2016

Q1 Software Engineering

1.Not every code development counts as "software engineering". List three aspects that are characteristic of true "software engineering" projects.

2.There are four factors of software development of which the client may prioritize three, By improving what property of the software development process can one gain an advantage regarding all four factors? Briefly point out why this is the case for two of these factors.[6 marks]

3.Is there a difference between "building the right (desired) system" and "building the system right (properly)"? Briefly explain your answer using corresponding software engineering terminology.[6 marks]

4.Briefly explain what the term "maintenance" means in software engineering and what typical activities are performed in the maintenance phase.[6 marks]

1. Briefly discuss the meaning and significance of "maintenance" in software engineering.

[4 marks]

1. What are symptoms of a software system that is hard to maintain? [5 marks
2. Fewer interfaces are considered better than many interfaces in software engineering. Briefly explain why this does not imply that zero interfaces are optimal. [5 marks]
3. Briefly discuss the potential benefits and dangers involved in reusing software compo­nents. [6 marks
4. [4 marks] What are the four factors in software development of which the client may prioritize a maximum of three?

[2 marks] Why do you think 40 years of software engineering practice and research have not been able to lead software development out of the above mentioned crisis?

(c) [4 marks] Briefly explain the terms “Correctness” and “Robustness” making sure the difference between them is clear.

(d) [10 marks] Describe the steps a software engineer can take to minimise the possibility of software project failing using at least two of the examples discussed by at least two different group project presentations in the class.

1. Briefly discuss the meaning and significance of "maintenance" in software engineering

(b) The maintainability of a component correlates with the size of its interface. Briefly describe this correlation and mention two technical properties that components with the desirable interface size will typically exhibit.

(c) If a software system is hard to change because any change may break the system in some way, what is the system suffering from and what system property could address the problem? [4 marks]

(d) Briefly discuss the potential benefits and dangers involved in reusing software components.

1. [4 marks] Briefly discuss the meaning and significance of "maintenance" in software engineering.
2. [6 marks] The maintainability of a component correlates with the size of its interface. Briefly describe this correlation and mention two technical properties that components with the desirable interface size will typically exhibit.
3. [4 marks] If a software system is hard to change because any change may break the system in some way, what is the system suffering from and what system property could address the problem?
4. [6 marks] Briefly discuss the potential benefits and dangers involved in reusing soft-ware components.

Q2 Design Principles

1.Why are classes with low coupling desirable? Tick only one box.[2 marks]

□They increase cohesion.□They decrease cohesion. □ They increase rigidity. □ They decrease rigidity.

2.Assume module A has five components with six connections and module **B** has five components with 8 connections. Which module is more likely to contain components better suited for reuse? Briefly explain your answer?[4 marks]

3.Which - "layers" or "partitions" - are useful to address modular continuity? Explain your answer.[6 marks]

4.Assume an implementation has been verified to be correct with respect to the specification. What other properties of the implementation may the customer be interested in?

5.Briefly list the pros and cons of pre-conditions and post-conditions for the purpose of achieving modular protection. [6 marks]

6. Briefly explain the relationship between coupling and information hiding.

(c) Why do internal criteria for a software system matter, if the client is only concerned with external criteria?

(d) What causes a software system to exhibit "rigidity"?

1. What causes a software system to exhibit "fragility"?
2. [5 marks] What is meant by a “System” in Software Engineering? Please describe what it is and give some examples.
3. [5 marks] What is a system boundary? Please describe what it is and give some examples.
4. [5 marks] Explain the relationship between coupling and information hiding.
5. [5 marks] So-called control-objects manage other, comparatively passive, objects. Discuss the pros and cons of such “manager” objects.
6. Why are classes with high cohesion desirable?

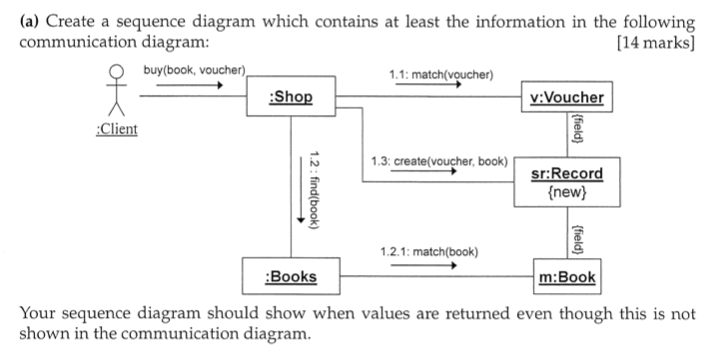
(b) Which - "layers" or "partitions" - are useful to address modular continuity? Explain your answer. [7 marks]

1. Which of the five modularity requirements that were discussed in lectures can help to improve continuity? Briefly explain your answer.

