

Project

Value

50%

Due in

2009-06-08 23:59UTC+01 (Monday evening) by email, first word of title is 'Linz' to dlightfoot@brookes.ac.uk with one attached *Microsoft Word* (or *pdf*) document (or *zipped* equivalent). You may send a scanned, handwritten document but I shall not be able to give marks if I cannot read it.

What you have to do

Put the *title*, *your name* and *your student number* on the first page of your document.

Give brief explanations of any assumptions you make. If you have any questions about what is required please send email to David Lightfoot at the address above.

Make use of the *Zed* font. In *Microsoft Word* use *Insert Symbol* to put the characters into your document and save your file with the option 'Embed Truetype fonts'.

Answer *all three* questions. This must be *individual* work, not group work.

Question 1 – formal specification

A company owns a holiday apartment that it rents out for certain whole weeks of the year (the weeks it makes available are its 'season') to one party of guests per week. Each party has a leader, who is responsible for making arrangements, payments and so on and a total number of guests for that week, which must not exceed the capacity of the apartment.

The company currently uses a computer spreadsheet to record bookings, but has had some problems (including double booking) and would like to have a proper computer booking system.

Include brief, narrative, English explanations of each of your schemas.

- a) Explain what *constraints* you perceive on the system and explain how these might not be readily controlled by a spreadsheet.
2 marks
- b) Write names and descriptions for the *basic types* that would be needed for a Z specification of the system.
2 marks
- c) Devise a Z schema to describe the state of the booking system including all *invariant* properties (constraints).
4 marks

- d) Write a schema for an *initial* starting state and show informally that this state satisfies the invariant properties.
2 marks
- e) Write an operation schema to make a *booking* for the apartment in a given week, for a given leader and a given number of guests. Explain informally how your initialisation schema maintains the invariant properties of the state.
4 marks
- f) Write a schema that returns the *weeks* when the apartment is available.
3 marks
- g) Write a schema to *cancel* the booking for a given week.
3 marks
- Total 20 marks**

Question 2 – refinement

- a) Propose a *refinement* for each of the abstract data structures of your formal specification of Question 1. Explain your refinement.
3 marks
- b) Write a state schema that defines your refinement and its invariant properties.
3 marks
- c) Write a schema for the *abstraction function* of your refinement (also known as *retrieve relation* or *gluing invariant*).
3 marks
- d) Write a schema for an initialisation operation for your refinement.
3 marks
- e) Write a schema for an operation to make a booking for the apartment, using the variables of your refinement.
3 marks
- Total 15 marks**

Question 3 – formal program derivation

- a) Explain, with the aid of examples, what is meant by the term 'precondition'.

1 mark

The remainder of this question uses the following declarations:

const capacity = (* some constant value, capacity > 0 *)

var

s: **array** capacity **of** char; (* indexed from 0 to capacity – 1 *)

len: integer; (* (0 ≤ len) & (len ≤ capacity) *)

(* array s holds a string of characters in elements 0 .. len – 1 *)

procedure Delete (p, n: integer);

(* deletes the n characters of s starting at p (indexed from 0) *)

- b) Sketch a diagram of the array s that indicates precisely the positions of 0, capacity and len and possible values of the parameters p and n.

2 marks

- c) Write a Boolean expression that states the precondition of the procedure *Delete*. Include in your precondition the terms *olds = s* and *oldLen = len*, where *olds* and *oldLen* are 'specification variables' that refer to the values of s and len before the procedure has operated. (Note that specification variables do not appear in your program text itself, but are used in comments as part of the explanation of the program).

2 marks

- d) Write an expression for the postcondition of *Delete(p, n)*. Make use of universal quantification (\forall) and refer to the *specification* variables *olds* and *oldLen*.

3 marks

- e) State a suitable *strategy* for finding an invariant for the repetition (loop) needed to achieve the postcondition of *Delete* and state the invariant that you find. You should omit the updating of len from the invariant since it need not be a part of the repetition.

3 marks

- f) State the *guard* of your repetition and justify its choice.

1 mark

- g) State a suitable *bound function* (*variant*) and use it to demonstrate that your repetition will terminate.

2 marks

- h) Bring all of the above together to write a fully annotated body for the procedure *Delete* including the updating of len.

1 mark

Total 15 marks