

**CSE4213 Examination  
2002  
Instructions to Candidates**

1. Examination time 2 hours
2. Answer all questions
3. There are 10 questions
4. Each question is worth 8 marks
5. Total marks 80
6. Calculators are not allowed
7. Use left hand page for rough working; this page will NOT be marked unless explicitly requested.
8. The *Concise Summary of the B mathematical toolkit* is supplied.

1. Give 4 advantages of using the B Method for software development

*(8 marks)*

2. Write a brief comment (no more than 100 words) on the role of formality in software engineering.

*(8 marks)*

3. Explain the difference between **specification** and **implementation** in the light of formal methods such as the B Method. (Hint: you may wish to illustrate your answer by referring to the Y2K bug.)

*(8 marks)*

4. In the **Square** machine, we formed an approximate square root of a given number by using a non-deterministic approximation *approx*, given by

```
sqrt <-- ApproxSqrt(num) =  
  PRE num : NAT  
  THEN ANY approx  
    WHERE approx : NAT &  
      approx*approx <= num &  
      num < (approx+1)*(approx+1)  
    THEN sqrt := approx  
  END  
END
```

Explain why this use of non-determinism is appropriate, and what must be done in refining this specification towards an implementation.

*(8 marks)*

5. Write a fragment of a B machine specification to show the use of a *deferred set*. Explain the difference between a *deferred set* and a *set parameter*.

(8 marks)

6. If we were modelling a taxi fleet company we might have three variables, *drivers*, *taxis* and *assigned* constrained by

$$\begin{aligned} \textit{drivers} &\in \mathbb{P} \textit{DRIVERS} \\ \textit{taxis} &\in \mathbb{P} \textit{TAXIS} \\ \textit{assigned} &\in \textit{drivers} \succ\!\!\rightarrow \textit{taxis} \end{aligned}$$

where *drivers* is the set of drivers working for the company, *taxis* is the set of taxis owned by the company, and *assigned* is a function recording the assignment of drivers to taxis.

The arrow  $\succ\!\!\rightarrow$  denotes a *partial injective* function.

- (a) Describe in your own words what a *partial injective* function is (that is, do **not** give a formal description).

(4 marks)

- (b) Why is *assigned* a partial function?

(1 marks)

- (c) Why is *assigned* an injective function?

(1 marks)

- (d) Specify the drivers who are currently assigned.

(1 marks)

(e) Specify the taxis that are currently unassigned.

*(1 marks)*

7. Perform the following predicate transformations and simplify the resultant predicate:

(8 marks)

(a)  $[x := x + 1]x < y + 1$

(b)  $[x := a + b, y := c + d] \exists p. p = \frac{x}{y} \wedge p < 100$

(c)  $[x > y \Rightarrow x := x - 1]y \leq x$

(d)  $[\mathbf{CHOICE} \ p := \mathit{new} \ \mathbf{OR} \ \mathit{users} := \mathit{users} - \{p\} \ \mathbf{END}] \ p \notin \mathit{users}$

8. Compute the proof obligations for the operation **ToGreen** in the following specification:

**MACHINE** *SimpleTwoWay*

**SETS**

$DIRECTION = \{ NorthSouth, EastWest \};$

$LIGHT = \{ Red, Green, Amber \}$

**VARIABLES**

*lights*

**INVARIANT**

$lights \in DIRECTION \rightarrow LIGHT \wedge$

$(lights (NorthSouth) \in \{ Green, Amber \} \Rightarrow lights (EastWest) = Red) \wedge$

$(lights (EastWest) \in \{ Green, Amber \} \Rightarrow lights (NorthSouth) = Red)$

