Family Name	____________________
First Name	____________________
Student Number	|__|__|__|__|__|__|__|__|__|
Venue	___________________
Seat Number	________

UNIVERSITY OF CANTERBURY
Te Whare Wānanga o Waitaha
CHRISTCHURCH NEW ZEALAND

No electronic/communication devices are permitted.

No exam materials may be removed from the exam room.

Computer Science and Software Engineering

EXAMINATION

Mid-year Examinations, 2019

SENG301-19S1 (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.

Any scientific/graphics/basic calculator is 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 16-page UC answer book

String to tie exam materials together.

This exam has 1 attachment

Instructions to Students:

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

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.

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.

Enlarged figures are provided in the Supplementary Material section at the end of the exam paper.

Questions Start on Page 3

1. **[10 marks for whole question]** *Requirement engineering*

   We discussed two ways of identifying users during early requirement engineering stages.

   a. **[5 marks]** What are these two types and how can you specify them?

   b. **[5 marks]** What is the main difference between these two types? Use an example for each of them to illustrate your argumentation.

2. **[10 marks for whole question]** *Agile Software Development*

   a. **[2 marks]** What does the "S" mean in Bill Wake's INVEST acronym in terms of the scope of a story?

   b. **[2 marks]** Why the SCRUM framework insists so much on cross-functional team members?

   c. **[2 marks]** What is the alternative release burndown chart?

   d. **[2 marks]** What is the main objective of a sprint retrospective in SCRUM?

   e. **[2 marks]** What is the key difference between KANBAN and SCRUM in terms of release management?

3. **[15 marks for whole question]** *Continuous integration and deployment*

   a. **[9 marks]** "Our project is using a decentralised source version control, such as GIT, where developers push their code when they finish their tasks. We use a branching and merging strategy where every feature has its own branch. After each task, the full set of unit tests is run against that particular branch to ensure the pushed code does not break the current development. Close to the end of each sprint, we freeze the development and merge all ongoing features that are stable into the main branch containing the production code. Every branch is merged one at a time and potential conflicts are fixed straight after the merging if necessary. When all branches are merged, the testing team goes through the delivered stories and check all acceptance criteria for all stories. When all stories pass, the main branch can be deployed on production by the deployment team that will conduct smoke tests after the deployment."
   This branching and merging strategy suffers from some flaws. Identify at least four (4) flaws and for each flaw, give actual advice to improve this flow. You have to preserve the feature-branching strategy, i.e. you are not allowed to explain a story branching for example. Focus your answer on the flaws and give solutions on how to improve it, do not re-explain a git flow.

   b. **[3 marks]** When merging a piece of code into the mainline of a codebase, what is the most important thing to review, according to you? Briefly justify why in two or three sentences.

   c. **[3 marks]** Why file-based configurations should be preferred to GUI-based configurations? Give one concrete example to support your argumentation.

4. **[10 marks for whole question]** *Software quality*

   a. **[4 marks]** Define the unit tests for the following Java method. You can express them in a mathematical way, in English or using java-like Junit syntax. What quality problem would you identify from this method implementation?

```
public int[] reverse(int[] array) {
  int temp, start = 0, end = array.length-1;
  int[] copy = array.clone();
  while (start < end) {
    temp = array[start];
    copy[start++] = array[end];
    copy[end--] = temp;
  }
  return copy;
}
```

   b. **[4 marks]** Define the unit tests for the following Java method. You can express them in a mathematical way, in English or using java-like Junit syntax. What restriction do you see in this implementation and what would you propose to tackle it?

```
public int times(int a, int b) {
  int result = a;
  for (int i = 2; i <= b;i++) {
    result += a;
  }
  return result;
}
```

   c. **[2 marks]** In terms of code quality, both samples are missing an important feature. What is it? Why is it important?

5. **[5 marks for whole question]** *Resilience Engineering*

   a. **[3 marks]** What are the three main types of strategies that can be put in place to deal with security threats? Explain each of them with two or three sentences.

   b. **[2 marks]** What is the availability of a system? How does it relate to the reliability?

6. **[4 marks for whole question]** *Design: Encapsulation*

   a. **[1 marks]** What is encapsulation leak?

   b. **[2 marks]** Using an example, briefly describe how encapsulation leak could occur even with private attributes.

   c. **[1 marks]** Explain the effect encapsulation leak has on data integrity.

7. **[5 marks for whole question]** *Design: Visibility*

   The primary function of accessors and modifiers (otherwise known as getters and setters) is to get and set the value of attributes. *"Always use accessors/modifiers"* is one side of the privacy continuum debate. Using an example and design principles, explain this side of the argument.

8. **[6 marks for whole question]** *Design: Agent classes*

   "Model[ling] the real world" often produces agent classes in software designs.

   a. **[3 marks for question]** Draw a class diagram of an example to explain what this means. From your explanation, the reader should be able to understand what agent classes are.

   b. **[3 marks for question]** List two cons to using agent classes; use your example to explain the cons.

