Family Name _____

First Name _____

Student Number _____

Venue _____

Seat Number _____

UNIVERSITY OF CANTERBURY
*Te Whare Wānanga o Waitaha*
CHRISTCHURCH NEW ZEALAND

**No form of communication or collaboration allowed during exam.**

## Computer Science and Software Engineering

## EXAMINATION

Mid-year Examinations, 2021

# SENG301-21S1 (C) Software Engineering II

**For Examiner Use Only**

| Question | Mark |
|---|---|
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

Total _____

**Examination Duration:** 180 minutes

**Exam Conditions:**

Open Book exam: Students may bring in any written or printed materials.

No calculator permitted.

**Materials Permitted in the Exam Venue:**

Open Book exam: Students may bring in any written or printed materials.

**Materials to be Supplied to Students:**

1 x Standard 8-page UC answer book.

String to tie exam materials together.

**Instructions to Students**

Answer all questions. Please write in a readable and intelligible manner, **unreadable answers will be ignored**.

Please turn the pages over and ensure you have seen the entire exam paper.

Read the entire question before answering any part of it. **A few questions have associated figures** or notes that are crucial to the question.

Some questions require you to describe, explain or ellaborate on some concepts. You are expected to write **complete sentences**. No answers made of bullet points will be accepted.

Check carefully the **number of marks allotted to each question**. This suggests the degree of detail required in each answer, and the amount of time you should spend on the question.

**No form of collaboration is permitted**

**Questions Start on Page 3**

# Question 1 Agile Software Development

15 marks for whole question

### Question 1.1 Scrum values

10 marks

List and discuss the five Scrum values. When discussing **each value**:

- explain why this value is important within a software development context
- explain which principle of the agile manifesto this value relates to

### Question 1.2 Retrospective

2 marks

What is a retrospective meeting in Scrum? Explain why it is important within the Scrum framework?

### Question 1.3 Action items

3 marks

As part of the retrospective, action items are created. What framework is usually used to design them?

- explain this framework
- give an example of a "good" action item

# Question 2 Continuous integration and deployment

10 marks for whole question

Please read below excerpt from a transcript of an interview at *PukekoDev* where they explain part of their continuous development strategy.

> "*PukekoDev* is a small software development company developing a web-based tool for site surveyor. We work closely with surveyors and the new features are driven by them, represented by one of our team lead who is a former surveyor herself.
>
> We are using a decentralised version control system. We use a sprint backlog to keep track of our work to do for the current sprint that last one week. For every task, developers create a new branch that they pull from the main branch where all our latest development is going on. They start working on their own to implement the task. When finished, they pull the latest changes locally, resolve any conflicts and push their changes to the main branch.
>
> When pushing the changes, the integration server runs the unit tests and deploy the latest successful build on the quality acceptance server. That server is used during our *"Let's Break it or Deploy Friday"* (LeBiDeF) sessions where the whole development team tests the system manually. Because we are a small team, we do not follow any predefined plan since everyone knows about their own features developed during that sprint.
>
> During a LeBiDeF session, any bug that is discovered is live-debugged and fixes are applied on the main branch as they are discovered. At 3PM, the teams gathers for a last huddle and take the decision to create a new release and deploy on the production server or not. If the team is not confident enough, we pre-fill the next spring backlog with the list of bugs that need to be fixed prior new development must be done so that we are informed before going into sprint planning. If the team agrees on a go, the system operators deploy the new release and update the deployment notes, if any changes are needed (e.g., configuration changes, external dependencies to install)."

This strategy has at least **5** process flaws that would need improvements. These flaws are either technical (*i.e.* the way the source code is managed) or methodological (*i.e.* the way the team works). Identify them and for each flaw, give **actual advice on how to improve the current strategy**. Focus your answer on the flaws and explain your solution on how to fix them, but **do not re-explain a git flow**.

