FACULTY OF
SCIENCE & TECHNOLOGY

Department of Computer Science

2015 – 2016

Code:	ECSE610	**Level:** 6	**Semester:** 1
Title:	Formal Specification		
Date:	12th May 2016		
Time:	10:00 – 12:00		

INSTRUCTIONS TO CANDIDATES

Answer ALL questions in Section A and TWO questions from Section B.
Section A is worth a total of 50 marks.
Each question in section B is worth 25 marks.
The B-Method's Abstract Machine Notation (AMN) is given in Appendix B.

Section A

Answer ALL questions from this section.
You may wish to consult the B-Method notation given in Appendix B.

Question 1

(a) The building block of a B-method specification is the concept of an *Abstract Machine (AM)*. Explain what a B Abstract Machine is, in particular, you should answer the following questions:

- What is it similar to?
- What is it a specification of?
- What are its main logical parts?

[6 marks]

(b) Explain the purpose of the following B Abstract Machine *clauses* and illustrate their meaning by giving an example for each clause.

- `SETS`
- `CONSTANTS`
- `PROPERTIES`
- `VARIABLES`
- `INVARIANT`
- `INITIALISATION`
- `OPERATIONS`

[14 marks]

[TOTAL 20]

Question 2

You are given the following collection of B set and function declarations:

$$SHAPE = \{ Oval, Circle, Triangle, Rectangle, Square, Rhombus, Pentagon, Hexagon \}$$

$$Quadrilaterals \in \mathbb{P}(SHAPE)$$
$$Quadrilaterals = \{ Rectangle, Square, Rhombus \}$$

$$NonPolygons \in \mathbb{P}(SHAPE)$$
$$NonPolygons = \{ Oval, Circle \}$$

$$edges \in SHAPE \rightarrow \mathbb{N}$$
$$edges = \{ Oval \mapsto 1, Circle \mapsto 1, Triangle \mapsto 3, Rectangle \mapsto 4, Square \mapsto 4, Rhombus \mapsto 4, Pentagon \mapsto 5, Hexagon \mapsto 6 \}$$

Evaluate the following expressions:

(a) $Quadrilaterals \cup NonPolygons$ **[1 mark]**

(b) $Quadrilaterals \setminus \{ Rhombus \}$ **[1 mark]**

(c) $\mathrm{card}(edges)$ **[1 mark]**

(d) $\mathrm{dom}(edges)$ **[1 mark]**

(e) $\mathrm{ran}(edges)$ **[1 mark]**

(f) $edges(Hexagon)$ **[1 mark]**

(g) $\mathbb{P}(NonPolygons)$ **[2 marks]**

(h) $edges \rhd \{ 4 \}$ **[2 marks]**

(i) $NonPolygons \lhd edges$ **[2 marks]**

(j) $(Quadrilaterals \cup NonPolygons) \lhd\!\!\!\!- edges$ **[3 marks]**

[TOTAL 15]

Question 3

Given the following B declarations of the set of $Letters$ and the relations R_1, R_2 & R_3:

$$Letter = \{ a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z \}$$

$$R_1 \in Letter \leftrightarrow \mathbb{N}$$
$$R_1 = \{ a \mapsto 1,\ b \mapsto 1,\ c \mapsto 3,\ d \mapsto 2,\ e \mapsto 4,\ f \mapsto 4,\ g \mapsto 5,\ h \mapsto 6 \}$$

$$R_2 \in Letter \leftrightarrow \mathbb{N}$$
$$R_2 = \{ a \mapsto 1,\ b \mapsto 1,\ b \mapsto 2,\ c \mapsto 3,\ d \mapsto 2 \}$$

$$R_3 \in \mathbb{N} \leftrightarrow Letter$$
$$R_3 = \{ 1 \mapsto x,\ 2 \mapsto y,\ 4 \mapsto z \}$$

(a) Evaluate the following expressions:

(i) $R_1 [\{ a,\ c,\ e \}]$ **[2 marks]**

(ii) $R_3 \lhd\!\!\!- \{ 0 \mapsto w,\ 4 \mapsto a \}$ **[3 marks]**

(iii) $R_2 ; R_3$ **[4 marks]**

(b) For each of the relations R_1, R_2 and R_3 state whether it is just a *relation* or is also a *function*. In addition, if you decide that one of these relations is a function then state what kind of function it is, e.g. partial, total, injective, etc. **[6 marks]**

[TOTAL 15]

Section B

Answer TWO questions from this section.
You may wish to consult the B-Method notation given in Appendix B.