(d) Briefly explain why even a correct implementation does not guarantee full customer satisfaction and why this circumstance is not used to change the traditional development process.

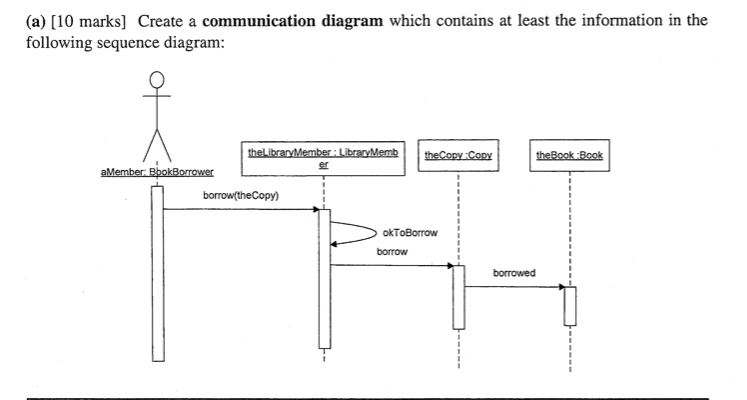
(e) [4 marks] Briefly discuss whether pre-conditions or post-conditions are a better means to achieve modular protection.

Q3 Interaction Diagrams

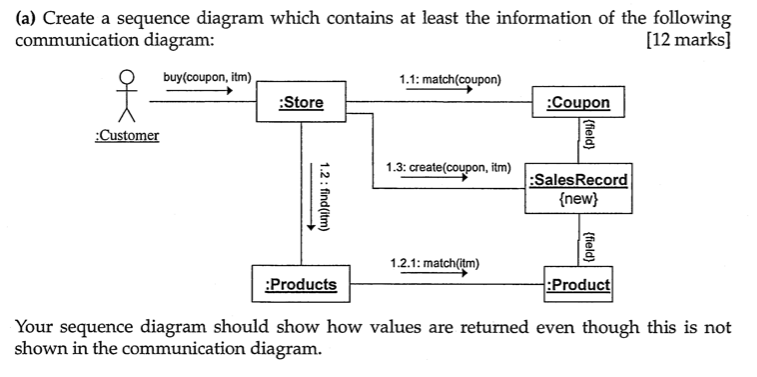


1. In which development phases can one use interaction diagrams? Briefly describe the purpose of interaction diagrams for each phase you name. [6 marks]
2. In what way can you capture alternative execution paths in a communication diagram?

[2 marks]



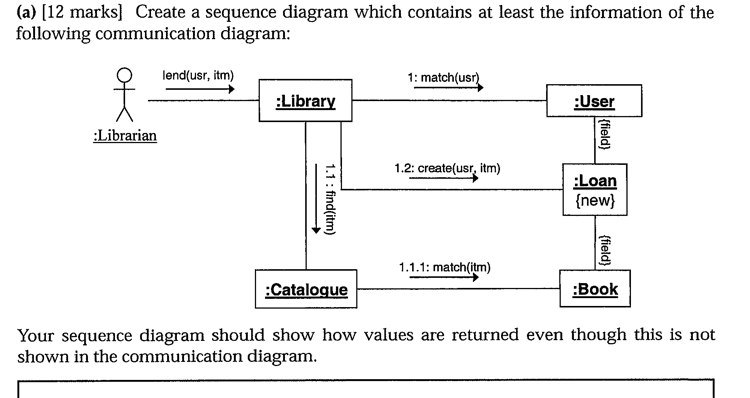
b) [5 marks] Compare and contrast sequence diagrams and communication diagrams. Discuss the advantages and disadvantages of both kinds of diagrams.



1. Briefly explain how you could use interaction diagrams in both implementation and testing phases respectively

(c) In what way can you capture alternative execution paths in a communication diagram?

1. In what way can you capture alternative execution paths in a sequence diagram?



(b) [4 marks] Briefly explain how you could use interaction diagrams in both implementation and testing phases respectively.

1. [2 marks] In what way can you capture alternative execution paths in a communication diagram?
2. [2 marks] In what way can you capture alternative execution paths in a sequence dia-grams?

