TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

EXAMINATIONS — 2008

END-OF-YEAR

SWEN 102 Introduction to Software Modelling

Time Allowed: 3 Hours

Instructions: There are 180 possible marks on the exam.
Answer all questions in the boxes provided.
Every box requires an answer.
If additional space is required you may use a separate answer booklet.
Some example Alloy code is provided on the last page.
Non-electronic foreign language dictionaries are allowed.
Calculators ARE NOT ALLOWED.
No other reference material is allowed.

Question	Topic	Marks
1.	Use Case Diagrams	30
2.	Object and Class Diagrams I	30
3.	Object and Class Diagrams II	30
4.	Writing Invariants	30
5.	Using Alloy	30
6.	State Machines	30
	Total	180

continued...

Student ID: . . . .

Question 1. Use Case Diagrams [30 marks]

(a) [3 marks] Perform a *textual analysis* on the following description, to find candidate use cases.

You should carefully and neatly underline key verb phrases in the text in the box.

The GearHead website lets people read and write reviews of musical equipment, like guitars, amplifiers, and keyboards.

GearHead is mainly used to read reviews. Anyone can find an item (e.g. a "Value Junior Amplifier") by searching, or by reading lists in each category (e.g. "Amplifiers"). The website then shows all the reviews for that item.

Anyone with an email address can register on the site as a Reviewer. Reviewers have to log in, and can then write a review of any item. Each review must be checked by a Checker (usually someone who has written several good reviews) before being posted.

New items can be added to GearHead by Manufacturers, who upload a picture and a description of their items to the categories. Manufacturers who read bad reviews of their items can request that those reviews are deleted.

Finally, system administrators must add and delete information about the Reviewers, Checkers, Manufacturers, and anyone else who is permitted to use the system.

Student ID: . . . .

(b) [6 marks] The description text is incomplete. Give **two** questions you would ask users or clients to clarify these requirements. If necessary, also give the answer you have **assumed** to this question.

Student ID: . . . .

(c) [12 marks] Draw **essential use case cards** for the following **three** use cases in this system.

Find Item by Browsing Categories

Write Review of Item

Add New Item

(d) [9 marks] Draw a **use case diagram** showing at least 3 actors and at least 6 use cases that you would produce in a model of this system.

Student ID: . . . .

Question 2. Object and Class Diagrams I [30 marks]

The following text describes a computerised entertainment system:

> A Seat-Back Entertainment system can display CDs, movies, and TV shows on the back of the seats in an aeroplane.
> A movie has a title, a display poster image, a director, and a running time (in minutes).
> A CD has a title, an artist, and a cover image; it also has a number of tracks each of which has its own title and length (also in minutes).
> A TV show also has a title and an advertising image. TV shows are always 30 minutes long.
> As well as playing whole CDs, the Seat-Back Entertainment system lets people define their own playlists. A playlist is just a list of tracks, taken from any number of different CDs. Playlists cannot include movies or TV shows.
> Some CDs, TV shows and movies are specially suitable for children. Users (most likely parents) can set up the Seat-Back entertainment system so that only the children's entertainment will be presented.

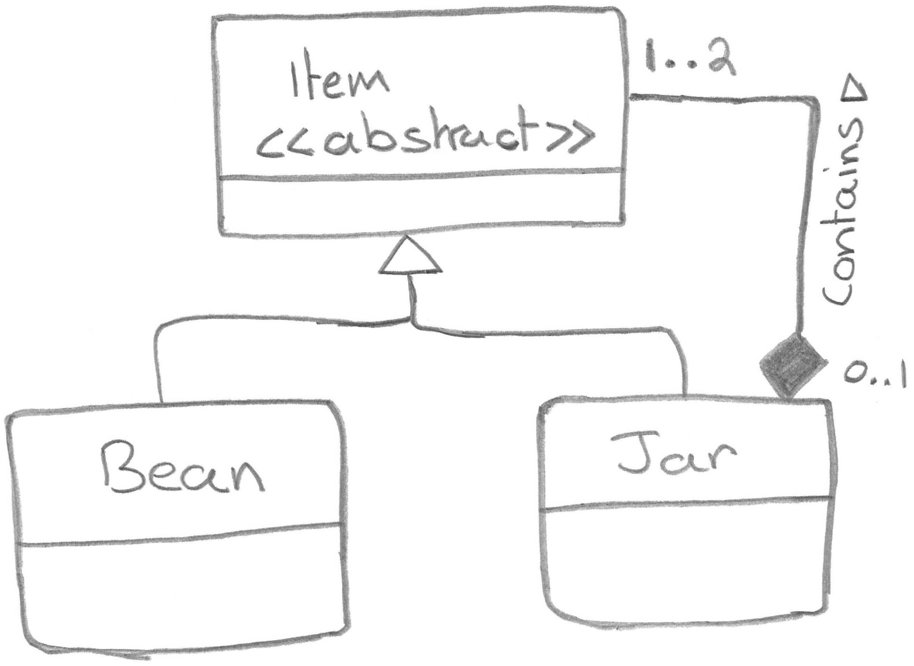
Student ID: ....