Question 4

Write a B-Method machine that specifies a *queue* of people waiting to be "served". For example, at a bank, supermarket checkout, etc. Due to the lack of available space the queue has a maximum permitted length.
Your B machine should deal with error handling where required and should include the following:

(a) Any sets, constants and variables, and any state invariant that the *queue* requires. **[9 marks]**

(b) The queuing operations:

(i) $JoinQueue$ – a new person joins the end of the *queue*. **[6 marks]**

(ii) $GetServed$ – the next person gets "served", i.e., leaves the front of the *queue*. **[6 marks]**

(iii) $QueueStatus$ – reports via a suitable message whether the queue is *empty*, *full* or *neither full or empty*. **[4 marks]**

[TOTAL 25]

Question 5

Appendix A contains the `Library` B machine, this specifies a simple book lending library.
The library has a catalogue of book titles (`BOOK`) and lends individual copies of each book (`COPY`) to its readers (`READER`).
The library's system holds the following information about its books and readers:

- The book title for each book copy (`copyof`).
- The books each reader has previously read (`hasread`).
- If a reader is currently reading a book then it records which copy he/she is reading (`reading`).

The system provides the following operations:

- Recording that a reader has started/finished reading a book.
- Check if a reader is currently reading a book; what book he/she is reading; has he/she read a book.

With reference to the `Library` B machine answer the following questions.

(a) The `Library`'s invariant, is given in the `INVARIANT` clause:

```
15  INVARIANT
16    hasread : READER <-> BOOK &              /* Inv-1 */
17    reading : READER >+> COPY &              /* Inv-2 */
18    (reading ; copyof) /\ hasread = {}       /* Inv-3 */
```

Using "plain English" only, answer the following questions:

(i) In `Inv-1` explain what the use of a *relation* `<->` ($\leftrightarrow$) means about the relationship between readers and books. Why would it not make sense to use a function? **[4 marks]**

(ii) In `Inv-2` explain what the choice of a *partial injection* `>+>` ($\rightarrowtail$) means regarding how many books a reader can read at any one time and how many readers can read the same book? What would it mean if this was changed to a relation? **[4 marks]**

(iii) Explain what `Inv-3` means. **[2 marks]**

[Continued Overleaf]

(b) Explain in "plain English" the meaning of the *preconditions* for the operations:

(i) `startReading` **[4 marks]**

(ii) `finishReading` **[3 marks]**

(iii) `currentlyReading` **[2 marks]**

(c) Draw the *Structure Diagram* for the `Reading` machine. **[6 marks]**

[TOTAL 25]

Question 6

(a) The B-Method is used to develop software systems. B is suitable for this task because it allows a system designer to specify important aspects of a system. Within this context of using a B machine to specify a system, answer the following questions.

(i) There are three categories of states that a system (B machine) can be in, what are they? Illustrate your answer by means of a diagram. **[4 marks]**

(ii) What is the role of the state invariant and what is its relationship to the system states? **[2 marks]**

(iii) Explain what important property the initial value(s) of a B machine's variables must satisfy. **[2 marks]**

(iv) When specifying an operation it is usually necessary to define an explicit precondition for the operation using PRE. What are preconditions and what is their purpose? **[4 marks]**

(v) If the specification of a machine's operation is required to be *"total"*, explain what this means. **[2 marks]**

(b) Given an abstract B machine specification, such as the following:

```
MACHINE Name

  SETS            Sets
  CONSTANTS       Consts
  PROPERTIES      Props
  VARIABLES       Vars
  INVARIANT       Inv
  INITIALISATION  Init

  OPERATIONS

  yy <-- op( xx ) =
         PRE  PC
         THEN Subst
         END ;
  ...
END
```

[Continued Overleaf]

Particular informal claims are usually made about its *correctness*. For example, it makes sense and is coherent; it can be instantiated; its operations perform correctly. These implicit claims are formalised in the B-Method by means of logical *"proof obligations"* on a B machine.