Q4 State Diagrams

1. Create a UML state diagram that describes the behaviour of a cleaning robot. Initially, the robot is idle. After 48 hours of inactivity or when the user presses the "go" button on a remote control, the robot starts cleaning the floor and normally keeps moving forward until it has covered all of the floor or the user presses the "cancel" button. The user can also set the speed to "fast" (noisier operation) or "slow" (quieter operation) at any time. If the robot encounters an obstacle, it turns by 15 degrees and then reverses direction - i.e., moves backwards if it has been moving forwards and moves forwards if it has been moving backwards. At any point in time an "error" condition may occur. If the "nature" of the error is "minor" the robot will self-repair and then continue. If the "nature" of the error is "fatal", the robot will terminate all activity immediately and indefinitely.

Marks are awarded for using advanced notation that goes beyond using only states and transitions.

1. Briefly explain what superstates are typically used for and why they are considered to be an important feature of state diagrams. [6 marks]
2. Name and very briefly describe three UML language/notation features designed to deal with complexity.
3. Briefly describe the UML's approach to characterising container types and write down how you would specify that a concept is used as i) an "Ordered Set" and ii) a "Sequence".

(c)Sally wrote the following OCL constraint in order express that if an employed person is a manager, the person's income must be at least $4000:

context Person inv appropriateSalary: if self.isEmployed then

self.isManager and self.income >= 4000 endif

What would you change and why?

Create a UML state diagram that describes the states and events of a phone with the following behaviour:

Initially, the phone is idle. When an incoming call arrives, it keeps ringing until the user picks up or the caller aborts the call. In the former case the phone is connected to the calling party, while in the latter case it becomes idle again.

For an outgoing call, when the user picks up the handle, the phone keeps accepting digits until a valid number has been dialled. When this happens, the phone becomes connected to the called party.

At any point during the dialling or while being connected the user may hang up the phone, causing it to become idle again.

Note: Marks are also awarded for the appropriate use of advanced notation.

(a) Create a UML state diagram that describes the behaviour of a cleaning robot. Initially, the robot is idle. After 24 hours of inactivity or when the user presses the "clean" button on a remote control, the robot starts cleaning the floor and normally keeps moving forward until it has covered all of the floor or the user presses the "abort" button. The user can also set the speed to "fast" (noisier operation) or "slow" (quieter operation) at any time. If the robot encounters an obstacle, it turns by 25 degrees and then reverses direction - i.e., moves backwards if it has been moving forwards and moves forwards if it has been moving backwards. At any point in time an "error" condition may occur. If the "nature" of the error is "minor" the robot will self-repair and then continue. If the "nature" of the error is "fatal", the robot will terminate all activity immediately and indefinitely.

Marks are awarded for the appropriate use of advanced notation. [15 marks]

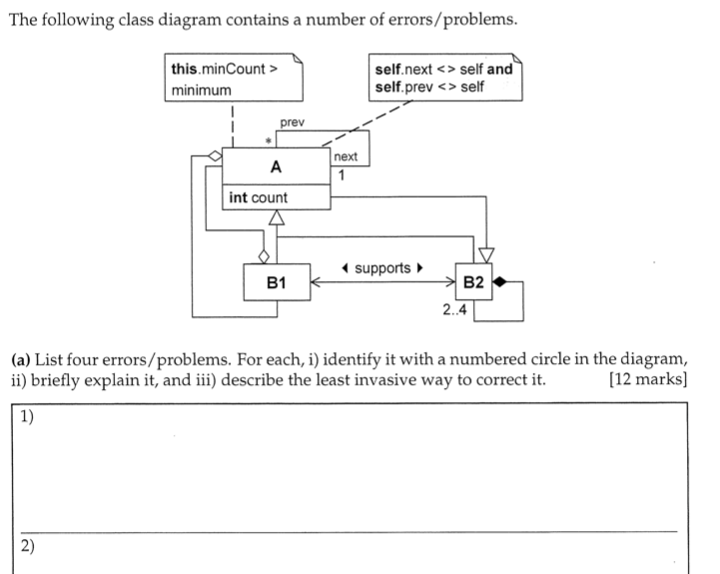
Briefly explain what superstates are typically used for and why they are considered to be an important feature of state diagrams.

