Universiti Malaysia PAHANG

Engineering • Technology • Creativity

FACULTY OF COMPUTER SYSTEMS & SOFTWARE ENGINEERING

ANSWERS SCHEMA FOR MIDDLE TERM TEST

COURSE	**:**	**FORMAL METHODS**
COURSE CODE	**:**	**BCS2213**
LECTURER	**:**	**VITALIY MEZHUYEV**
DATE	**:**	**15 APRIL 2015**
DURATION	**:**	**2 HOURS**
SESSION/SEMESTER	**:**	**SESSION 2014/2015 SEMESTER II**
PROGRAMME CODE	**:**	**BCS / BCN / BCG**

QUESTION 1 **[27 Marks]**

a) Using predicate logic translate the following sentences into logical expressions.

(i) Every software class inherits a superclass.

[3 Mark]

$\forall x \exists y Lxy$

(ii) A superclass is inherited by its classes.

[3 Mark]

$\exists x \forall y Lyx$

(iii) Every recursion function calls itself.

[3 Mark]

$\forall x Lxx$

(iv) A function is called by other function.

[3 Mark]

$\exists x \exists y Lyx$

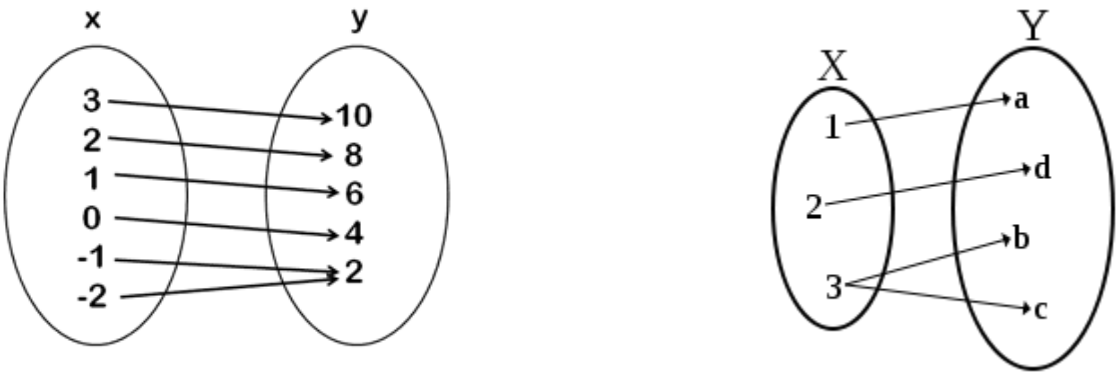
(vi) All the functions call all the other functions.

[3 Mark]

$\forall x \forall y Lxy$

b) Which of the diagrams a), b) or both, as shown on Figure 1, define functions from X to Y? Justify your answer.

[3 Marks]



a) **b)**

Figure 1

Diagram a) define a total function X→Y.

Diagram b) does not define a function because an element 3 in the set X has two possible values (b, c) in the set Y.

c) What is underlying mathematics for Z and TLA.

[3 Marks]

- Z is based on the mathematical notation based on *axiomatic* set theory, lambda calculus and first order predicate logic.
- TLA uses first order predicate logic and set theory for expressing ordinary mathematics. TLA expands ordinary mathematics by temporal operators (like *next*, *always*, *eventually*, etc.).

d) Specify *types* of systems, can be modelled with Z and TLA notations.

[3 Marks]

All notations are intended for modelling computer discrete systems, i.e. hardware and software.

- Z can be used to represent structure of software code: data types, procedures, functions, modules, classes, objects etc.
- TLA is applicable for specifying *distributed*, *concurrent* and *asynchronous* systems.

e) Explain the difference between Z schemas, marked by Δ (Delta) and Ξ (Xi) symbols.

[3 Marks]

- Using Δ mean state change of a schema
- Using Ξ means no state change

QUESTION 2 **[38 Marks]**

Develop model of an elevator control system in Z notation. Comment your statements.