With reference to the above B machine describe the three main types of *proof obligations*:

(i) Data Proof Obligation **[3 marks]**

(ii) Initialisation Proof Obligation **[3 marks]**

(iii) Operation Proof Obligation **[5 marks]**

[TOTAL 25]

Appendix A. Library B Machine

The following is a B Machine – `Library` that specifies a simple book lending library.

```
1    MACHINE  Library
2
3      SETS
4        READER ;  BOOK ;  COPY ;  RESPONSE = { yes, no }
5
6      CONSTANTS
7        copyof
8
9      PROPERTIES
10       copyof : COPY -->> BOOK          /* Total Surjection */
11
12     VARIABLES
13       hasread, reading
14
15     INVARIANT
16       hasread : READER <-> BOOK &       /* Relation */
17       reading : READER >+> COPY &       /* Partial Injection */
18       (reading ; copyof) /\ hasread = {}
19
20     INITIALISATION
21       hasread := {} || reading := {}
22
23     OPERATIONS
24
25     startReading( rr, cc ) =
26       PRE
27           rr : READER  &  cc : COPY        &
28           copyof(cc) /: hasread[ { rr } ]  &
29           rr /: dom(reading)               &
30           cc /: ran(reading)
31       THEN
32          reading := reading \/ { rr |-> cc }
33       END ;
34
```

[Continued on next page.]

```
35      finishReading( rr, cc ) =
36        PRE
37           rr : dom( reading ) &
38           rr : READER  &  cc : COPY  &  reading(rr) = cc
39        THEN
40           hasread :=  hasread \/ { rr |-> copyof(cc) }  ||
41           reading :=  { rr } <<| reading
42        END ;
43
44      resp <-- isReading( rr ) =
45        PRE
46           rr : READER
47        THEN
48          IF   ( rr : dom(reading) )
49          THEN resp := yes
50          ELSE resp := no
51          END
52        END ;
53
54      bb <-- currentlyReading(rr) =
55        PRE
56           rr : READER  &  rr : dom(reading)
57        THEN
58           bb := copyof( reading(rr) )
59      END ;
60
61      resp <-- hasReadBook( rr, bb ) =
62        PRE
63           rr : READER  &  bb : BOOK
64        THEN
65          IF ( bb : hasread[ { rr } ] )
66          THEN resp := yes
67          ELSE resp := no
68          END
69        END
70
71    END /* Library */
```

Appendix B. B-Method's Abstract Machine Notation (AMN)

The following tables present AMN in two versions: the "pretty printed" symbol version & the ASCII machine readable version used by the B tools: *Atelier B* and *ProB*.

B.1 AMN: Number Types & Operators

B Symbol	ASCII	Description
$\mathbb{N}$	`NAT`	Set of natural numbers from 0
$\mathbb{N}_1$	`NAT1`	Set of natural numbers from 1
$\mathbb{Z}$	`INTEGER`	Set of integers
$\mathrm{pred}(x)$	`pred(x)`	predecessor of x
$\mathrm{succ}(x)$	`succ(x)`	successor of x
$x + y$	`x + y`	x plus y
$x - y$	`x - y`	x minus y
$x * y$	`x * y`	x multiply y
$x \div y$	`x div y`	x divided by y
$x \bmod y$	`x mod y`	remainder after x divided by y
$x \mathbin{\widehat{}} y$	`x ** y`	x to the power y, x^y
$\min(A)$	`min( A )`	minimum number in set A
$\max(A)$	`max( A )`	maximum number in set A
$x \mathinner{..} y$	`x .. y`	range of numbers from x to y inclusive

B.2 AMN: Number Relations

B Symbol	ASCII	Description
$x = y$	`x = y`	x equal to y
$x \neq y$	`x /= y`	x not equal to y
$x < y$	`x < y`	x less than y
$x \leq y$	`x <= y`	x less than or equal to y
$x > y$	`x > y`	x greater than y
$x \geq y$	`x >= y`	x greater than or equal to y

B.3 AMN: Set Definitions

B Symbol	ASCII	Description
$x \in A$	`x : A`	x is an element of set A
$x \notin A$	`x /: A`	x is not an element of set A
$\varnothing$, $\{\ \}$	`{}`	Empty set
$\{\ 1\ \}$	`{ 1 }`	Singleton set (1 element)
$\{\ 1, 2, 3\ \}$	`{ 1, 2, 3 }`	Set of elements: 1, 2, 3
$x\,..\,y$	`x .. y`	Range of integers from x to y inclusive
$\mathbb{P}(A)$	`POW(A)`	Power set of A
$\mathbb{P}_1(A)$	`POWn(A)`	Power set of Non-empty sets A
$\mathrm{card}(A)$	`card(A)`	Cardinality, number of elements in set A