(a) [15 marks] Create a UML state diagram that describes the behavior of an elevator. Initially, the elevator waits on the first floor. When a “button press” event occurs, the elevator moves to the floor number specified by the event. It is important for the elevator to move in the correct direction (up or down). When the elevator reaches the target floor, an “arrived” event is generated. The elevator should then open its doors. The elevator should close the door before it moves and return to the first floor after 30 seconds of user inactivity (i.e., no button presses). At any point during the elevator’s operation, it is possible to press the “emergency button” which will cause the elevator to return to the first floor.

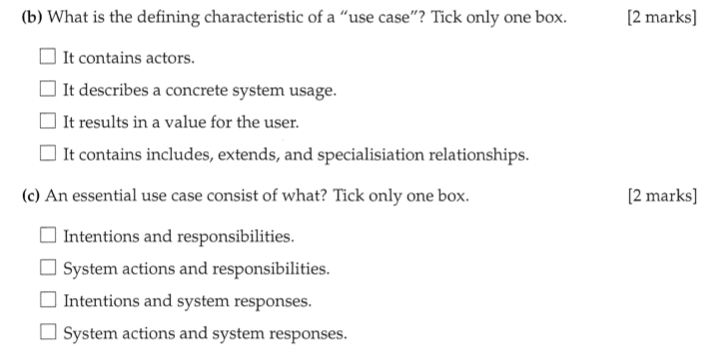
Marks are awarded for the appropriate use of advanced notation.

(b) [5 marks] Briefly explain why substates, i.e., the ability to use concurrent lanes each specifying reactive behaviour that contributes to an overall combined behaviour, can be used to reduce the complexity of state diagrams.

Q5 Modelling

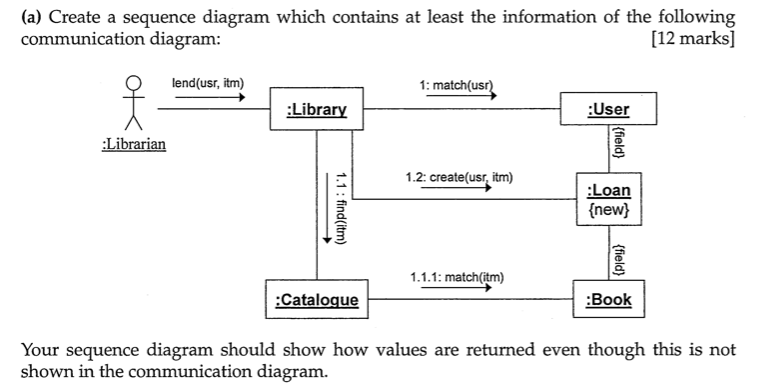


3) 4)



1. A colleague asks you what the direction of the inheritance relationship between the

concepts "Rectangle" and "Square" should be. Advise your colleague of three alternative options and briefly explain the rationale for each option. [8 marks]