   Note that agent classes do not refer to intelligent/adaptive/autonomous agents or bots.

9. **[5 marks for whole question]** *Design: Principles*

   Using a small example, briefly explain constructor injection. What is the primary principle that constructor injection tries to uphold?

These notes apply to questions 10 and 11 below

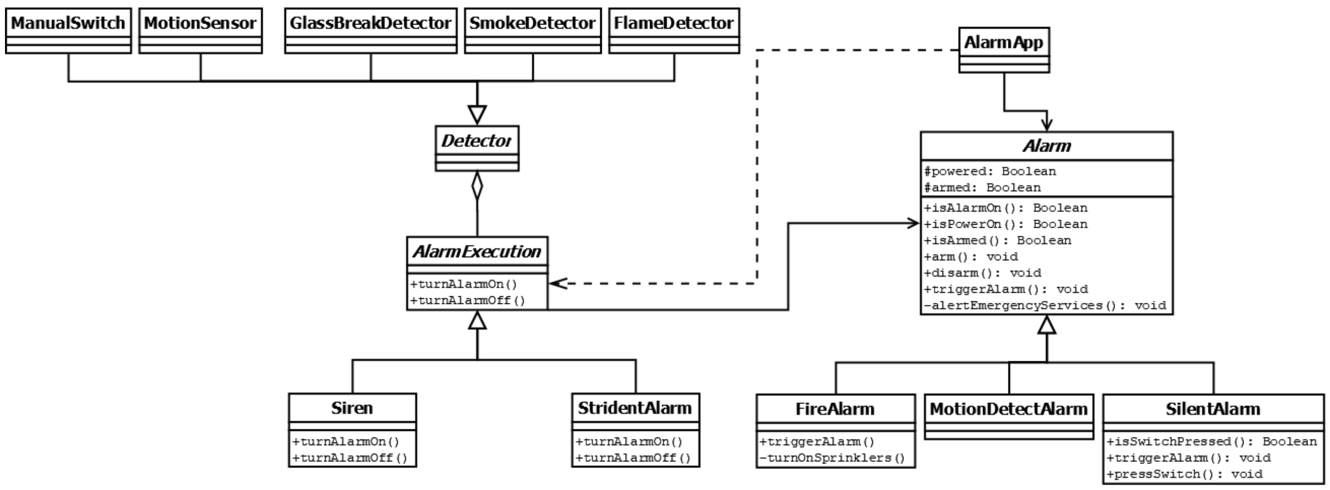


Figure 1: Partial class diagram for Alarm System

Figure 1 shows part of the class diagram for an alarm system; note that functions or datatypes that are not relevant to the question are not shown. An enlarged version of this diagram can be found in the supplementary section at the end of this exam paper.

For an `Alarm` to work, the power has to be on, the alarm has to also be switched on, and the alarm should be armed. If these are done before the alarm is triggered, an audible alarm will sound and the emergency services will be alerted.

There are three types of alarms: a `FireAlarm`, `MotionDetectAlarm`, and a `SilentAlarm`. There are also different types of detectors that are used to trigger the alarm: a `ManualSwitch`, `MotionSensor`, `GlassBreakDetector`, `SmokeDetector`, and `FlameDetector`. A `FireAlarm` is guaranteed to always be powered, turned on, and armed. It responds to smoke, heat, and flames, detected by the respective detectors. A `MotionDetectAlarm` is triggered when motion is detected using the `MotionSensors` and `GlassBreakDetectors`. A `SilentAlarm` is used in banks and is always powered and turned on; there is no need to arm this alarm. Triggering it requires the teller to press a switch located under their desk. The alarm is not audible but, as with the other alarms, the emergency services are alerted.

Each alarm makes different sounds when triggered. The `FireAlarm` lets out a strident (high pitch) sound, while the `MotionDetectAlarm` lets out a siren.

10. **[10 marks for whole question]** *Design by Contract*

Make sure you have read the notes above.

Describe whether the Alarm class hierarchy shows good or bad inheritance. To answer this question, document the `triggerAlarm()` method's contracts in a table (as learnt in class) showing pre-conditions and post-conditions. State whether the pre- and post-conditions have been loosened, tightened, or have remained the same, and whether that is "OK" or "Not OK" in terms of DBC. Finally state whether each class is "OK" or "Not OK" giving a short explanation. Note any assumptions you have made.

11. **[20 marks for whole question]** *Design Patterns*

Make sure you have read the notes above.

a. **[10 marks for question]** Identify and document **one** GoF design pattern that you learnt in class, already contained in this design.

When documenting the pattern, provide a table (format it as you have learnt in class) that shows how every feature in the standard pattern (class names, attribute names, method names, and relationships) corresponds to a feature (or features) in this design. Few marks will be given for naming patterns. The majority of marks will be allocated for correctly describing how the pattern is applied. If a GoF concept is omitted in this application of the pattern, say so in the table. If a feature has no name, describe it, e.g. "the relationship between class A and class B." Do not discuss the intent of the patterns; just show where they are used. Do not document more than one pattern; if you do, only the first one will be marked.

b. **[10 marks for question]** You have been asked to add to this design by providing a facility for sounds to sometimes be at a higher volume. This is so that they can be used in retirement homes where many of the residents have partial hearing loss.

