

END-YEAR

COMP 462

OBJECT-ORIENTED PARADIGMS

Time Allowed: 3 Hours

Instructions:

- *Read each question carefully before attempting it.*
- This examination will be marked out of **120** marks.
- You may answer the questions in any order. Make sure you clearly identify the question you are answering.
- Many of the questions require you to express and justify an opinion. For such questions, you will be assessed on your *justification*.
- Some of the questions ask for examples from object-oriented languages. Your answers need only refer to object-oriented languages discussed in the course, but you may refer to other languages if you wish.
- Non-electronic foreign language-english dictionaries are permitted.

Question 1. Object-oriented programming

[30 marks]

The statement below (or something like it) was used several times in the course.

An object-oriented program is a set of objects that interact by sending messages to each other to provide functionality.

Explain each of the following concepts in terms of the above quote: Class, Encapsulation, Polymorphism, Inheritance, Types, Identity. Your explanation may include examples from the various object-oriented programming languages surveyed in the course.

Question 2. Domain Analysis

[10 marks]

- (a). Briefly describe the main phases in performing a textual analysis.
- (b). Could any of the stages be omitted — why or why not?

Question 3. Use Cases

[10 marks]

Below is a first draft of an essential use case for an automatic ticket kiosk at a cinema.

- (a). List three things wrong with this use case.
- (b). Write a better version.

<u>user intention</u>	<u>system responsibility</u>
	display welcome screen. play happy music
identify self swipe credit card in card reader	verify identification check credit card number
offer movie choices	choose movie
chose number of tickets	offer number of tickets debit account
collect tickets walk away	
see movie	show movie

Question 4. CRC Cards

[10 marks]

- (a). Sketch one example of a CRC Card, labelling your sketch to describe each part of your card.
- (b). Describe one main advantage of CRC Cards.

Question 5. Heuristics and Metrics

[10 marks]

- (a). Give one example of an object-oriented design heuristic.
- (b). Give one example of an object-oriented design metric.
- (c). Describe a design situation where using metrics would be better than using heuristics.

Question 6. Inheritance

[10 marks]

Java does not have multiple inheritance, but it does have the `interface` mechanism. Explain how this mechanism can be used to provide one of the benefits of multiple inheritance.

Question 7. Design Patterns

[10 marks]

Design Patterns are often described as a *solution to a problem in a context*. Explain what this means, paying particular attention to the emphasised words.

Question 8. Object-oriented Frameworks

[10 marks]

Imagine you are designing a tournament management framework, and your framework must support at least:

- tennis knockout tournaments — the tournament tree can be represented by binary trees of matches.
- soccer leagues — every team plays every other team.
- New Zealand National Provincial Rugby — every team plays every other team (like a soccer league) then the top four teams play a knockout tournament (like tennis).

- (a). Draw object diagrams for examples of the three types of tournaments.
- (b). Draw a class diagram that uses the composite pattern to represent all three object diagrams.

Question 9. Extreme Programming (XP)

[10 marks]

The 12 practices of XP are: Planning Game, Frequent Releases, System Metaphor, Simple Design, Testing, Refactor Mercilessly, Pair Programming, Collective Code Ownership, Continuous Integration, Forty Hour Week, Onsite Customer, Coding Standards.

Choose **two** XP Practices. For each practice:

- (a). Briefly describe what it involves — what people have to do.
- (b). Explain how it contributes to the goals of XP.

Question 10. Alternative Object Models

[10 marks]

Explain, with examples, what a “prototype”-based language (such as Self) is, and how it differs from a class-based language.