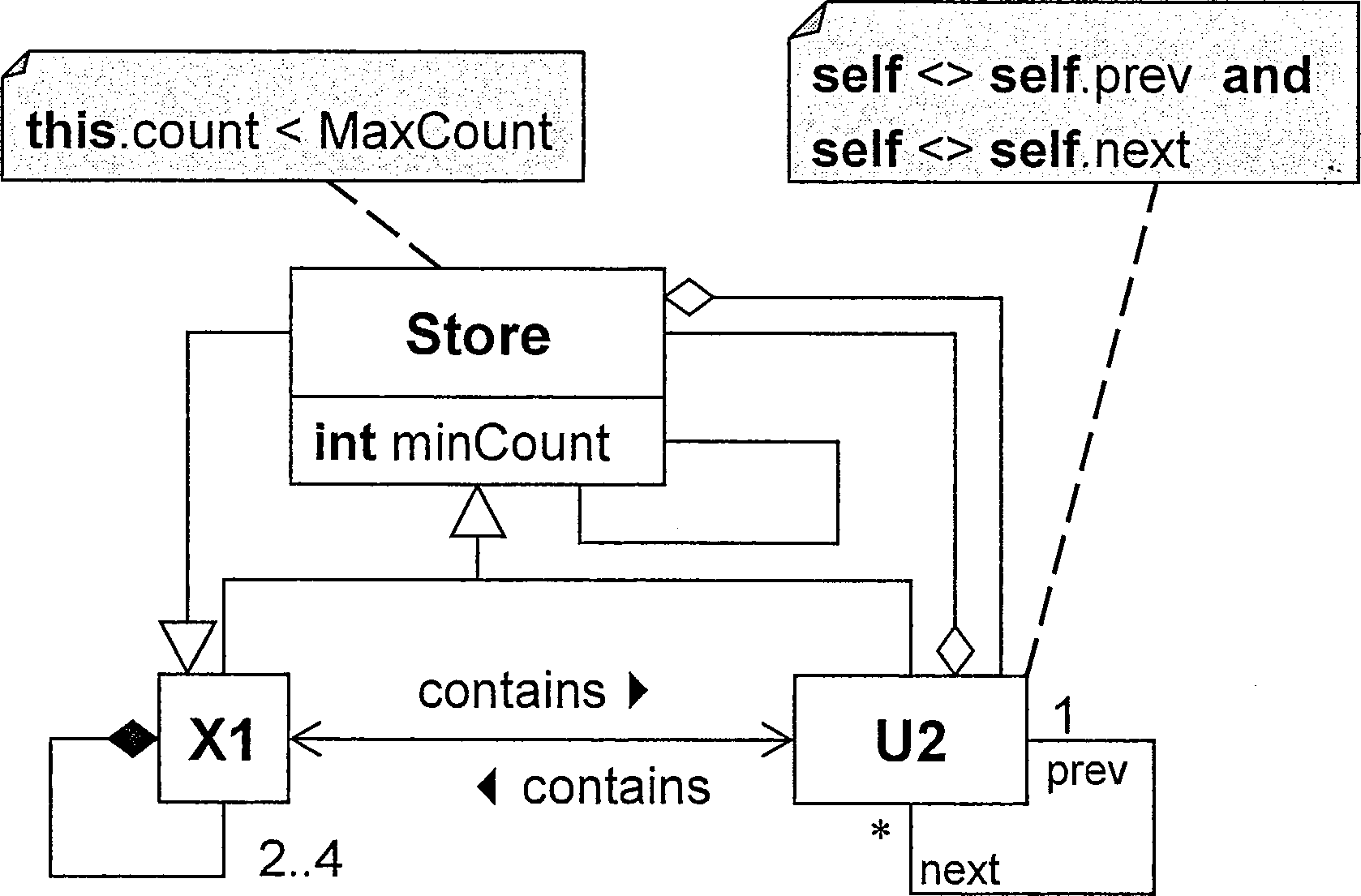


1. In what way can you capture alternative execution paths in a communication diagram?
2. Briefly explain for which purpose you would choose a sequence diagram over a communication diagram and vice versa.

(a) Create a UML state diagram that describes the states and events of a phone with the following behaviour: Initially, the phone is idle. When an incoming call arrives, it keeps ringing until the user picks up or the caller aborts the call. In the former case the phone is connected to the calling party, in the latter case it becomes idle again. In the case of an outgoing call, when the user picks up the handle, the phone keeps accepting digits until a valid number has been dialled. In the latter case, it becomes connected to the called party. At any point during the dialling or while being connected, the user may hang up, causing the phone to become idle again. Marks are awarded for the appropriate use of advanced notation. [15 marks]

b) Briefly explain when and why one would use substates, i.e., the ability to use concurrent lanes each specifying reactive behaviour that contributes to an overall combined behaviour.

The following class diagram contains a number of errors/problems.



(a) List four errors/problems. For each, i) identify it with a numbered circle in the diagram, ii) briefly explain it, and iii) describe the least invasive way to correct it.

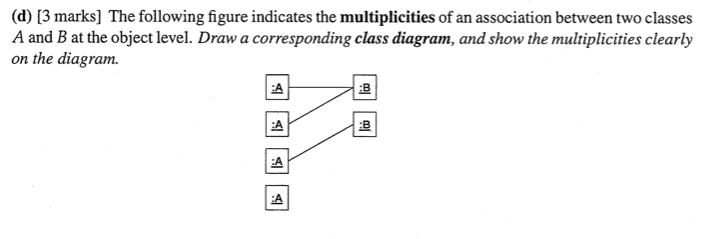
1）2）3）4）

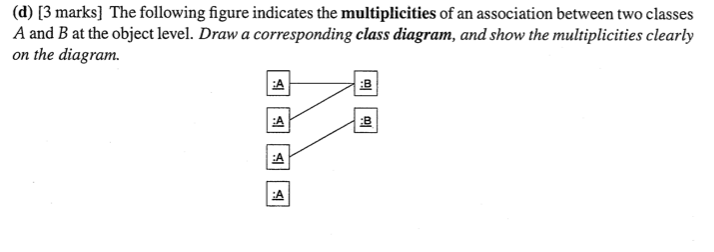
(b) A colleague asks you what the direction of the inheritance relationship between the concepts "Circle" and "Ellipse " should be. Advise your colleague of three alternative options and briefly explain the rationale for each option.