B.4 AMN: Set Operators & Relations

B Symbol	ASCII	Description
$A \cup B$	`A \/ B`	Union of A and B
$A \cap B$	`A /\ B`	Intersection of A and B
$A \setminus B$	`A \ B`	Set subtraction of A and B
$\bigcup AA$	`Union AA`	Distributed union of AA
$\bigcap AA$	`Intersection AA`	Distributed intersection of AA
$A \subseteq B$	`A <: B`	A is a subset of or equal to B
$A \not\subseteq B$	`A /<: B`	A is not a subset of or equal to B
$A \subset B$	`A <<: B`	A is a strict subset of B
$A \not\subset B$	`A /<<: B`	A is not a strict subset of B
$\{\ x \mid x \in TS \wedge C\ \}$	`{ x | x : TS & C }`	Set comprehension

B.5 AMN: Logic

B Symbol	ASCII	Description
$\neg P$	`not P`	Logical negation (not) of P
$P \wedge Q$	`P & Q`	Logical and of P, Q
$P \vee Q$	`P or Q`	Logical or of P, Q
$P \Rightarrow Q$	`P => Q`	Logical implication of P, Q
$P \Leftrightarrow Q$	`P <=> Q`	Logical equivalence of P, Q
$\forall\, xx \cdot (P \Rightarrow Q)$	`!(xx).(P => Q)`	Universal quantification of xx over $(P \Rightarrow Q)$
$\exists\, xx \cdot (P \wedge Q)$	`#(xx).(P & Q)`	Existential quantification of xx over $(P \wedge Q)$
$TRUE$	`TRUE`	Truth value $TRUE$.
$FALSE$	`FALSE`	Truth value $FALSE$
$BOOL$	`BOOL`	Set of boolean values $\{ TRUE, FALSE \}$
$bool(P)$	`bool(P)`	Convert predicate P into $BOOL$ value

B.6 AMN: Ordered Pairs & Relations

B Symbol	ASCII	Description
$X \times Y$	`X * Y`	Cartesian product of X and Y
(x, y)	`(x, y)`	Ordered pair
$x \mapsto y$	`x |-> y`	Ordered pair, (maplet)
$\mathrm{prj}_1(S,T)(x,y)$	`prj1(S,T)(x, y)`	Ordered pair projection function
$\mathrm{prj}_2(S,T)(x,y)$	`prj2(S,T)(x, y)`	Ordered pair projection function
$\mathbb{P}(X \times Y)$	`POW(X * Y)`	Set of relations between X and Y
$X \leftrightarrow Y$	`X <-> Y`	Set of relations between X and Y
$\mathrm{dom}(R)$	`dom(R)`	Domain of relation R
$\mathrm{ran}(R)$	`ran(R)`	Range of relation R

B.7 AMN: Relations Operators

B Symbol	ASCII	Description
$A \lhd R$	`A <| R`	Domain restriction of R to the set A
$A \lhd\!\!\!\!- R$	`A <<| R`	Domain subtraction of R by the set A
$R \rhd B$	`R |> B`	Range restriction of R to the set B
$R \rhd\!\!\!\!- B$	`R |>> B`	Range anti-restriction of R by the set B
$R[B]$	`R[B]`	Relational Image of the set B of relation R
$R_1 <\!\!+ R_2$	`R1 <+ R2`	R_1 overridden by relation R_2
$R\,;Q$	`R ; Q`	Forward Relational composition
$\mathrm{id}(X)$	`id(X)`	Identity relation
R^{-1}	`R~`	Inverse relation
R^n	`iterate(R,n)`	Iterated Composition of R
R^+	`closure1(R)`	Transitive closure of R
R^*	`closure(R)`	Reflexive-transitive closure of R