An elevator system to be installed in a building with m floors. The problem concerns the logic to move elevators between floors according to the following constraints:

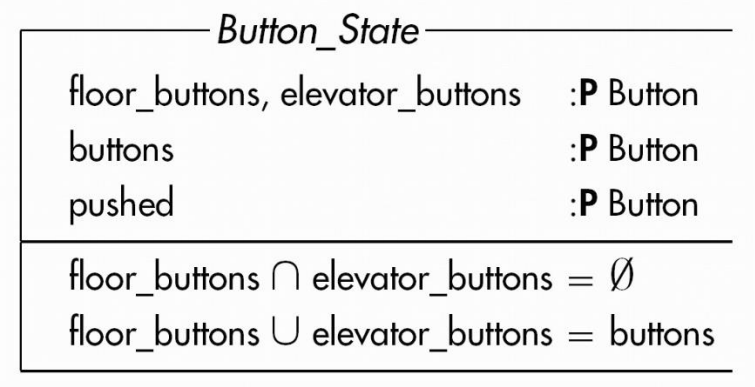
- An elevator has a set of buttons, one for each floor. These illuminate when pressed and cause the elevator to visit the corresponding floor. The illumination is cancelled when the corresponding floor is visited by the elevator.
- Each floor has two buttons (except the top and bottom floors), one to request upward travel and one to request downward travel. These buttons illuminate when pressed. The illumination is cancelled when an elevator visits the floor and is either moving in the desired direction or has no outstanding requests.
- When an elevator has no requests to service, it should remain at its final destination with its doors closed and await further requests.

Marks distribution:

- Buttons state schema (done) **[15 Marks]**
- Init schema **[5 Marks]**
- Buttons operation schema **[15 Marks]**
- Comments **[3 Marks]**

Possible subsets of the set *Button*

- ☐ The floor buttons
- ☐ The elevator buttons
- ☐ buttons (the set of all buttons in the elevator problem)
- ☐ pushed (the set of buttons that have been pushed)

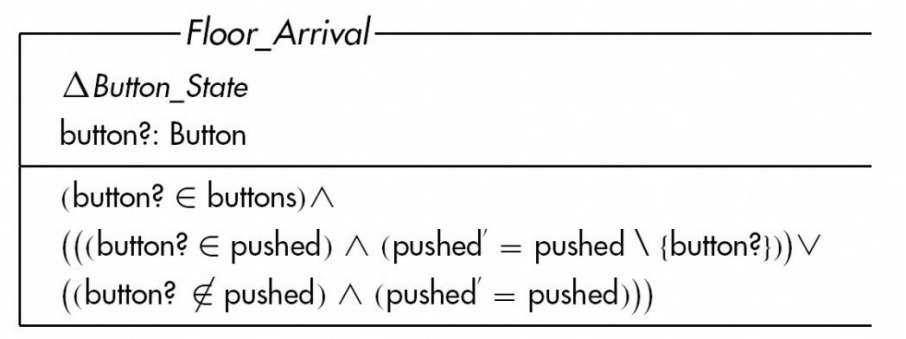


Initial State

B*utton_Init* ≡ [*Button_State*' | pushed' = ∅]

Operations

- Button pushed for first time is turned on, and added to set *pushed*
- Without third precondition, results would be unspecified



- If elevator arrives at a floor, the corresponding button(s) must be turned off
- The solution does not distinguish between up and down floor buttons

QUESTION 3 **[35 Marks]**

Develop TLA specifications of the following situation. You can use TLATEX or ASCII notation. Comment your statements.

When rain comes, the cat goes in the room or in the basement. When the cat is in the room, the mouse goes in the hole, and the cheese is put in the fridge. If the cheese is on the table, and the cat is in the basement, then the mouse is in the room. Now comes the rain, and cheese is on the table.

```
EXTENDS Naturals, TLC

\* Define Cat, Mouse, Cheese as Variables
VARIABLES Cat, Mouse, Cheese

\* Define Room, Base, Table, Hole, Fridge as Constants
CONSTANT  Room, Base, Table, Hole, Fridge

\* Definition of type invariant and initial condition
TypeInvariant ==    /\ Cheese \in {Table, Fridge}
                    /\ Cat \in {Room, Base}
                    /\ Mouse \in {Room, Hole}

Init == /\ Cheese = Table

\* Definition of actions
Next1 ==    /\ Cat = Room
            /\ Mouse' = Hole
            /\ Cheese' = Fridge
            /\ UNCHANGED <<Cat>>

Next2 ==    /\ Cat = Base
            /\ Mouse' = Room
            /\ Cheese = Table
            /\ UNCHANGED <<Cat, Cheese>>

\* Specification and theorem to prove
Spec == Init /\ [][Next1 \/ Next2 ] _ <<Cat, Mouse, Cheese>>
THEOREM Spec => []TypeInvariant
```