(a) [10 marks] Draw a **class diagram** showing class attributes, to model this system.

(b) [5 marks] Describe how your model handles children's entertainment. Why did you chose this design?

Student ID: . . . .

Consider the following class diagram for modelling beans in Jars:



(c) [5 marks] In your own words, describe two important properties of `Beans` and `Jars` which are captured by the class diagram above.

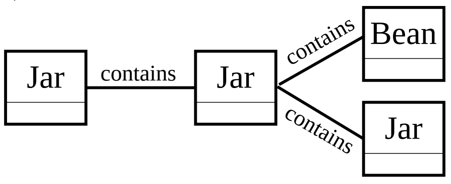
i)

ii)

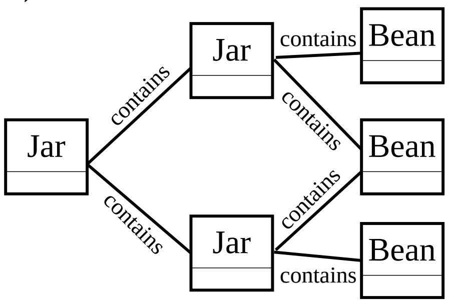
Student ID: ....

(d) [10 marks] For each of the following object diagrams indicate whether or not it is *consistent* with the class diagram on the previous page and, if not, explain why.

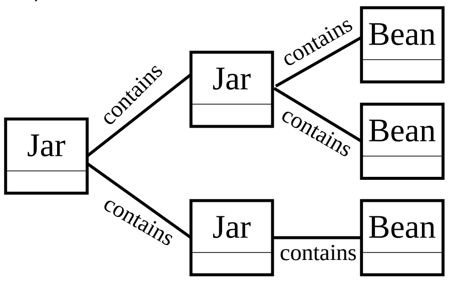
i)



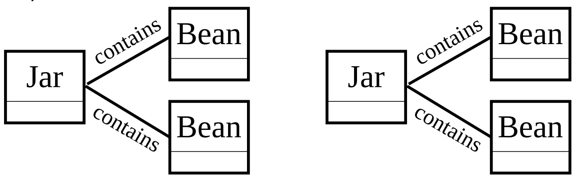
ii)



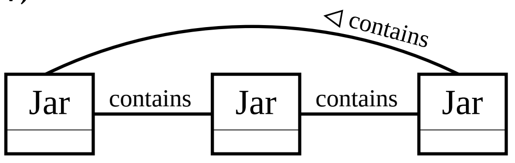
iii)



iv)



v)



Student ID: . . . .

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.

Student ID: . . . .

