] [HARD] When using the Composite Pattern, it may be desirable to enforce that the shape of the object graph is a tree. Rewrite here a modified version of your 2(c) answer; this modified code should guarantee that all graphs of composites are trees.

(e) [3 marks] How do you insert new operations in the Composite Pattern?

Why is this inconvenient when there are many operations?

(f) [5 marks]

Rewrite and adapt the code of Question2 point(c) (interface I and classes A,B) to use the Visitor Pattern.

(g) [5 marks]

Building on the example before (interface I and classes A,B) provide an implementation for a CloneVisitor, and one for a ToStringVisitor .

Question 3. Testing

[30 marks]

(a) [5 marks] Explain the difference between Manual Testing and Automated Testing.

What is the benefit of Automated testing over Manual Testing?

(b) [5 marks] Explain the difference between Unit tests and Integration tests.

(c) [5 marks] Explain what it means for two different units of code to be independently testable.

(d) [5 marks] Describe Mocking with a simple example.

(Do not put code now, we will ask for code later in this question)

(e) [5 marks] Show a minimal example of code, where the Design Pattern "Abstract Factory" is used to allow Mocking.

(f) [5 marks]

Contracts: Preconditions, Postconditions and class Invariants can aid testing: Explain why using assert to verify contracts in the code greatly increases the likelihood that a test can discover a programming error.

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Question 4. Functional Design

[30 marks]

(a) [5 marks] Behaviour of ‘ assert ’ statements and ‘Futures’ should be “pure”.

Describe what we mean by “pure” behaviour.

(b) [5 marks] Behaviour of ‘ assert ’ statements and ‘Futures’ should be “pure”.

Why is this especially important for assert statements and Futures? What issues we can have if we do not use pure behaviour in those cases?

(c) [5 marks] Explain why a library offering a functional observable interface can be easier to use and harder to misuse.

(d) [5 marks]

"instances of the child class must be a valid substitute for instances of the super class"

Explain what this sentence means in the context of method pre and post conditions.

(e) [10 marks]

[Hard] Write an example implementation for a Functional list/stack of `int` elements.

Be sure to include operations to add and remove elements, as well as a way to obtain an empty list.

When you use a Pattern in your code, put a comment with the name of such Pattern.

* * * * * * * * * * * * * * *

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.