# Question 3 Code quality and reviews

15 marks for whole question

Listing 1 reproduces some sample code. Take a close look at it before answering the next questions.

```
1  /**
2   * Generic method to replace an element from a collection with another
3   *
4   * @param elements   a collection of items
5   * @param oldElement the element to be replaced
6   * @param newElement the new element to substitute the oldElement with
```

```java
   * @return element removed from list
   */
  public <T> T updateMe(List<T> elements, T oldElement, T newElement) {
    if (null != oldElement && null != newElement) {
      ;
    } else {
      return null;
    }

    int currentIdx = 0;
    for (T element : elements) {
      if (element.equals(oldElement)) {
        elements.add(currentIdx, newElement);
        elements.remove(currentIdx + 1);
      }
      currentIdx += 1;
    }
    return oldElement;
  }
```

Listing 1: Generic update method (see Question 3)

**Question 3.1 Unit tests** 5 marks

Specify the unit tests you would write for this method. You can express the tests either in plain English or in a mathematical form.

**Question 3.2 Code review** 10 marks

Write a review for this piece of code as you would do when assessing a merge request. Assume the task associated to this merge request was to create this reusable method to update any sorts of list of elements and therefore, this merge request is composed by this method only (with its docstring).

## Question 4 Weekly readings and additional material 10 marks

### Question 4.1 Test-Driven Development

**Question 4.1.1** 1 mark

In his video *"Five underplayed premises of TDD"*, GeePaw Hill says that *"We're in this for the money"*. What does *"this"* refer to?

**Question 4.1.2** 4 marks

From other parts of the course material, we learned how testing, tasking and coding are all intertwinned, something GeePaw explains using the testability angle. Can you ellaborate on that aspect, i.e. how the testability principle influences all three activities (tasking, coding and testing)?

### Question 4.2 Basket of options and option on a basket

**Question 4.2.1** 1 mark

Explain the metaphor used by Kent Beck in his blog post *"Decisions, Decisions or Why Baskets of Options Dominate"*.

**Question 4.2.2** 2 marks

Explain how this may influence the planning activity in Scrum?

**Question 4.2.3** 2 marks

Critically discuss why project roadmaps may be inevitable and how you can work to minimise their drawbacks?

## Question 5 Design patterns

11 marks for whole question

**Question 5.1** 1 mark

Two of the GoF patterns **covered in lectures** focus on the encapsulation of algorithms. Name them.

**Question 5.2** 4 marks

In your own words, clearly and concisely explain *how* each pattern encapsulates algorithms.

**Question 5.3** 4 marks

A financial application includes a class which models a client's portfolio of investments. The class needs to have a method which calculates the amount of tax that should be paid on a portfolio. The algorithm for calculating tax depends on the country where the client lives. In your own words, clearly and concisely explain how you would decide which of the two patterns, identified in question Question 5.1, you would recommend using to implement the tax calculation method. You are not required to write any code, but may sketch examples if relevant: your answer should focus on the relevant criteria and the reasons for your recommendation.

**Question 5.4** 2 marks

Would it ever be appropriate to use both patterns together? Briefly explain your answer.

## Question 6 Design principles

6 marks for whole question

**Question 6.1** 2 marks

In your own words, clearly and concisely describe the open-closed principle and briefly explain how applying it can benefit software engineers.

**Question 6.2** 4 marks

The open-closed principle is associated with a number of the GoF design patterns. Name one such pattern and explain how the open-closed principle applies.

## Question 7 Software Design Improvements

14 marks for whole question

Ron is writing a game in which characters have the ability to eat food in order to maintain their energy levels. Characters gain energy in various ways, such as eating food, and lose it when they exert themselves during the game. The class diagram of Figure 1 shows how he has modelled various kinds of food.

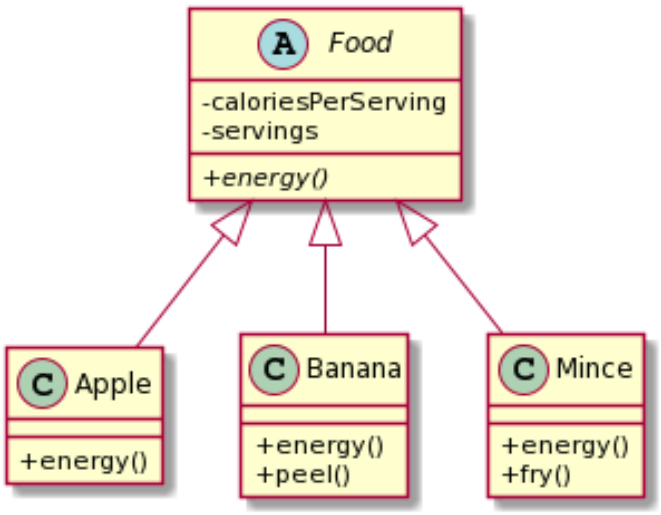


Figure 1: Ron's class diagram (see Question 7)