SPARE PAGE FOR EXTRA ANSWERS

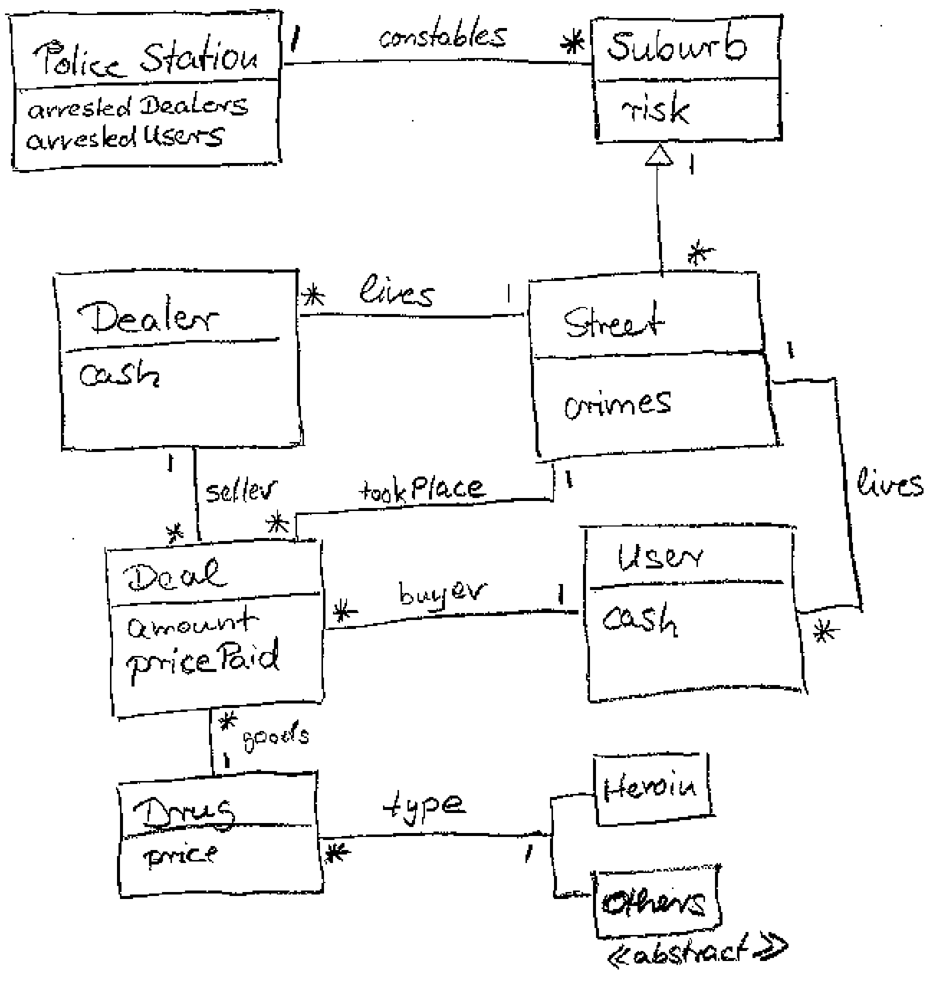
Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Student ID: ....

Question 3. Object and Class Diagrams II [30 marks]

As part of New Zealand's National Drug Policy, a simulation of drug use and markets is developed. The class diagram below shows an attempt to model the system according to the description given next.

> A suburb is made up of one or more streets. The main characteristic of a street is the number of crimes recorded. Each street also has a wealth value, indicating the social and material capital of the place. Risk values are used to calculate social dissatisfaction at the level of the suburb.
>
> There is exactly one Police Station for the whole system. The Police Station sends constables to suburbs with risk values higher than 5.
>
> The simulation includes social agents like users and dealers, for which cash and home street are given. Dealers sell drugs to users, and for each transaction (deal) the drug type, the amount, the price, and the agents involved are recorded.



Student ID: ....

(a) [15 marks] Circle and number **five distinct** problems in the class diagram on the previous page. Describe why each problem is a problem and how you would solve it.

1.

2.

