1. [2 points] a) Draw the transition system obtained by synchronising the following two transition systems with respect to the set \( \text{Sync} = \{(!m_1,?m_1),(!m_2,?m_2),(?a_1,!a_1),(?a_2,!a_2)\} \)

b) The transition system obtained describes the communication protocol between a sender and a receiver. The sending of messages is denoted by the actions !m_1 and !m_2, while the receiving of messages is denoted by the actions ?m_1 and ?m_2, respectively. Similarly, the sending of acknowledgements is denoted by !a_1 and !a_2, while their reception is denoted by ?a_1 and ?a_2, respectively.

Let \( p \) be the atomic proposition “m1 has been communicated” and \( q \) be the atomic proposition “m2 has been communicated”. Write a CTL formula specifying that m2 will always be communicated after m1 has already been communicated.

c) If we label the state <s0,r1> of the synchronised transition system with the atomic proposition \( p \), and the state <s1,r2> with \( q \), which of these two states satisfy your CTL formula above.

2. [2 points] Using the labelling algorithm described in the course, give the set of all states of the following transition system satisfying the CTL formula E[p U AF q].
3. [1 point] Give a simple CTL model and a state satisfying the CTL formula EF→p but not satisfying ¬AF p.

4. [1 point] Give the definition of validity of the partial correctness triple [l φ l] c [l ψ l] and of the total correctness triple [l φ l] c [l ψ l].

5. [2 point] The execution of the command

   repeat { c } until b

is equivalent to c; while ¬b { c }. Derive from the proof system for partial correctness given below a partial correctness proof rule for this repeat-until command.

6. [3 point] Given the following command c

   while x ≠ y {
     if x < y then { y := y – x }
     else { x := x – y }
   }

prove (using a proof outline) the validity of the following partial correctness triple

   (l x ≥ 0 ∧ y ≥0 l) c (l x = y l)

Proof system for partial correctness:

1. (l φ l) skip (l φ l)

2. (l φ[e/x] l) x:= e (l φ l)

   (l φ l) c1 (l φ l) (l φ l) c2 (l ψ l)

3. --------------------------------------------

   (l φ l) c1 ; c2 (l ψ l)

   (l φ ∧ b l) c1 (l ψ l) (l φ ∧ ¬b l) c2 (l ψ l)

4. --------------------------------------------

   (l φ l) if b then { c1 } else {c2} (l ψ l)

   (l φ ∧ b l) c (l φ l)

5. --------------------------------------------

   (l φ l) while b do { c } (l φ ∧ ¬b l)

   φ ⇒ φ1 (l φ1 l) c (l ψ1 l) ψ1 ⇒ ψ

6. --------------------------------------------

   (l φ l) c (l ψ l)