He has written the corresponding Java classes, together with a `Character` class which contains the following `eat()` method. The `eat()` method is shown in Listing 2. `Character` also has a property, `energy`, and a method, `updateEnergy()` that adds or removes energy.

When Hermione sees what Ron has done, she is unimpressed. He tells her "I've done everything right! I've used inheritance to model the different kinds of food and I'll add even more kinds soon. In the `eat()` method, I use `instanceof` to check what kind of food each object is. If necessary, I cast the object to the appropriate type so I know it actually has the method I'm going to call." Hermione replies "You may be using inheritance, but you're not using polymorphism — they're not the same thing! No wonder your `eat()` method is so ugly."

```
public void eat(List<Food> meal) {
  for (Food f : meal) {
    if (f instanceof Banana) {
      ((Banana) f).peel();
    } else {
      if (f instanceof Mince) {
        ((Mince) f).fry();
      }
    }
    updateEnergy(f.energy());
  }
}
```

Listing 2: Ron's `eat()` method (see Question 7)

**Question 7.1** 4 marks

Do you agree with Hermione? She's usually right... Clearly and concisely explain the main problems, if any, that Ron's design has.

**Question 7.2** 4 marks

How would you improve Ron's design? Briefly explain any decisions you have had to make. If you propose any changes to the class diagram then sketch the updated diagram. If you propose any changes to the `eat()` method then sketch them also.

**Question 7.3** 2 marks

Ron wants to remove each food from the meal as it is consumed. He tried inserting the statement `meal.remove(f);` after the call to `updateEnergy()`. Will this work? If so, explain why; if not, explain why not and suggest an alternative way to achieve Ron's goal.

**Question 7.4** 4 marks

Hermione also tells Ron that he should have specified a contract for the `eat()` method before he wrote its code. In your own words, clearly and concisely describe what is meant by the contract of a method. How are contracts expressed?

## Question 8 Sofware design

16 marks for whole question

Here is a description of part of the design of an aquarium simulation.

> There will be lots of different kinds of fish, but only ever one tank. Fish spawn new fish. Sometimes fish form schools, and all move around together. Individual fish join or leave a school whenever they wish. Some fish have remoras, sea lice or other parasites on them. Our monitoring and computer vision system supports several views of the aquarium — these show in various ways what is happening in the aquarium.

**Question 8.1** 10 marks

Consider which GoF design patterns would be most suitable to help design the system. Sketch a UML class diagram to show your design. Your diagram need not be formally complete in every detail, but should include sufficient information to show your ideas. You may include brief explanations of any significant assumptions or decisions you have made.

**Question 8.2** 6 marks

Document, in the format covered in class, two of the patterns you have used. If you haven't used two then you have missed some.

## Question 9 Design patterns as haiku

3 marks for whole question

A *haiku* is a short poetry form that originated in Japan. Though brief, they can express complex and deep content succinctly. A *haiku* consists of 17 syllables in a three-line form as shown in Figure 2(a).

| | |
|---|---|
| First five syllables<br>Second, a line with seven<br>Then another five | Secret algorithm<br>Strategy encapsulates<br>I can change my mind |
| (a) The form | (b) Paddy's *haiku* |

Figure 2: Example *haiku* (see Question 9)

Choose a GoF design pattern that we have covered in class and write an original *haiku* that describes it. Figure 2(b) shows an example you may remember from class. Your answer should make it clear which pattern is being referred to, and should include key elements of the pattern's intent, participants, applications etc. You may provide a brief explanation if you wish.