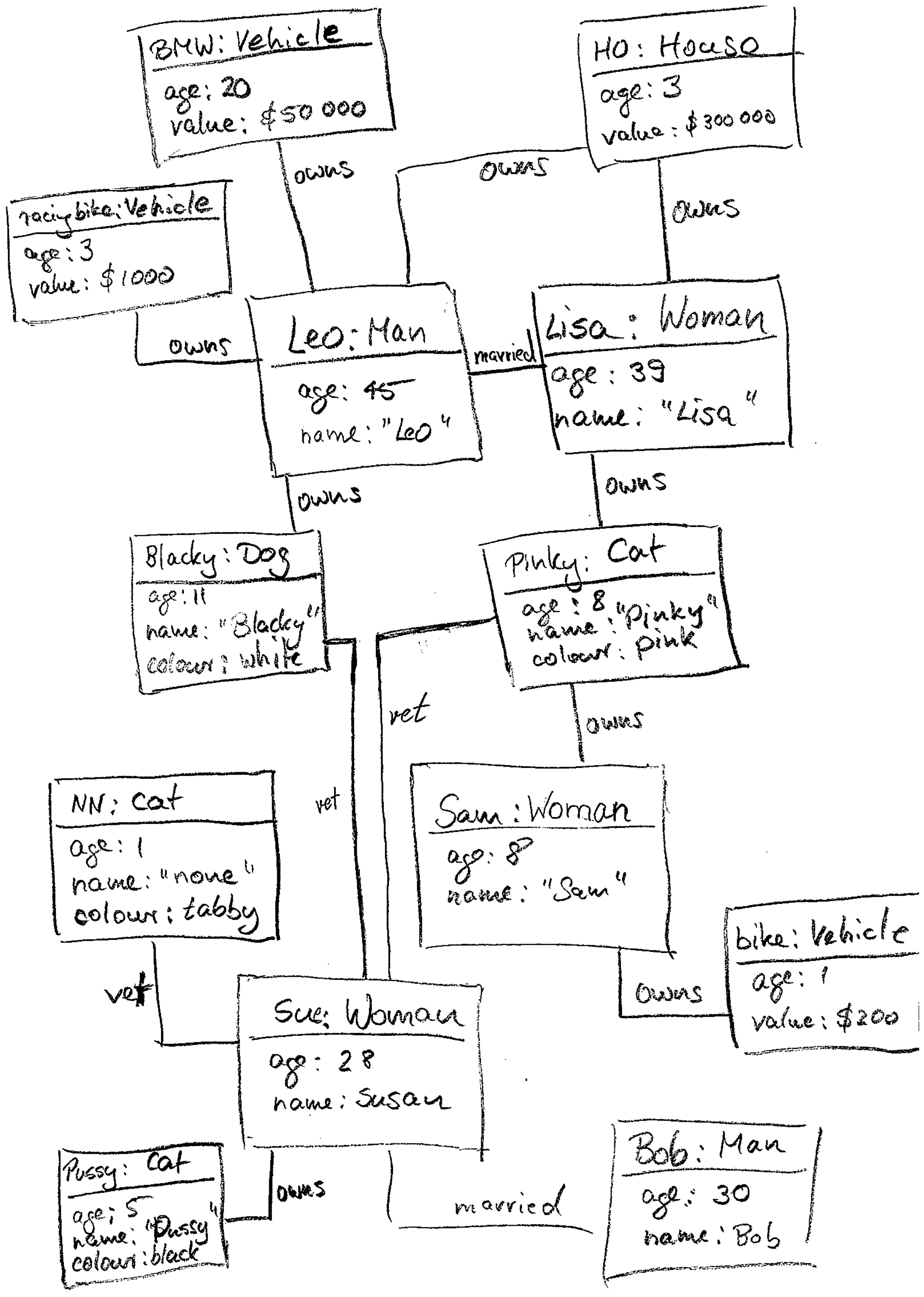
3.

4.

5.

Student ID: . . . .

(b) [15 marks] Consider the object diagram below and draw a corresponding class diagram on the facing page.



Student ID: . . . .

Student ID: . . . .

SPARE PAGE FOR EXTRA ANSWERS

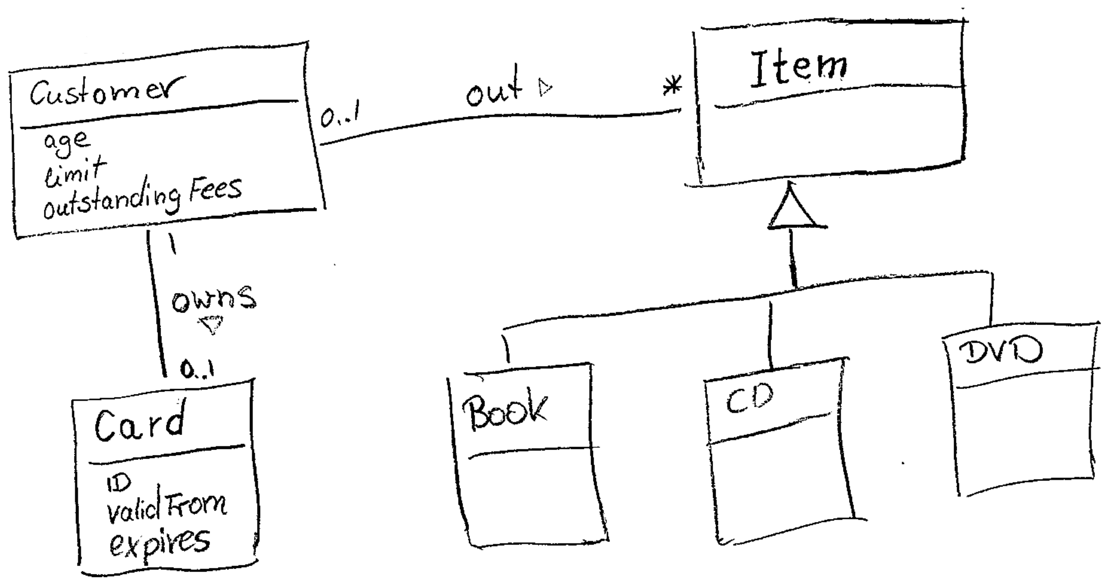
Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Student ID: . . . .