**INITIALISATION**

$lights := \{ NorthSouth \mapsto Red, EastWest \mapsto Red \}$

**OPERATIONS**

**ToRed** (*dir*)  $\hat{=}$

**PRE**  $dir \in DIRECTION \wedge lights (dir) = Amber$

**THEN**  $lights (dir) := Red$

**END ;**

**ToGreen** (*dir*)  $\hat{=}$

**PRE**  $dir \in DIRECTION \wedge lights (dir) = Red \wedge$

$(dir = NorthSouth \Rightarrow lights (EastWest) = Red) \wedge$

$(dir = EastWest \Rightarrow lights (NorthSouth) = Red)$

**THEN**  $lights (dir) := Green$

**END ;**

**ToAmber** (*dir*)  $\hat{=}$

**PRE**  $dir \in DIRECTION \wedge lights (dir) = Green$

**THEN**  $lights (dir) := Amber$

**END**

**END**

(Write your answer overleaf)

(8 marks)

*(Question 8 Answer Page)*

9. In an example discussed in lectures, the specification of a *SimpleLibrary* machine contained the operation

```
MACHINE SimpleLibrary ( BOOK , maxuser )
...
OPERATIONS
  AddBook ( book ) =
    PRE book ∈ BOOK ∧
        book ∉ books_in_library
    THEN books_in_library := books_in_library ∪ {book}
    END ;
...
END
```

---

The following B fragments define a refinement to *SimpleLibrary*

```
MACHINE SimpleLibraryAPI ( BOOK , maxuser )
CONSTRAINTS maxuser ∈ ℕ1
INCLUDES SimpleLibrary ( BOOK , maxuser )
SETS
  RESPONSE = { OK , BookInLibrary , NoNewUsers , NotRegisteredUser ,
                BookNotForLoan , BookNotOnLoan }
OPERATIONS
...
END
```

---

Define a **robust** operation  $response \leftarrow AddBookR(book)$  that has a trivial precondition, adds the book *book* to the library with response *OK* if it is not already in *books\_in\_library*, and otherwise returns the response *BookInLibrary*.

(8 marks)

(You may continue your answer overleaf)

*(Question 9 Answer continued)*

10. The following tutorial example defines the *BagMath* machine. Explain i) the definition *Bag*, and ii) the use of the 3 constants *emptyBag*, *BagCount*, *BagUnion*. You may assume that the machine *Value\_TYPE* defines the set *VALUE*.

**MACHINE** *BagMath*

**SEES** *Value\_TYPE*

**CONSTANTS**

*emptyBag* , *BagCount* , *BagUnion*

**PROPERTIES**

$$\begin{aligned}
 & \textit{emptyBag} \in \textit{VALUE} \rightarrow \{ 0 \} \wedge \\
 & \textit{BagCount} \in \textit{Bag} ( \textit{VALUE} ) \rightarrow ( \textit{VALUE} \rightarrow \mathbb{N} ) \wedge \\
 & \forall \textit{bb} . ( \textit{bb} \in \textit{Bag} ( \textit{VALUE} ) \Rightarrow \textit{BagCount} ( \textit{bb} ) = \textit{emptyBag} \Leftarrow \textit{bb} ) \wedge \\
 & \textit{BagUnion} \in \textit{Bag} ( \textit{VALUE} ) \times \textit{Bag} ( \textit{VALUE} ) \rightarrow \textit{Bag} ( \textit{VALUE} ) \wedge \\
 & \forall ( \textit{b1} , \textit{b2} ) . ( \textit{b1} \in \textit{Bag} ( \textit{VALUE} ) \wedge \textit{b2} \in \textit{Bag} ( \textit{VALUE} ) \Rightarrow \\
 & \quad \textit{BagUnion} ( \textit{b1} , \textit{b2} ) = \{ \textit{vv} , \textit{nn} \mid \textit{vv} \in \textit{dom} ( \textit{b1} ) \cup \textit{dom} ( \textit{b2} ) \wedge \\
 & \quad \quad ( \textit{nn} \in \mathbb{N}_1 \wedge \textit{nn} = \textit{BagCount} ( \textit{b1} ) ( \textit{vv} ) + \textit{BagCount} ( \textit{b2} ) ( \textit{vv} ) ) \} )
 \end{aligned}$$

**DEFINITIONS**

$$\textit{Bag} ( X ) \hat{=} X \rightarrow \mathbb{N}_1$$

**END**

(8 marks)