B.8 AMN: Functions

B Symbol	ASCII	Description
$X \nrightarrow Y$	`X +-> Y`	Partial function from X to Y
$X \rightarrow Y$	`X --> Y`	Total function from X to Y
$X \rightarrowtail Y$	`X >+> Y`	Partial injection from X to Y
$X \rightarrowtail Y$	`X >-> Y`	Total injection from X to Y
$X \twoheadrightarrow Y$	`X +->> Y`	Partial surjection from X to Y
$X \twoheadrightarrow Y$	`X -->> Y`	Total surjection from X to Y
$X \rightarrowtail\!\!\!\!\!\twoheadrightarrow Y$	`X >->> Y`	(Total) Bijection from X to Y
$f <\!\!+ g$	`f <+ g`	Function f overridden by function g

B.9 AMN: Sequences

B Symbol	ASCII	Description
$[\,]$	`[]`	Empty Sequence
$[\,e1\,]$	`[ e1 ]`	Singleton Sequence
$[\,e1,\ e2\,]$	`[ e1, e2 ]`	Constructed (enumerated) Sequence
$\mathrm{seq}(X)$	`seq( X )`	Set of Sequences over set X
$\mathrm{seq}_1(X)$	`seq1( X )`	Set of non-empty Sequences over set X
$\mathrm{iseq}(X)$	`iseq( X )`	Set of injective Sequences over set X
$\mathrm{iseq}_1(X)$	`iseq1( X )`	Set of non-empty injective Sequences over set X
$\mathrm{perm}(X)$	`perm( X )`	Set of bijective Sequences (permutations) of set X
$\mathrm{size}(s)$	`size( s )`	Size (length) of Sequence s

B.10 AMN: Sequences Operators

B Symbol	ASCII	Description
$s \frown t$	`s^t`	Concatenation of Sequences s & t
$e \rightarrow s$	`e -> s`	Insert element e to front of sequence s
$s \leftarrow e$	`s <- e`	Append element e to end of sequence s
$rev(s)$	`rev( s )`	Reverse of sequence s
$first(s)$	`first( s )`	First element of sequence s
$last(s)$	`last( s )`	Last element of sequence s
$front(s)$	`front( s )`	Front of sequence s, excluding last element
$tail(s)$	`tail( s )`	Tail of sequence s, excluding first element
$conc(SS)$	`conc(SS)`	Concatenation of sequence of sequences SS
$s \uparrow n$	`s /| n`	Take first n elements of sequence s
$s \downarrow n$	`s \\/ n`	Drop first n elements of sequence s

B.11 AMN: Miscellaneous Symbols & Operators

B Symbol	ASCII	Description
$var := E$	`var := E`	Assignment
$var :\in A$	`var :: E`	Nondeterministic assignment an element of set A to var
$S1 \| \| S2$	`S1 || S2`	Parallel execution of $S1$ and $S2$

B.12 AMN: Operation Statements

B.12.1 Assignment Statements

```
xx := xxval

xx, yy, zz := xxval, yyval, zzval

xx := xxval  ||  yy := yyval
```

B.12.2 Deterministic Statements

```
skip

BEGIN  S  END

PRE  PC  THEN  S  END

IF  B  THEN  S  END

IF  B  THEN  S1  ELSE  S2  END

IF  B1  THEN  S1  ELSIF B2  THEN  S2  ELSE  S3  END

CASE  E  OF
  EITHER  v1  THEN  S1
  OR      v2  THEN  S2
  OR      v3  THEN  S3
  ELSE
          S4
END

LET  xx  BE  xx = E  IN  S  END
```

B.12.3 Nondeterministic Statements

```
xx :: AA

ANY xx WHERE P THEN S END

CHOICE  S1  OR  S2  OR  S3  END

SELECT  B1 THEN S1
  WHEN  B2 THEN S2
  WHEN  B3 THEN S3
  ELSE
        S4
END
```

B.13 B Machine Clauses

```
MACHINE Name( Params )

  CONSTRAINTS    Cons

  EXTENDS        M1, M2, ...

  INCLUDES       M3, M4, ...
  PROMOTES       op1, op2, ...

  SEES           M5, M6, ...
  USES           M7, M8, ...

  SETS            Sets
  CONSTANTS       Consts
  PROPERTIES      Props

  VARIABLES       Vars
  INVARIANT       Inv
  INITIALISATION  Init

  OPERATIONS

    yy <-- op( xx ) =
           PRE  PC
           THEN Subst
           END ;
    ...
END
```