Question 4. Writing Invariants [30 marks]

Consider the following description for a *library system*, which is made up of some *text* and a *class diagram*:

> The Tauwhare Country Library System serves the people residing in and around Tauwhare. Customers can use their cards to borrow items like books, CDs, and DVDs.
>
> Each library card has a unique ID, a valid from date and a date for when the card expires. The items a customer currently has out are recorded. Each customer has a limit associated and can't have more items out than this limit allows. If a customer does not return an item on time, a fee is charged. Outstanding fees can neither be negative nor exceed 500. The upper limit prevents the fees from rising indefinitely if an item is never returned.



Student ID: . . . .

(a) [10 marks] By considering the text and class diagram given for the library system identify (in English) five *candidate invariants*:

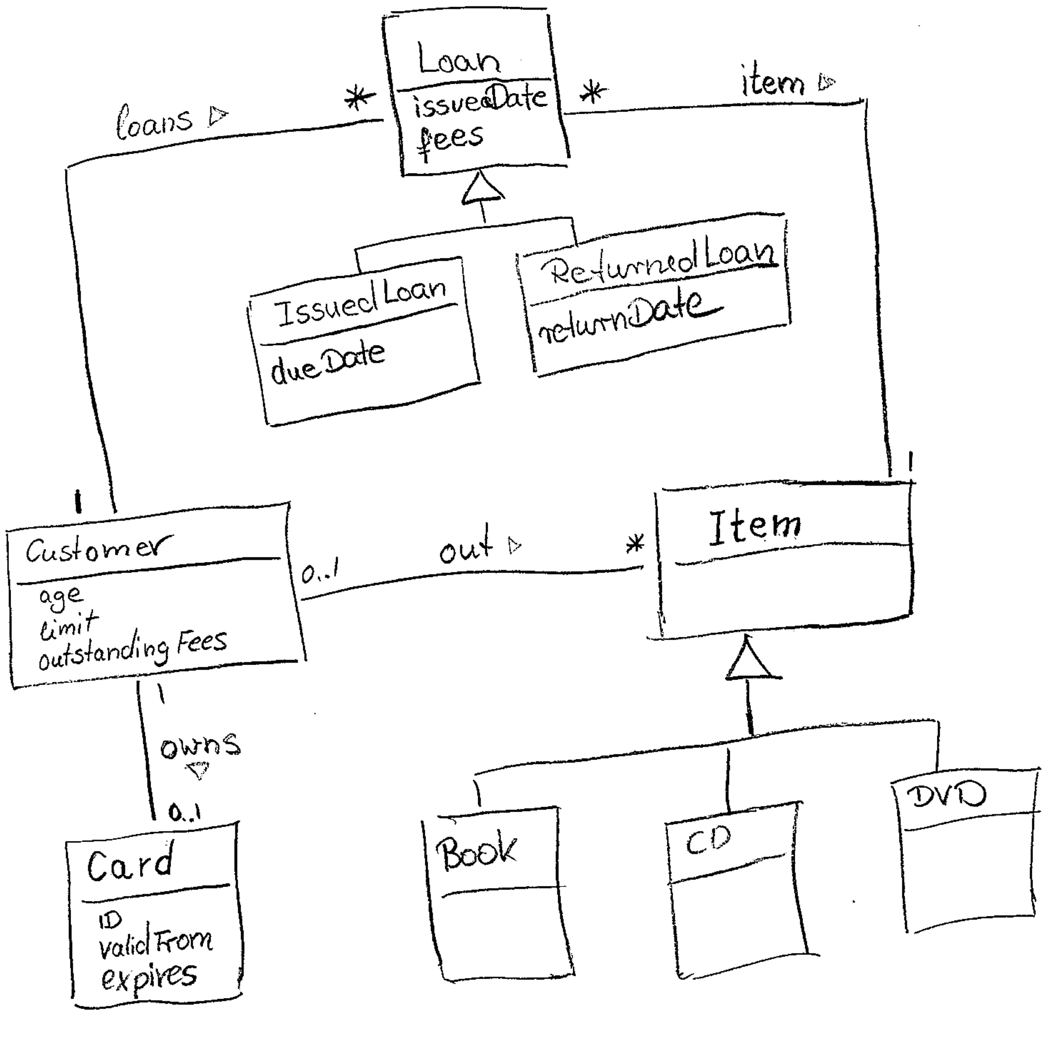
Student ID: . . . .