Using a GoF design pattern, add this functionality to the current design. To do this, **state the design pattern**, **draw the section of the class diagram** showing this new functionality, and **document the pattern**. In this question, the switch itself is irrelevant; it may either be a physical switch or a programmable interface.

Document the pattern as you were asked to in (a).

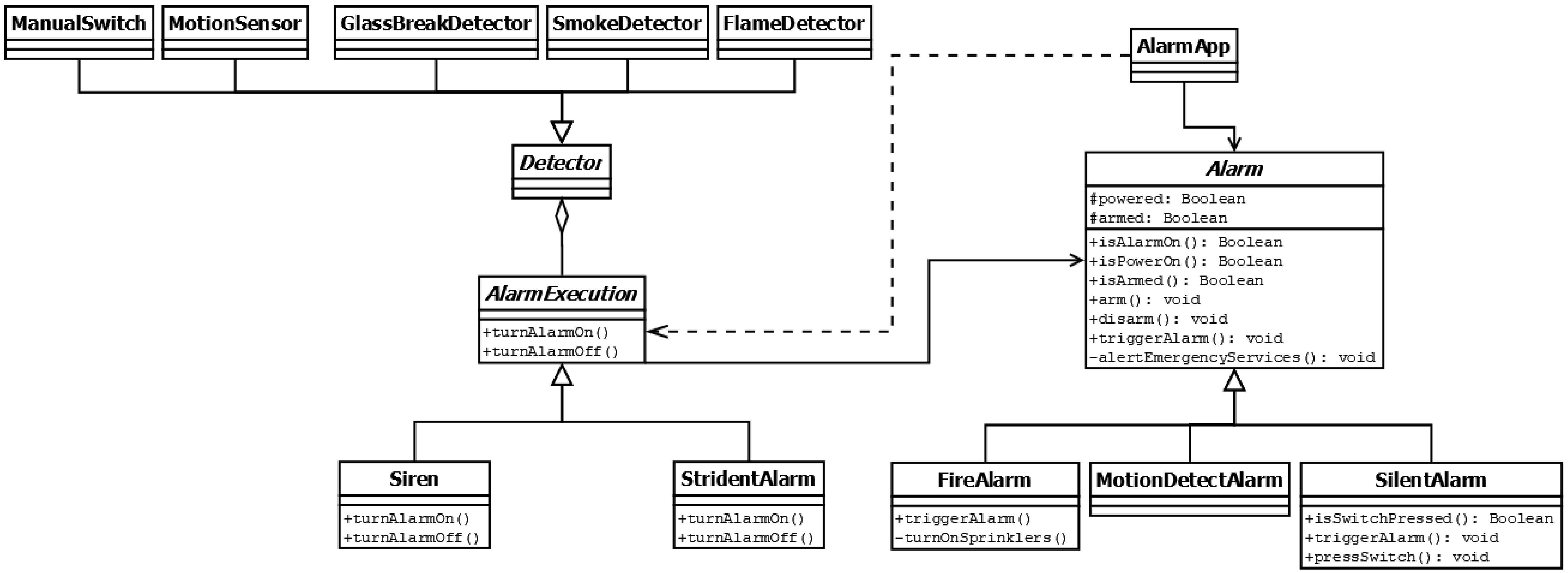
You do not need to draw the entire design's class diagram, only draw the section you are adding and clearly show where it joins onto the original design. If you draw on the enlarged figure provided in the supplementary materials, do not forget to put your details on that sheet of paper and submit it with your answer booklet. You do not need to write any code.

End of Examination

SENG 301 2019 Exam Supplementary Material

Please make sure that your details are filled in and that this is tied to your answer booklet.

Family Name	____________________
First Name	____________________
Student Number	|__|__|__|__|__|__|__|__|
Venue	___________________
Seat Number	________


ManualSwitch
MotionSensor
GlassBreakDetector
SmokeDetector
FlameDetector
AlarmApp
Detector
AlarmExecution
+turnAlarmOn()
+turnAlarmOff()
Alarm
#powered: Boolean
#armed: Boolean
+isAlarmOn(): Boolean
+isPowerOn(): Boolean
+isArmed(): Boolean
+arm(): void
+disarm(): void
+triggerAlarm(): void
-alertEmergencyServices(): void
Siren
+turnAlarmOn()
+turnAlarmOff()
StridentAlarm
+turnAlarmOn()
+turnAlarmOff()
FireAlarm
+triggerAlarm()
-turnOnSprinklers()
MotionDetectAlarm
SilentAlarm
+isSwitchPressed(): Boolean
+triggerAlarm(): void
+pressSwitch(): void