­­ [4 marks] List and discuss the properties that make a good class diagram.

[2 marks] Explain why class diagrams are already created in the analysis stage of software engineering.

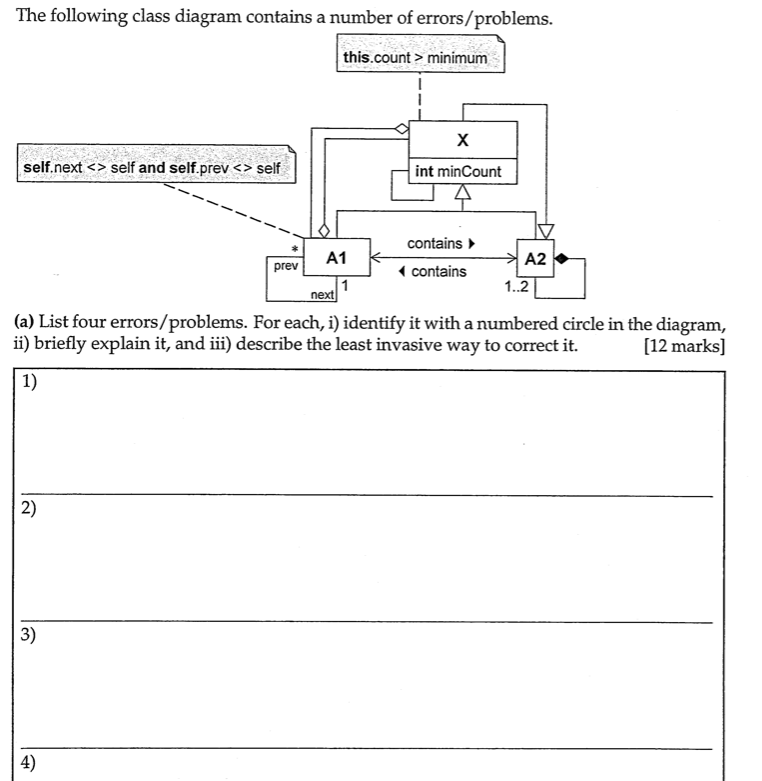
[6 marks] Aggregation and composition are two kinds of association in class diagrams. Give an example for each of them. Discuss the differences between aggregation and composition. Use your examples to illustrate the differences.





1. [6 marks] Discuss why the use of association classes and higher-arity associations is a good idea. Give an example of a higher-arity association to illustrate the benefits.
2. Explain the purpose of the Object Constraints Language (OCL).
3. [6 marks] What are invariants in the context of OCL? Give an example of an invariant using OCL and explain its meaning.

b) A colleague asks you what the direction of the inheritance relationship between the shape concepts "Square" and "Rectangle" should be. Advise your colleague of the options available and explain what principle you use to justify each option.



[8 marks] A colleague asks you what the direction of the inheritance relationship between the concepts "Dictionary" and "Set " should be. Advise your colleague of three alternative options and briefly explain the rationale for each option.

UML

(a)Briefly explain what a "use case" is. Include the ultimate criterion that determines whether something really should be regarded as a use case.

(b)[2 marks] Briefly explain the idea of an "essential use case" (as opposed to a "system use case").

(b)Briefly explain the difference between a "system use case" and an "essential use case".

Describe two different kinds of situations when one would need to add textual constraints to a conceptual diagram. Provide an example for each respectively. [6 marks]

Consider a high security facility. Access is possibly only to individuals that can identify themselves with a staff card or per finger print reader. In rare "alert" situations, access is possible only for a subset of "VIP" users.

(d)In what way can the potentially rich use case "Access Facility" be factored into multiple smaller parts that are easier to deal with on their own? Base your factorization on the three UML use case relationships.

(e)Draw the UML use case diagram for your design of question (d).

Describe two different kinds of situations when one would need OCL constraints in modelling. Provide an example for each respectively.

Consider a university library. When library users return items they need to identify them-selves through their library card or a staff card. In rare cases it may not be possible for the library staff to accept an item because the latter needs to be handled by a special staff member that is absent.

(d) [4 marks] In what way can the potentially rich use case "Return Item" be factored into multiple smaller parts that are easier to deal with on their own? Base your factorization on the three UML use case relationships.

(e)[5 marks] Draw the UML use case diagram for your design of question (d