(b) [10 marks] Translate your candidate invariants from **(a)** into the Alloy-like syntax presented in lectures (there is example Alloy code provided on the last page):

Student ID: . . . .

The library system was extended to support records of loans as shown in the following class diagram. For each item a customer has out, an issued loan record is created. When the customer returns an item, the issue and return date as well as the fees are recorded as a returned loan for later reference.

The following *invariants* are given for the extended library system.



```
all i: IssuedLoan | i.item in i.~loans.out

Customer invariants:

  all i: out | one l : IssuedLoan | i = l.item and l in loans

  0 <= outstandingFees and outstandingFees <= (sum l: loans | l.fees)

  age < 18 implies no out & DVD
```

Student ID: . . . .

(c) [10 marks] Translate the four invariants in Alloy given on the previous page into written English:

Student ID: . . . .

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.

Student ID: . . . .

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.

Student ID: . . . .

Question 5. Using Alloy [30 marks]

Consider the following description of a road network:

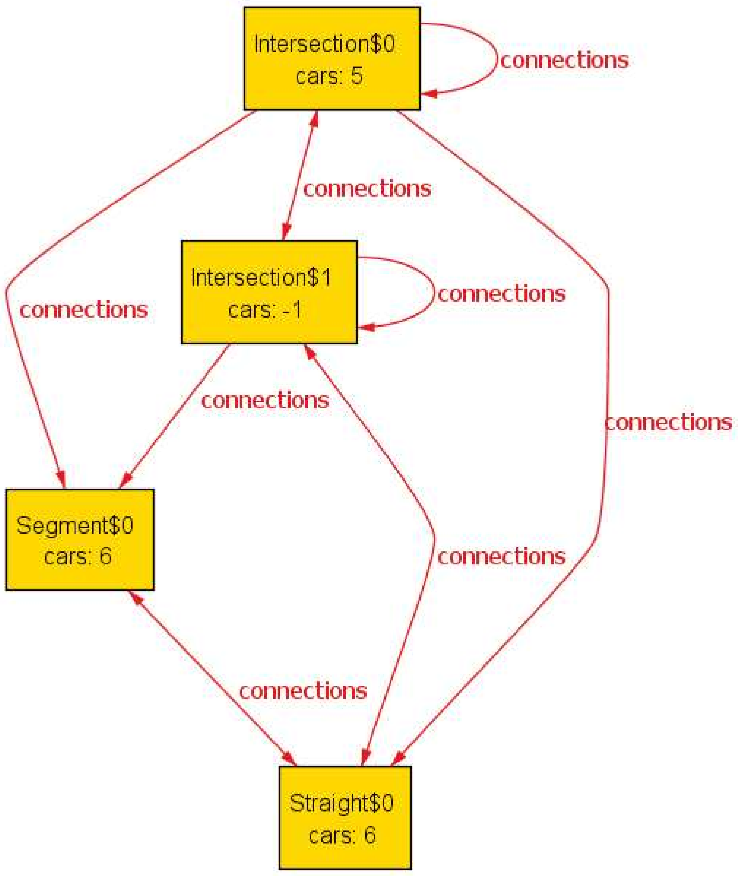
> "A road network consists of one or more road segments connected together. A road segment is either a straight segment, or an intersection. Straight segments have two connection points, whilst intersections have three or more connection-points. Road segments cannot connect to themselves! Also, if one road segment is connected to another, then the opposite must be true (i.e. the other must be connected to the first). An intersection always connects to straight segments, never directly to other intersections. Finally, each road segment counts the number of cars using it at that moment in time."

An *incorrect* model of a road network in Alloy and an *object diagram* of that model is given below:

```
sig Segment {
 connections : some Segment,
 cars : Int
}{}

sig Straight extends Segment {}{
 #connections = 2
}

sig Intersection extends Segment {}{
 #connections >= 3
}
```



The above object diagram was generated using the command "`run {} for 4`" from the Alloy model.

Student ID: . . . .

(a) [5 marks] Evaluate each of the following Alloy expressions on the object diagram given on the previous page:

```
Intersection$0.connections
```

```
Segment$0.~connections
```

```
Segment$0.connections & Intersection$1.connections
```

Student ID: ....

(b) [10 marks] Circle and number five distinct ways in which the object diagram given on the previous page is *inconsistent* with the description of a road network. For each, write a brief (i.e. one line) **description of the problem** in the corresponding box below.

1)

2)

3)

4)

5)

Student ID: . . . .

(c) [15 marks] For each problem identified in **(b)**, indicate what changes you would make to the Alloy model to fix it and **give Alloy code to illustrate**.

1)

2)

3)

4)

5)

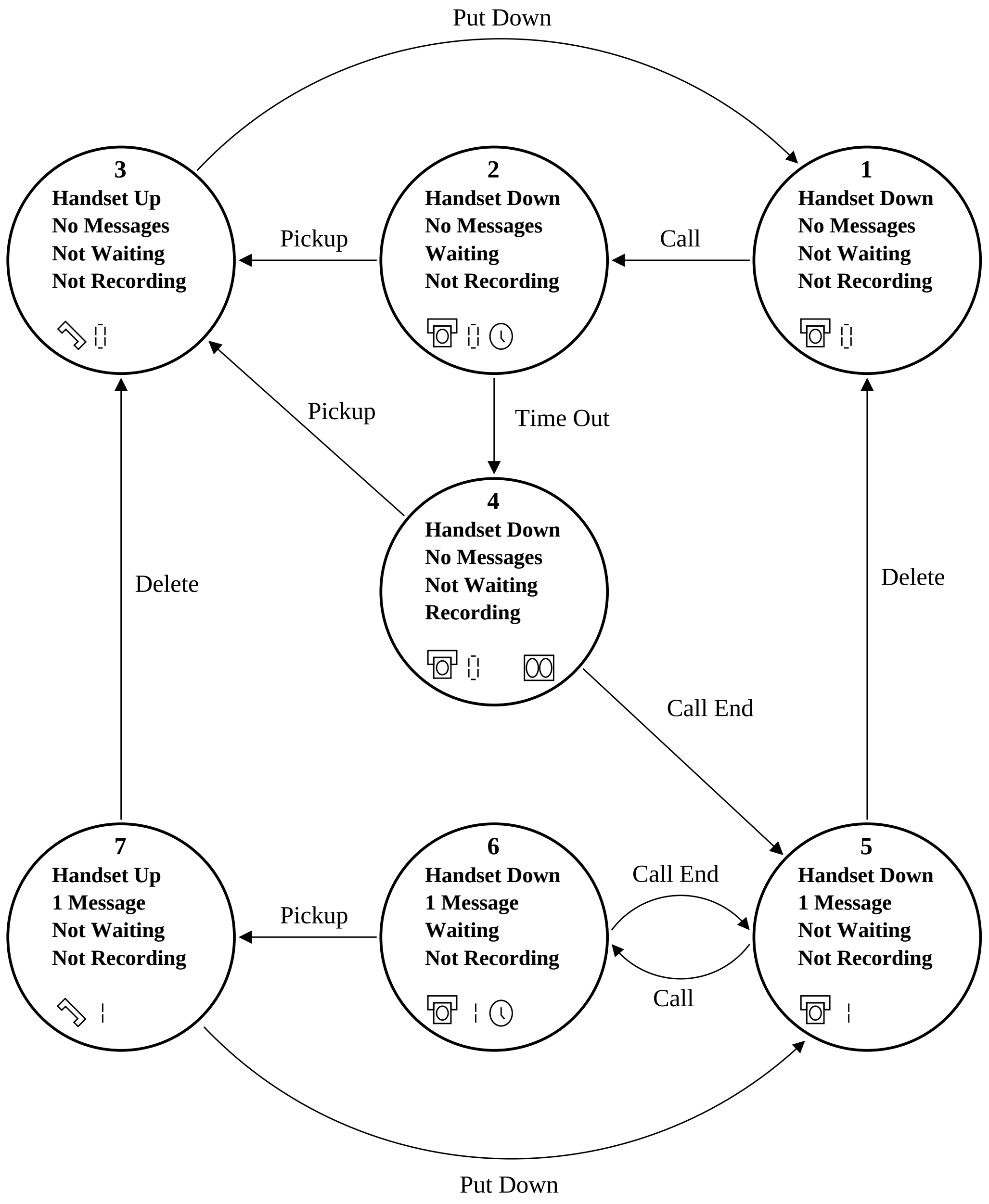
Student ID: . . . .

Question 6. State Machines [30 marks]

Consider the following description of a simple *answering machine*:

> “The answering machine can store up to one phone message, and has a handset for answering calls manually. When an incoming call rings, a timer is started. If the user picks up the handset before the timer elapses, then he/she may answer the call. Otherwise, the machine begins to record the message. At this point, the user may still pick up the handset and answer the call. ”

A *state machine diagram* for the answering machine has been provided:



Student ID: . . . .

(a) [10 marks] For each of the following statements, indicate whether you think it is a true or false statement based on the state machine diagram.

(i) In states 1 and 5, the answer machine is just waiting for the next call.

(ii) In states 3 and 6, the user is answering a call.

(iii) If an incoming call occurs and the handset is up, the machine will record a message.

(iv) The user may delete a message whilst answering a call.

(v) If the machine has already stored a message, then it will not record another until the first is deleted.

(b) [4 marks] Provide a suitable *execution trace* for the following scenarios. For each state, you need only list its number. Your execution trace may start from any state you chose.

(i)

"When the phone rang, Jane answered it immediately. It was a wrong number, so she put the handset down."

(ii)

"John deleted the message. Then, the phone rang, but he could not answer it. So, a new message was recorded."

Student ID: . . . .

(c) Consider the following *incomplete* Alloy model of the answer machine:

```
sig MachineState { waiting : Bool, recording : Bool, messages : Int }

pred call(s1, s2 : MachineState) {
  s1.waiting = False
  s2.waiting = True
  s2.recording = False
  s1.messages = s2.messages
}

pred timeout(s1, s2 : MachineState) {
  s1.waiting = True
  s2.waiting = False
  s2.recording = True
  s1.messages = s2.messages
}

pred callend(s1, s2 : MachineState) {
  s1.recording = True
  s2.waiting = False
  s2.recording = False
  s1.messages+1 = s2.messages
}

sig ExecutionTrace { states : seq MachineState }{
  states[0].waiting = False
  states[0].recording = False
  states[0].messages = 0

  all disj i,j : states.inds | j=i+1 implies
    (call[states[i],states[j]] or timeout[states[i],states[j]] or callend[states[i],states[j]])
}
```

(i) [2 marks] What difference is there between a state in the state machine diagram, and a state in the Alloy model above?

(ii) [2 marks] Which transitions from the state machine diagram are missing in the Alloy model?

(iii) [2 marks] What functionality is provided by the Alloy model but was not in the state machine diagram?

(iv) [2 marks] What can you tell about the first state in an ExecutionTrace?

(v) [2 marks] The state machine diagram has seven states. How many does the Alloy model have?

(vi) [2 marks] When can a transition occur in the Alloy model but not in the state machine diagram?

(d) The following fact was added to the Alloy model on the previous page.

```
fact {
  all m : MachineState | one e : ExecutionTrace | m in e.states.elems
}
```

(i) [2 marks] Describe in your own words what this fact says.

(ii) [2 marks] Why do you think this fact may be useful?

****

Student ID: . . . .

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.

Student ID: . . . .

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.

Student ID: ....

(This page may be detached)

Appendix

Example Alloy code for the Monopoly board is given here for reference.

```
sig Player {} {}
abstract sig Boolean {}
lone sig True, False extends Boolean {}

abstract sig Building {}
sig House extends Building {}
sig Hotel extends Building {}

sig Property {
  name: String, owner: lone Player, buildings: set Building,
  mortgage : Boolean, colourGroup : one ColourGroup
}{
 buildings in Property some -> set Building
 #buildings <= 4
 some h : Hotel | h in buildings
 all h : Hotel | h in buildings implies no h' : House | h' in buildings
 mortgage = True => no buildings
 some buildings implies one owner
}

sig ColourGroup { colour: String
}{
 all cg : ColourGroup | cg.colour = colour implies cg = this

 #~colourGroup >= 2 && #~colourGroup <= 3

 all p : this.~colourGroup | (some p.buildings) implies
   (all p' : this.~colourGroup | p.owner = p'.owner)

 all disj p, p' : this.~colourGroup, h: Hotel |
   (h in p.buildings) implies
    ((some h' : Hotel | h' in p'.buildings) || #p'.buildings = 4)

 all disj p, p' : this.~colourGroup |
   (no h : Hotel | h in p.buildings) implies (
   #p.buildings = #p'.buildings ||
   #p.buildings = #p'.buildings + 1 ||
   #p.buildings = #p'.buildings - 1 ||
   (#p.buildings = 4 && some h : Hotel | h in p'.buildings))
}
```